

MERVA ESA Components



MERVA USE & Branch for Windows NT Installation and Customization Guide

Version 4 Release 1

MERVA ESA Components



MERVA USE & Branch for Windows NT Installation and Customization Guide

Version 4 Release 1

Note!

Before using this information and the product it supports, be sure to read the general information under “Appendix C. Notices” on page 203.

Fifth Edition, September 2001

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

This edition replaces SH12-6335-03.

Changes to this edition are marked with a vertical bar.

© Copyright International Business Machines Corporation 1999, 2001. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About This Book	vii
Who Should Read This Book	vii
How This Book is Organized	vii
Conventions and Terminology Used in This Book	vii

Part 1. Installing MERVA USE & Branch 1

Chapter 1. Planning for the Installation of MERVA USE & Branch 3

Introducing MERVA USE & Branch	3
MERVA Link – the Connection to Other MERVA Systems	3
SWIFT Link – the Connection to SWIFT	3
MERVA USE & Branch Instance	4
MERVA USE & Branch Authority	4
Multiple User Support	5
Hardware and Software Requirements	5
Hardware Requirements	5
Software Requirements	6

Chapter 2. Installing MERVA USE & Branch 9

Installing the Prerequisite Software	9
Installing the Program Files	10
Installing Service	14
Creating a MERVA Instance	14
Creating the MERVA Databases	15
Installing the MERVA License Keys	16
MERVA USE & Branch for Windows NT	16
MERVA USE & Branch for Windows NT with SWIFT Link	17
MERVA Message Processing Client for Windows NT	17
Adding Users	18
Overview	19
Adding an Initial MERVA User	19
Adding Administrators	20
Adding Other Users	21
Configuring the MERVA Control Process	22
Verifying the Installation	22
Settings and Status of MERVA Instance	23
Setting Up a MERVA User Administrator	23
Setting Up the Default Customization for MERVA USE & Branch for Windows NT without SWIFT Link	24
Uninstalling MERVA USE & Branch	24
Removing the MERVA Databases	24
Removing a MERVA Instance	25
Uninstalling the MERVA Program Files	25

Chapter 3. Running MERVA under Windows NT Server 4.0, Terminal Server Edition	27
Installation of License Use Runtime	27
Installation of Java Runtime Environment	27
Configuring Terminal Server Clients	27

Part 2. Customizing MERVA USE & Branch 29

Chapter 4. Starting MERVA USE & Branch Customization 31

Customization Procedure	31
The MERVA Customizer Window	32
Saving Changes and Stopping the Customization Mode	33
Generating Customization Reports	33

Chapter 5. Defining the Message Routing 35

Setting up Message Queues	36
Setting up a New Queue	36
Deleting a Queue	36
Defining Message Fields	37
Defining a New Field and Specifying Information for Fields	37
Changing Field Definitions	39
Deleting a Field	39
Defining Constants	39
Setting up Routing Conditions	40
Selecting Routing Conditions	40
Defining Routing Conditions	41
Creating Routing Conditions	41
Copying Routing Conditions from Other Queues	43
Debugging Routing Conditions	44

Chapter 6. Customizing the SWIFT Link 47

Defining Logical Terminal (LT) Identifiers	47
Defining Synonym LTs	48
Defining Queue Assignments for Logical Terminals	48
Defining the Network Data for SWIFT Link	49
Defining the FIN Copy Service	51
Hints to Set up Routing Related to FIN Copy Service	55

Chapter 7. Customizing the MERVA Link 57

Introduction to MERVA Link	57
MERVA Link Structure and Resources	57
Scheduling of MERVA Link	59
Defining Details for MERVA Link	59
Defining Details for Partner Nodes	60
Defining ASP Information	62

MERVA Link Remote Front-End Scenario	67
The CREATING System	67
The FRONTEND System	67
Returning the Acknowledged Message	67
Conditions for Generating and Sending a Status Report	69
Merging the Acknowledgment Information with the Original Message	69
Routing Conditions for the CREATING System	69
Routing Conditions for the FRONTEND System	71

Chapter 8. Customizing SWIFT USE 75

Defining Message Headers	75
Defining SWIFT Destinations	76
Understanding the USE Background Process	76
Defining Timeout	77
Defining BKE Start Conditions	78
Supporting the SWIFT USE Card Reader	78
Customizing Devices for the SWIFT USE Card Reader	78
Customizing TCP/IP for the Remote Card Reader Client	79
MERVA Card Reader Server	79

Chapter 9. Customizing Other MERVA Parameters 81

Customizing Printer Conditions	81
Automatic Printing	81
Selecting the Default Printer	82
The Message Print Separator	82
Maintaining Alarms	83
Currency Code Customization	84
Setting the Date Format	85
Destination Checking	86
Setting Logging Levels	86

Chapter 10. Exporting and Importing Customization Data 89

Using the Export and Import Commands	89
Export Command	89
Import Command	90
Components and Files	90
Exporting Customization Data	91
Importing Customization Data	91
Understanding the Keywords Used in the ASCII Files	93
Notational Convention	93
Alarm Maintenance: Keyword List	95
Automatic Print: Keyword List	96
Routing: Keyword List	97
MERVA Link Component: Keyword List	100
SWIFT Link Component: Keyword List	103
User Access Rights: Keyword List	107
The SWIFT USE Component: Keyword List	109

Part 3. Establishing Connections in MERVA 111

Chapter 11. Customizing TCP/IP for MERVA Applications 113

TCP/IP Services	113
TCP/IP Sockets Interface	113
TCP/IP Host Name and Port Number	113
MERVA Inetd Service	113
Customizing TCP/IP Services	114
Customizing Hosts Table	114
Customizing Client Network Services	115
Customizing MERVA Inetd Subservices	115

Chapter 12. MERVA Link 117

Customizing TCP/IP Services for MERVA Link	117
Customizing Hosts Table	117
Customizing Client Network Services	117
Customizing MERVA Inetd Service	117
TCP/IP Conversation Security	118
MERVA Link Outbound Conversation Security Information	118
MERVA Link Outbound Conversation Security	119
MERVA Link Inbound Conversation Security Information	119
MERVA Link Inbound Conversation Security	119
Customizing a Windows NT Communications Server for MERVA Link	120
VTAM APPC LU Definition	120
Node Configuration	120
Device Configuration	121
Connection Configuration	122
LU 6.2 Sessions	123
Automatically Starting Communications Server Resources	128
SNA APPC Conversation Security	128

Chapter 13. SWIFT Link 137

X.25 Adapter Configuration	137
Installing the ARTIC Support for Windows NT	137
Installing the ARTIC Support for X.25 on Windows NT	138
Configuring the ARTIC Support for X.25 on Windows NT	138

Part 4. Appendixes 143

Appendix A. MERVA Link Application User Exits 145

Data Types of the User-Exit Interface	145
Data Fields of the User-Exit Interface	147
MERVA Link Status-Report Header Fields	148
MERVA Link User-Exit Interface Functions	148
ENMGetField—Copy Data from the Specified Field	149
ENMPutField—Put Data to the Specified Field	150
ENMTraceFields—Write Information to Trace File	151
ENMWriteToLogFile—Write Information to Diagnosis Log and Message Console	152
ENMGenerateDefaultStatusReport—Generate the Status Report for SWIFT Messages	153

ENMCorrelateDefaultStatusReport—Update the Message Field	154	MQRCV: Messages Received from MQSeries	193
ENMGetStatusRepHeader—Copy the Status Header Information	155	MQSND1: Messages to Be Sent to MQSeries Awaiting SWIFT ACK or NAK	193
ENMPutStatusRepHeader—Change the Status Report Header Information.	156	MQSND2: Messages to Be Sent to MQSeries	193
ENMGetStatusRepData—Copy the Information of a Status Report Data Field	157	MQWAIT: Messages awaiting SWIFT ACK or NAK information	194
ENMPutStatusRepData—Copy the Data and the Identifiers of a Report Data Field	158	SL_IN: SWIFT Messages Loaded at the API	194
ENMIsStatusRepDataValid—Check if a Status Report Data Element Exists.	159	SLINCMS1: Incoming GPA Messages	194
MERVA Link Application User Exit Functions	160	SLINCMS2: Incoming FIN Messages.	194
ENMReadyToSend—Change Message Text or Related Data	161	SLMANAUT: Manual Authentication	194
ENMOutgoing—Send the Message or Status Report.	162	SLPRINT1: Messages to Be Printed	195
ENMGenerateSR—Generate the Message Type Specific Status Report	163	SLPRINT2: Messages to Be Printed	195
ENMConfirmed—Create Application or User-Specific Acknowledgment	164	SLRCVFIN: Message Received OK (Other Than 19x and 29x).	195
ENMIncomingAM—Message-Type Specific Processing	165	SLRCVNST: Message Received OK (Message Types 19x and 29x)	195
ENMIncomingSR—Update Message.	166	SLRCVSY: Distribution of Delivery or Non-Delivery Information	196
Generating a MERVA Link Application User Exit	166	SLRL1ACK: Acknowledged Messages	196
Defining a User Exit for MERVA Link	167	SLRNRM02, SLRURG02: Ready-to-Send FIN Normal/Urgent Messages	196
		SLRSYS01: Ready-to-Send GPA System Messages.	196
		SLRSYS02: Ready-to-Send FIN System Messages	197
		SMAUTH1: Messages Being Authorized	197
		SMCREATE: Newly Created Messages	197
		SMEDIT: Messages to Be Corrected	198
		SMINCMPT: Newly Created Messages and Messages Created But Incomplete	198
		SMVERIFY: Messages to Be Verified by Retyping	199
		TP2_ACK: Positively Acknowledged Telex Messages	199
		TP2_NAK: Negatively Acknowledged Telex Messages.	199
		TP2_RCV: Received Telex Messages	199
		TP2_SND: Ready-to-Send Telex Messages	200
		TP2_WAIT: Telex Messages Waiting for Acknowledgment	200
		UFROMSWF: USE Messages from SWIFT	200
		UNAKED: Negatively Acknowledged USE Messages	201
		USEINCMD: Incoming USE Commands	201
		USESEND: USE Messages to Send	201
		UWOPREAG: Incoming MT 960 without Pre-Agreement	201
Appendix B. Standard Routing	169		
Purpose Groups	170		
Standard Message Queues	171		
Message Parts	173		
Layout of Message Part MSGTXT4 FIELD (Telex Envelope Data).	174		
Telex Header	174		
Telex Information	175		
Fields	176		
Constants	178		
Routing Conditions	180		
Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link	180		
Routing Conditions for MERVA USE & Branch for Windows NT using MERVA Link	184		
Routing Conditions for MERVA USE & Branch for Windows NT using MERVA-MQI	187		
Attachment	187		
Routing Descriptions for Each Source Queue	191		
MBCRCERR: Messages That Failed CRC Checking	191		
MBDELETE: Messages To Be Deleted	191		
MBMERROR: Messages That Failed Routing	191		
MBRERROR: Messages That Caused an Internal Routing Failure.	191		
MLAKWAIT: Messages to Be Correlated with Incoming Status Reports.	191		
MLCNTRL: Message Control	192		
MLMERROR: Messages That Failed Normal Processing	192		
MLRECEIV: Messages Received from Another MERVA Link	192		
MLSEND: Messages to Be Sent to MERVA Link	192		
		Appendix C. Notices	203
		Trademarks	204
		Glossary of Terms and Abbreviations	207
		Bibliography.	219
		MERVA ESA Publications	219
		MERVA ESA Components Publications	219
		Other IBM Publications	219
		S.W.I.F.T. Publications	219
		Index	221

About This Book

This book describes the procedures required to install and customize the IBM licensed programs

- MERVA USE & Branch for Windows NT
- MERVA USE & Branch for Windows NT with SWIFT Link

Both programs are part of MERVA ESA Components Version 4 Release 1.

The book explains how to:

- Install MERVA from the distribution media
- Configure MERVA
- Create a MERVA instance and MERVA databases
- Customize message routing
- Configure the link to the SWIFT network
- Set up MERVA Link connections
- Configure background message printing
- Change system default settings

Who Should Read This Book

This book is written for administrators or developers responsible for the installation and customization of MERVA.

It is assumed throughout this book that you are familiar with:

- Windows NT
- The SWIFT network

How This Book is Organized

The first part of this book provides information about the installation of MERVA by giving a brief overview of the product and by describing what you have to consider when you install it for the first time. It also describes the installation and deinstallation procedure in detail. The second part describes how to customize MERVA. Part 3 tells you how to work with MERVA Link and SWIFT Link. The appendixes describe user exits and standard routing.

This book also contains a glossary of terms and abbreviations, a bibliography, and an index.

Conventions and Terminology Used in This Book

In this book, the following format conventions apply:

- Menu instructions

The following format shows you which item to select from a cascading menu:

Click **File** → **Open**.

For more than two items:

Click **File** → **Tools** → **User preferences**.

In this book, the following naming conventions apply:

- MERVA USE & Branch is used when the description applies to the following features at the same time:
 - MERVA USE & Branch for Windows NT
 - MERVA USE & Branch for Windows NT with SWIFT Link
- MERVA USE & Branch for Windows NT is used when the description applies only to MERVA USE & Branch for Windows NT.
- MERVA USE & Branch for Windows NT with SWIFT Link is used when the description applies only to MERVA USE & Branch for Windows NT with SWIFT Link.
- The term MERVA ESA used in this publication applies to MERVA products for both MVS and VSE.

Part 1. Installing MERVA USE & Branch

Chapter 1. Planning for the Installation of MERVA USE & Branch

This chapter gives you information that you need when you plan and prepare the installation of MERVA USE & Branch for the first time.

Introducing MERVA USE & Branch

MERVA USE & Branch has routing and queueing functions for financial messages. It also has interfaces to various financial networks, such as SWIFT, and it consists of different components.

MERVA USE & Branch for Windows NT consists of:

- MERVA Link to communicate with other MERVA USE & Branch systems
- Application Programming Interface (API)
- User and operator functions to control the system and process messages
- Background processes that have common services to all MERVA USE & Branch functions, such as message queue management, audit trail, and access security

MERVA USE & Branch for Windows NT with SWIFT Link consists of:

- The same components as MERVA USE & Branch for Windows NT
- SWIFT Link to communicate with the SWIFT network

MERVA Link – the Connection to Other MERVA Systems

The MERVA Link component allows you to exchange messages between MERVA systems on all platforms. This component lets you build a MERVA network and use the MERVA functions in various systems on the same or on a different platform. By using MERVA Link, you can, for example:

- Build a branch network. In this network, you prepare the messages in the branches that are connected to a central location by MERVA Link. The central location runs the communication links to external networks, such as SWIFT
- Use MERVA USE & Branch as a communication front end for a MERVA ESA system.
- Use MERVA USE & Branch to run all SWIFT User Security Enhancement (USE) functions. It can distribute the resulting and the bilateral keys to other MERVA systems using MERVA Link.

Note that you have to define the location for functions, the message flow, and the network topology if you want to build a MERVA network.

SWIFT Link – the Connection to SWIFT

MERVA USE & Branch for Windows NT with SWIFT Link provides a single link to the SWIFT network. X.25 is used as the communication protocol. Up to 16 Logical Terminals (LTs) of the same or of different institutions can share the link.

You can define different physical links in Windows NT and switch between the definitions by using logical names for the connections.

MERVA USE & Branch supports SWIFT USE services. SWIFT session and authentication keys can be distributed to and received from other MERVA systems. For detailed information refer to the *MERVA USE Administration Guide*.

By using TCP/IP, MERVA USE & Branch can connect SWIFT card readers to remote systems. This helps, for example, a SWIFT operator to access a SWIFT card reader in the office while MERVA USE & Branch runs on a remote system in another location. For detailed information on how to customize a TCP/IP connection for remote SWIFT USE card readers, refer to “Supporting the SWIFT USE Card Reader” on page 78.

MERVA USE & Branch Instance

Each MERVA USE & Branch system on a machine is called a MERVA instance. A MERVA instance consists of several MERVA databases containing, for example, customization data, message queues, user access rights, and correspondents data.

To identify a MERVA instance, you define a name for it. The name is contained in the first part of the Message Reference Number (MRN) of each message that is created in this instance. This helps you identify the source of a message if various MERVA USE & Branch systems exchange messages via MERVA Link.

Note that you have to authorize users to work with a MERVA instance.

MERVA USE & Branch Authority

The following authorities are valid in MERVA USE & Branch:

- MERVA System Administration (SYSADM) authority
- MERVA User Administration (USERADM) authority
- MERVA End User Administration (USER) authority

System Administration Authority

The SYSADM authority is the highest level of administrative authority.

With SYSADM authority you can:

- Start and stop the MERVA USE & Branch control process as a Windows NT service
- Start and stop MERVA USE & Branch in multiple user mode
- Start and stop MERVA USE & Branch in customization mode
- Create and remove a MERVA instance
- Create and remove a MERVA database
- Access each data in each table of each MERVA database

You have SYSADM authority only if you are a member of all of the following groups:

- Local administrator group
- MERVA administrator group **mervasys**
- MERVA user group, for example, **umerva1**

User Administration Authority

The USERADM authority is the second highest level of administrative authority.

With USERADM authority you can:

- Start and stop MERVA USE & Branch in multiple user mode if the MERVA USE & Branch control process is already running
- Start and stop MERVA USE & Branch in customization mode if the MERVA USE & Branch control process is already running

- Administer MERVA USE & Branch user rights

You have USERADM authority only if you are a member of all of the following groups:

- Users group
- MERVA administrator group **mervasys**
- MERVA user group, for example, **umerva1**

End User Authority

The USER authority is the lowest level of administrative authority.

With USER authority you can:

- Log on to MERVA USE & Branch
- Use MERVA functions depending on your access rights
- Access and process MERVA messages depending on your access right

You have USER authority only if you are a member of all of the following groups:

- Users group
- MERVA program group **mervalpp**
- MERVA user group, for example, **umerva1**

Multiple User Support

MERVA USE & Branch can be used by more than one user at the same time. With the MERVA Message Processing Client for Windows NT, also message-processing functions, such as create, edit, retype, and authorize can be used by several users at the same time. Other multiple user support can be achieved by using the Windows NT Server 4.0, Terminal Server Edition.

Hardware and Software Requirements

The following tables list the hardware and software requirements that are necessary to install and run MERVA USE & Branch.

Hardware Requirements

Table 1. Hardware Requirements for MERVA USE & Branch

Processor	Any system that runs under Windows NT Version 4.0 or a subsequent release. An Intel x86 compatible processor with 166 MHz. A processor with 333 MHz or higher is recommended.
RAM	A minimum of 96 MB. 192 MB are recommended.
Disk space	A minimum of 1 GB on a NTFS formatted drive is required. 4 GB are recommended.
CD-ROM drive	To install software.
Diskette drive	To install software keys.
Display	Any graphical display with a screen resolution of 1024 x 768 pixels or higher.
Printer (optional)	Any printer supported by Windows NT.

Table 1. Hardware Requirements for MERVA USE & Branch (continued)

Communications equipment	<p>For the SWIFT network connection via X.25:</p> <ul style="list-style-type: none"> • ARTIC X.25 Interface CoProcessor Adapter (PCI or ISA bus) <ul style="list-style-type: none"> – Public Switched Telephone Network (PSTN) connections to SWIFT network require a cable and a modem that supports V.25 dialing. This modem allows MERVA USE & Branch to use controlled automatic dialing and ensures most integrated customization and processing. – Device driver: IBM ARTIC Support for Windows NT – X.25 software: IBM ARTIC Support for X.25 on Windows NT • Encryptor box SecureX.25 provided by SWIFT You can use this box between the computer and the modem. It is optional for PSTN connections but mandatory for PSPDN connections. <p>For the MERVA-to-MERVA connection:</p> <p>MERVA Link, the MERVA Client, and the connection features support the communication between MERVA systems within an SNA network that uses Logical Unit (LU) 6.2 services or TCP/IP network. One of the following network control facilities is required:</p> <ul style="list-style-type: none"> • IBM 37xx communication controller • ATM card or SDLC card
Multiple user support	<p>If you use Windows NT Server 4.0, Terminal Server Edition, 20 MB of RAM for each additional user are required.</p>

Software Requirements

Table 2. Software Requirements for MERVA USE & Branch

Operating system	<p>MERVA V4.1 is based on Microsoft Windows NT 4.0, Service Pack Level 4, US English. It also runs under subsequent releases or modification levels of Windows NT unless otherwise indicated.</p>
License Use Management	<p>License Use Runtime Version 4.5.1 using the software key capability.</p>
Database system	<p>DB2[®] Universal Database Personal Edition Version 5.2, Service Pack Level 9094, or Version 6.1, unless otherwise indicated. US English is recommended.</p>
Communication system	<p>If SNA is used, one of the following is required:</p> <ul style="list-style-type: none"> • IBM eNetwork Personal Communication Version 4.2, or a subsequent release (5639-B94) • IBM eNetwork Communications Server for Windows NT 6.0 (5639-F25) <p>TCP/IP must be part of the Microsoft Windows NT 4.0 product.</p>
Additional program requirements for MERVA USE & Branch	<ul style="list-style-type: none"> • Object Rexx Runtime Edition Version 1.0.2.3 • Java Runtime Environment (JRE) Version 1.2 • IPFC Runtime that is included in the installation of the DB2 online help.
SWIFT Link	<p>IBM ARTIC Support for X.25 on Windows NT.</p>
API	<p>One of the following compilers is required:</p> <ul style="list-style-type: none"> • VisualAge[®] for C++ for Windows Compiler Version 3.5.4 • Microsoft Visual C++ 6.0

Table 2. Software Requirements for MERVA USE & Branch (continued)

Multiple user support	Windows NT Server 4.0, Terminal Server Edition Version 4.0
-----------------------	--

Chapter 2. Installing MERVA USE & Branch

This chapter describes the installation and configuration procedure for MERVA USE & Branch. It also describes the procedure for deinstallation.

Before you begin to install MERVA USE & Branch, ensure that you have the correct hardware and software prerequisites as listed in “Hardware Requirements” on page 5 and “Software Requirements” on page 6. You should also read the README.1ST file in the **Root** folder and the README.TXT file in the **USE_And_Branch** folder of the CD ROM.

You must install and configure MERVA USE & Branch in the following order:

1. Install the prerequisite software that is not yet installed.
2. Install the program files.
During installation of the program files, check the README file for additional information.
3. Create a MERVA instance.
4. Create the MERVA databases.
5. Install the MERVA license key.
6. Add initial MERVA users.
7. Verify the installation.
8. Setup the default customization for MERVA USE & Branch for Windows NT.
9. Verify that the correct time zone is set in Windows NT (this is required by USE).

Notes:

1. You can pause after each step and continue with the installation later. If a system or program abend occurs during installation, repeat the current step.
2. The installation program uses the environment variable **TEMP** or **TMP** to get a valid temporary directory.
3. Ensure that no other processes are running during the installation. This prevents the routine that checks the disk space from delivering incorrect results.

The following sections describe the installation and customization steps in detail.

Installing the Prerequisite Software

Ensure that you install the prerequisite software in the following order:

1. Microsoft Windows NT 4.0, Service Pack Level 4

Note: You must define a TCP/IP network in your Windows NT installation. If you do not have a network adapter installed, you must add and configure the MS Loopback Network Adapter.

2. Object Rexx Runtime Edition Version 1.0.2.3 (included on the MERVA USE & Branch installation CD)
3. DB2 Universal Database®

Note: If DB2 is already installed, ensure that all Object Rexx entries in the system environment variable **PATH** come before all SQLLIB entries of DB2.

4. Java Runtime Environment (JRE) Version 1.2 (included on the MERVA USE & Branch installation CD)
5. License Use Runtime Version 4.5.1 (included on the MERVA USE & Branch installation CD)

After successful installation of the License Use Runtime, start the License Key Server as a Windows NT service. To do this:

1. Click **Start → Programs → License Use Runtime → Configuration Tool**
2. Click **Configure As**.
3. Select **Nodelocked License Server and Advanced Configuration**.
4. Ensure that all other check boxes are cleared.
5. Click **Start up**.
6. Select **Start service at system startup**.
7. Close the window.
8. Click **Yes** to save the changes.
9. Ignore the information message displayed to you.
10. Restart your system.

Installing the Program Files

Note that you need Windows NT administration authorization for this task.

To install MERVA USE & Branch:

1. Log on with the user ID that has Windows NT administration authorization.
2. Insert the CD labeled MERVA ESA Components in the CD ROM drive.
3. Select your CD ROM drive.
4. Select the **MERVA_Features** folder.
5. Select the **USE_And_Branch** folder.
6. Start the setup program. To do this, double-click the **setup.exe** file.

First, the installation procedure checks whether the MERVA Message Processing Client for Windows NT is already installed. If it is not installed, you get the installation procedure for the MERVA Message Processing Client for Windows NT. After successful installation of the MERVA Message Processing Client for Windows NT, you get the Welcome window of MERVA USE & Branch. The following figure shows an example of this window.

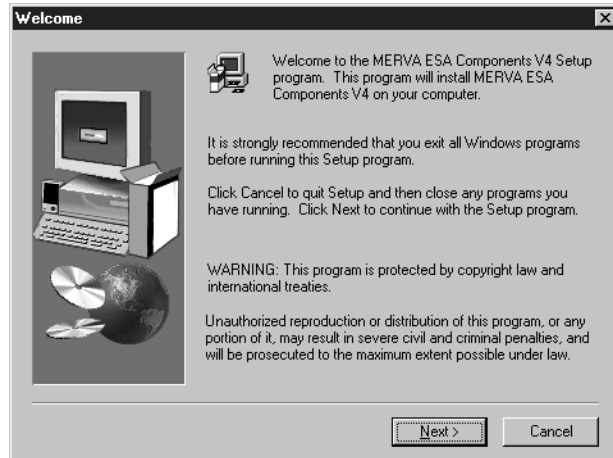


Figure 1. Welcome Window of MERVA USE & Branch

7. Click **Next** or press Enter to continue.

The installation procedure checks whether MERVA USE & Branch is already installed. If it is already installed, you get an information message, and the installation procedure is stopped. If it is not installed, the installation procedure continues, and the Software License Agreement is displayed.

8. Accept the Software License Agreement.
9. You then get the README file. Read the displayed information carefully, then click **Next**.
10. You then get the User Information window. The following figure shows an example of this window.

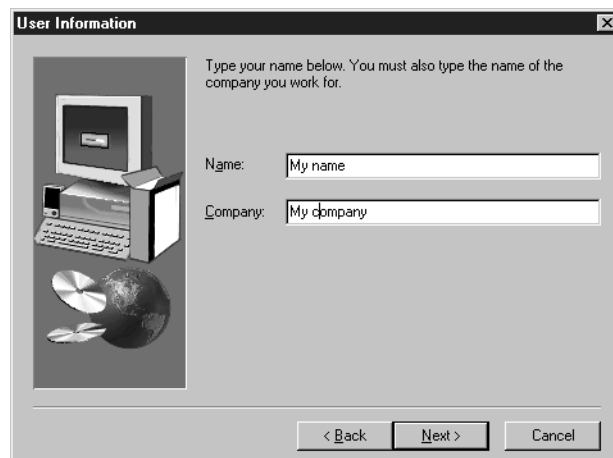


Figure 2. User Information Window

11. Type your name and the name of the company you work for.
12. Click **Next**.
13. You then get the Choose Destination Location window. The following figure shows an example of this window.

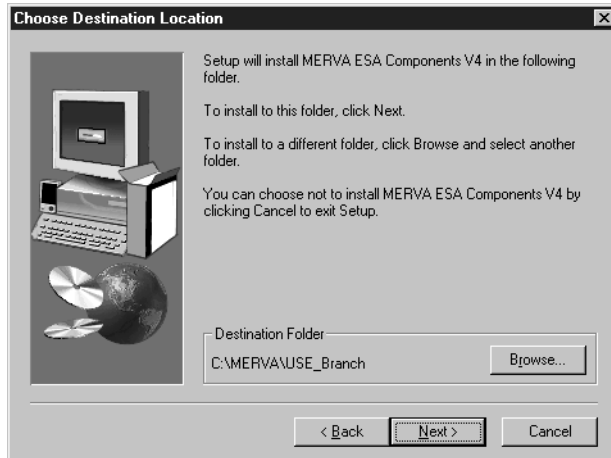


Figure 3. Choose Destination Location Window

14. Click **Next** to accept the destination folder C:\MERVA\USE_Branch or click **Browse** to select a different folder.

Notes:

- a. MERVA USE & Branch must be installed in an NTFS file system.
 - b. If the destination folder exists, it must be empty or include only the subdirectory **instances**.
 - c. The name of the destination folder must not include spaces.
15. You then get the Select Program Folder window. Select a program folder to which the program icons are to be added.
 16. The installation procedure checks the hardware and software requirements. You get the Result of MERVA Requirements Checking window. The following figure shows an example of this window.

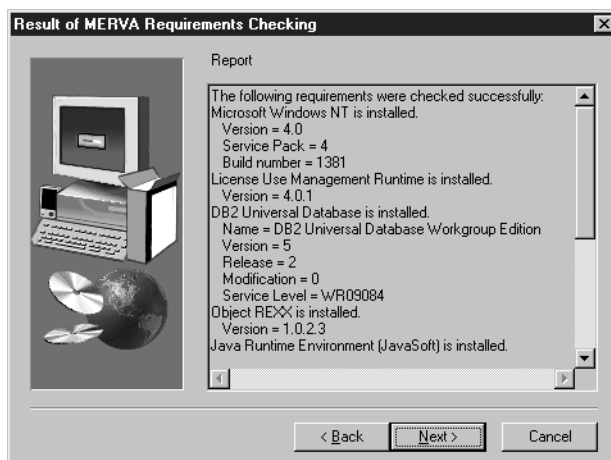


Figure 4. Result of MERVA Requirements Checking Window

17. Check the window contents carefully.
18. Click **Next**.
 - If all requirements are met, or if you get only a warning message, the installation proceeds.
 - If you get an error message, the installation abends.

19. You then get the Start Copying Files window. The following figure shows an example of this window.

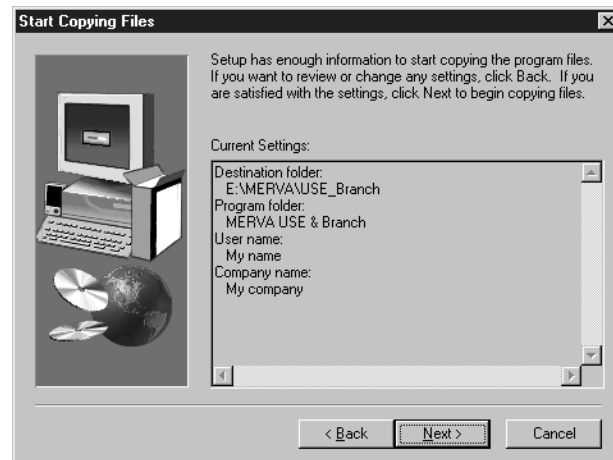


Figure 5. Start Copying Files Window

20. Check the window contents carefully.
21. If the displayed information is correct, click **Next**.
22. The installation procedure performs these actions:
- Decompresses the product files.
 - Copies the product files from the CD ROM to the destination folder.
 - Updates the Windows NT registry by adding information about MERVA USE & Branch.
You can find this data in **SOFTWARE\IBM\MERVA ESA Components\MERVA USE_Branch** under the root key **HKEY_LOCAL_MACHINE**.
 - Updates the system environment variables **BOOKSHELF**, **HELP**, **LOCPATH**, and **PATH**.
 - Creates the system environment variable **ENM_PATH** that contains the MERVA USE & Branch installation directory.
 - Installs the MERVA Inetd service.
 - Adds the MERVA program folder to **Programs** of the **Start** menu.
 - Creates the MERVA user groups **mervasys** and **mervalpp**.
 - Updates the access control list of all product files.
23. After successful completion of these actions, you are asked if you want to view the installation history. You can also view this data later. The installation history file **ENMWINST.LOG** is created in the temporary directory of the system. Usually, this is **C:\TEMP**. After successful installation, this file is stored in the subdirectory **install** of the MERVA USE & Branch installation directory.
24. The Setup Complete window is displayed. An information message tells you that the installation completed successfully. You are also asked to restart the system. This is recommended.
25. Click **Yes** to restart your system.

Installing Service

If service is available when you install MERVA USE & Branch for Windows NT, perform the installation in the following order:

1. Install the program files.
2. Install the service.
3. Create a MERVA instance.

Creating a MERVA Instance

Note that you need Windows NT administration authorization for this task.

To create a MERVA instance:

1. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
2. Select **Local System**.
3. Right-click **Local System**.
4. Click **Create instance...**
5. You then get the MERVA Instance Settings window. The following figure shows an example of this window.

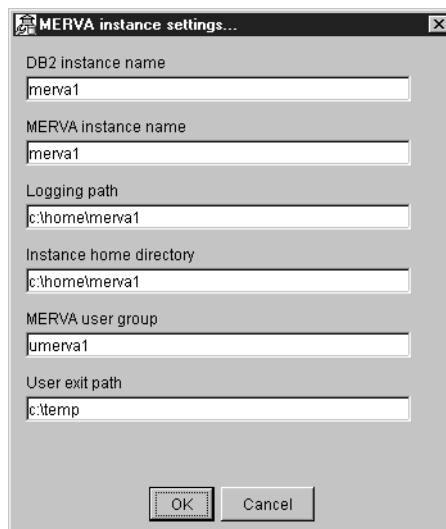


Figure 6. MERVA Instance Settings Window

6. In the field **DB2 instance name**, type the name of the DB2 instance that is to be created for the MERVA instance.
The sample name for the DB2 instance is **merva1**.
7. In the field **MERVA instance name**, type a name that is unique to all MERVA instances on all machines. This is especially important if you want to connect the MERVA instances to MERVA Link.
The sample name for this MERVA instance is **merva1**.
8. In the field **Logging path**, type the name of the directory in which all MERVA USE & Branch log and trace files for this instance are to be stored. You can type, for example, **C:\home\merva1**.
9. In the field **Instance home directory**, type the name of the directory in which MERVA USE & Branch output files are to be stored.
Output files can be:

- Automatic print file
- Message integrity control file for MERVA Link
- Message integrity control file for the connection feature

You can type, for example, **C:\home\merva1**.

10. In the field **MERVA user group**, type the name of the user group for which the MERVA instance is to be created, for example, **umerva1**.
11. In the field **MERVA user exit path**, type the name of the directory that should contain all user exits defined for MERVA USE & Branch, such as MERVA Link user exits.
12. Click **OK**.
The MERVA USE & Branch installation procedure then performs these actions:
 - Sets the system variable **ENMD_IPC_DIR** including the MERVA instance name.
 - Establishes MERVA USE & Branch as a Windows NT service with the descriptive name **MERVA-<MERVA Instance name>**, for example, **MERVA-merva1**.
 - Creates the MERVA user group.
 - Creates the control process configuration file. This file is located in the subdirectory **Instances**.
 - Creates the logging subdirectory structure and adds the MERVA user group to the access control list of the logging subdirectories.
13. Restart your system.

Creating the MERVA Databases

To create the MERVA databases, log on with a user ID that has Windows NT administration authorization. Note that the user ID must not exceed 8 characters.

To install the MERVA databases:

1. Click **Start → Programs → MERVA USE & Branch → Control Center**.
2. Double-click **Local System**.
3. Right-click the MERVA instance, for example, **merva1**.
4. Click **Databases → Install all**.
5. The DB2 instance is then created. All MERVA databases are created and filled with initial data.

MERVA uses the following databases:

Name	DB2 Identifier	Content
Control Database	ENMCNTRL	Customization data, users data, correspondents data, SWIFT USE data
Message Database	ENMQMSGC	Messages in MERVA queues
Logging Database	ENMLOGDB	Audit information

If an error occurs during installation or binding of the database, the error is displayed in the control center.

To install the MERVA databases on a user-defined drive:

1. Specify the drive with the user environment variable **DB2INSTPROF**, for example, **DB2INSTPROF=E:**.

2. Follow the instructions to install the MERVA databases on page 15.

Installing the MERVA License Keys

This section describes how to install the license keys for the following programs:

- MERVA USE & Branch for Windows NT
- MERVA USE & Branch for Windows NT with SWIFT Link
- MERVA Message Processing Client for Windows NT

MERVA USE & Branch for Windows NT

When you install MERVA USE & Branch for Windows NT, you must specify the license type that you want to use.

Installing the Regular License Key

To install this license key:

1. Insert the diskette labeled **MERVA USE & Branch License Key**.
2. Click **Start** → **Programs** → **License Use Runtime** → **Basic License Tool**.
3. Click **Products** → **Enroll Product**.
4. You then get the Enroll Product window.

The following figure shows an example of this window.

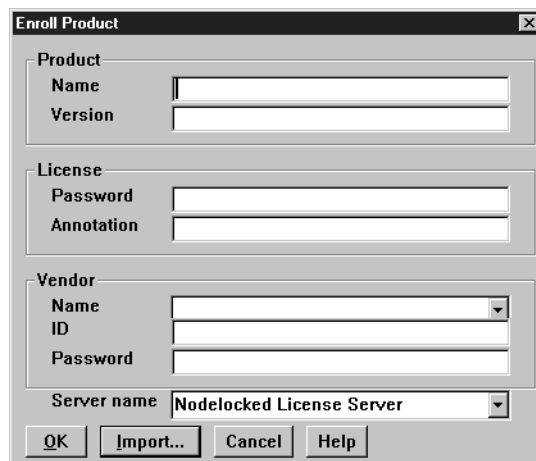


Figure 7. Enroll Product Window

5. Click **Import...**
6. Select the file **MERVA_USE_Branch.lic** from the license key list.
7. Click **OK**.
8. You then get the Enroll Product window again. It shows now the required license key information.
9. Click **OK**.

Installing the Try & Buy License Key

This Try & Buy license key is identical to the one of MERVA USE & Branch for Windows NT with SWIFT Link.

To install this license key:

1. Insert the CD labeled **MERVA ESA Components V4**.
2. Click **Start** → **Programs** → **License Use Runtime** → **Basic License Tool**.

3. Click **Products** → **Enroll Product**.
4. You then get the Enroll Product window.
Figure 7 on page 16 shows an example of this window.
5. Click **Import...**
6. Verify your CD-ROM drive letter. If the drive letter is **G**, select **G:\MERVA_Features\Evaluation_Keys\MERVA_SWIFT_Try_Buy.lic**. If the drive letter is not **G**, enter the corresponding drive letter.
7. Click **OK**.

MERVA USE & Branch for Windows NT with SWIFT Link

When you install MERVA USE & Branch for Windows NT with SWIFT Link, you must specify the license type that you want to use.

Installing the Regular License Key

To install this license key:

1. Insert the diskette labeled **MERVA USE & Branch License Key**.
2. Click **Start** → **Programs** → **License Use Runtime** → **Basic License Tool**.
3. Click **Products** → **Enroll Product**.
4. You then get the Enroll Product window.
Figure 7 on page 16 shows an example of this window.
5. Click **Import...**
6. Select the file **MERVA_SWIFT.lic** from the license key list.
7. Click **OK**.
8. You then get the Enroll Product window again. It shows now the required license key information.
9. Click **OK**.

Installing the Try & Buy License Key

To install this license key:

1. Insert the CD labeled **MERVA ESA Components V4**.
2. Click **Start** → **Programs** → **License Use Runtime** → **Basic License Tool**.
3. Click **Products** → **Enroll Product**.
4. You then get the Enroll Product window.
Figure 7 on page 16 shows an example of this window.
5. Click **Import...**
6. Verify your CD-ROM drive letter. If the drive letter is **G**, select **G:\MERVA_Features\Evaluation_Keys\MERVA_SWIFT_Try_Buy.lic**. If the drive letter is not **G**, enter the corresponding drive letter.
7. Click **OK**.

MERVA Message Processing Client for Windows NT

When you install the MERVA Message Processing Client for Windows NT, you must specify the license type that you want to use.

Installing the Regular License Key

To install this license key:

1. Insert the diskette labeled **MERVA Msg Processing License Key**.
2. Click **Start** → **Programs** → **License Use Runtime** → **Basic License Tool**.
3. Click **Products** → **Enroll Product**.

4. You then get the Enroll Product window.
Figure 7 on page 16 shows an example of this window.
5. Click **Import...**
6. Select the file **Client.lic** from the license key list.
7. Click **OK**.
8. You then get the Details Of window. It shows the required license key information.
9. In the field **Product Information**, enter the number of licenses to be enrolled. This number is identical to the number of clients ordered.
10. Click **OK**.

Installing the Try & Buy License Key

To install this license key:

1. Insert the CD labeled **MERVA ESA Components V4**.
2. Click **Start → Programs → License Use Runtime → Basic License Tool**.
3. Click **Products → Enroll Product**.
4. You then get the Enroll Product window.
Figure 7 on page 16 shows an example of this window.
5. Click **Import...**
6. Verify your CD-ROM drive letter. If the drive letter is **G**, select **G:\MERVA_Features\Evaluation_Keys\Client_Try_Buy.lic**. If the drive letter is not **G**, enter the corresponding drive letter.
7. Click **OK**.

Adding Users

This sections explains to you in a step-by-step description how to add initial MERVA users, administrators, and other users to Windows NT.

User groups and initial user IDs ensure access control to the MERVA instance.

Notes:

1. You need Windows NT administration authorization for this task.
2. **Important: All user names and passwords are case sensitive!**
3. User names and passwords must not exceed eight characters.
4. If MERVA is installed in a network with a Windows NT server as Primary Domain Controller (PDC), global groups and global domain users must not be assigned to the local MERVA groups.

Overview

The following table gives an overview of users and groups.

Description	User Name	Groups
Initial MERVA user	merva	Users mervalpp MERVA user group (umerva1)
MERVA system administrator	For example, mervaadm	Administrators mervasys MERVA user group (umerva1)
MERVA user administrator	For example, merva1	Users mervasys MERVA user group (umerva1)
Other MERVA users	Name	Users mervalpp MERVA user group (umerva1)

Adding an Initial MERVA User

To add a user:

1. Click **Start** → **Programs** → **Administrative Tools** → **User Manager**.
2. Click **User** → **New User**.
3. In the field **User name**, type **merva**.
4. In the field **Description**, type **Initial MERVA user**.
5. In the field **Password**, type any password. You have to change the password when you log on with the user ID **merva** for the first time.
6. Ensure that the user must change the password at the next logon.
7. Click **Groups**.
8. You then get the Group Memberships window.

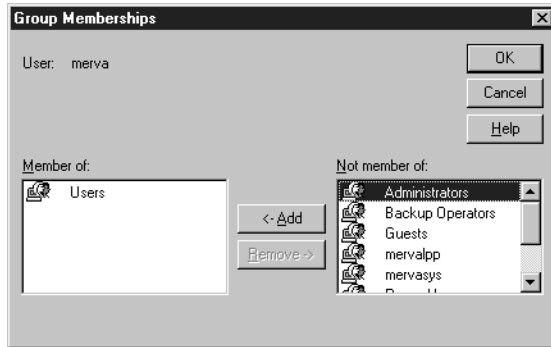


Figure 8. Group Memberships Window

9. In the field **Not member of**, select the group **mervalpp** and the MERVA user group, for example, **umerva1**. Note that the user must also be a member of the group **Users**.
10. Click **Add** → **OK**.
11. You then get the New User window.
12. In this window, click **Profile**.
13. In the field **Local path**, type a valid directory, for example, **C:\home\merva**. Note that this directory contains user-dependent configurations. Therefore, the name should be different for each user.
14. Click **OK**, then close the New User window.

Adding Administrators

Additionally, you must add the MERVA system administrator and the MERVA user administrator. The sample names are recommended names.

Adding the MERVA System Administrator

To add the MERVA system administrator:

1. Click **Start** → **Programs** → **Administrative Tools** → **User Manager**.
2. Click **User** → **New User**.
3. In the field **User name**, type a user name, for example, **mervaadm**.
4. In the field **Description**, type **MERVA system administrator**.
5. In the field **Password**, type a password.
6. Clear the field **User Must Change Password at Next Logon**.
7. Select the field **Password Never Expires**.

If you do not select this field, you cannot start MERVA USE & Branch after the password of the MERVA system administrator is expired. In this case, you have to change the password of the MERVA system administrator and the password in the Service window. For a description on how to change the password in the Service window, refer to “Configuring the MERVA Control Process” on page 22.

8. Click **Groups**.
9. You then get the Group Memberships window.
10. In the field **Not member of**, select the group **Administrators** and **mervasys**, and the MERVA user group, for example, **umerva1**.
11. Click **Add** → **OK**.
12. You then get the New User window.
13. In this window, click **Profile**.

14. In the field **Local path**, type a valid directory, for example, **C:\home\mervaadm**. Note that this directory contains user-dependent configurations. Therefore, the name should be different for each user.
15. Click **OK**, then close the New User window.

The MERVA system administrator must have the right to act as part of the operating system. To access this right:

1. Click **Start → Programs → Administrative Tools → User Manager**.
2. In the field **User name**, select the user name of the MERVA system administrator, for example, **mervaadm**.
3. Click **Policies**.
4. Click **User Rights...**
5. You then get the User Rights Policy window.
6. Select **Show Advanced User Rights**.
7. From the list **Right**, select **Act as part of operating system**.
8. Click **Add → Show Users**.
9. In the field **Names**, select the MERVA system administrator, for example, **mervaadm**.
10. Click **Add → OK**, then close the User Rights Policy window.

Adding the MERVA User Administrator

To add the MERVA user administrator:

1. Click **Start → Programs → Administrative Tools → User Manager**.
2. Click **User → New User**.
3. In the field **User name**, type a user name, for example **merva1**.
4. In the field **Description**, type **MERVA user administrator**.
5. In the field **Password**, type a password.
6. Clear the field **User Must Change Password at Next Logon**.
7. Click **Groups**.
8. You then get the Group Memberships window.
9. In the field **Not member of**, select the group **mervasys** and the MERVA user group, for example, **umerva1**.
Note that the user must also be a member of the group **Users**.
10. Click **Add → OK**.
11. You then get the New User window.
12. In this window, click **Profile**.
13. In the field **Local path**, type a valid directory, for example, **C:\home\merva1**.
Note that this directory contains user-dependent configurations. Therefore, the name should be different for each user.
14. Click **OK**, then close the New User window.

Adding Other Users

1. Click **Start → Programs → Administrative Tools → User Manager**.
2. Click **User → New User**.
3. In the field **User name**, type a user name.
4. In the field **Description**, type **MERVA user**.
5. In the field **Password**, type a password.
6. Click **Groups**.

7. You then get the Group Memberships window.
8. In the field **Not member of**, select the group **mervalpp** and the MERVA user group, for example, **umerva1**.
Note that the user must also be a member of the group **Users**.
9. Click **Add** → **OK**.
10. You then get the New User window.
11. In this window, click **Profile**.
12. In the field **Local path**, type a valid directory, for example, **C:\home**. Note that this directory contains user-dependent configurations. Therefore, the name should be different for each user.
13. Click **OK**, then close the New User window.

Configuring the MERVA Control Process

This section shows how to configure the MERVA Control Process.

1. Click **Start** → **Settings** → **Control Panel**.
2. Click **Services**.
3. Select the MERVA service, for example, **MERVA - merva1**.
4. Click **Startup...**
5. You then get the Service window.

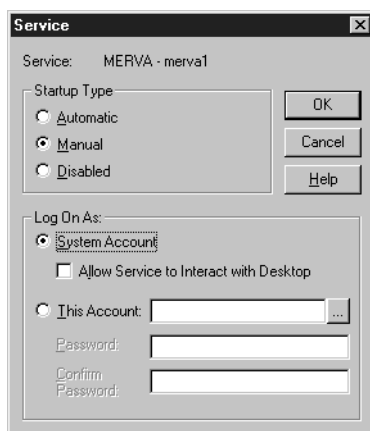


Figure 9. Service Window

6. In the field **Startup Type**, select **Automatic**.
7. From the list **Log On As**, select **This Account**.
8. In the field **This Account**, type the user ID of your MERVA system administrator, for example, **mervaadm**.
9. In the field **Password**, type the corresponding password.
10. Click **OK**, then close the Service window.
11. Restart your system.

Verifying the Installation

You can use the following procedures to ensure that MERVA works correctly.

Settings and Status of MERVA Instance

The first step is to verify settings and status of the MERVA instance. To do this:

1. Log on to Windows NT as the initial MERVA user with the user ID **merva**.
2. You are prompted to change your system password. Type a password other than **merva6k**. The password must not exceed eight characters.
3. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
4. Double-click **Local System**.
5. Select the MERVA instance, for example, **merva1**.
6. Right-click the selected MERVA instance.
7. Click **Show settings**.
8. Check whether the settings for the MERVA instance are displayed.
9. Click **Show status**.

The status of the MERVA instance should be **running**.

Setting Up a MERVA User Administrator

The next step is to create a MERVA user administrator. To do this:

1. Click **Start** → **Programs** → **MERVA USE & Branch** → **Main menu**.
2. The MERVA Logon window is displayed.
3. Type your current system password (Windows NT).
4. You get an error message that the password of your operating system does not match your MERVA password. The reason for this error message is that your system password is different from the initial MERVA password **merva6k**. Click **OK** to close the message box.
5. The MERVA Logon window is redisplayed.
6. Type your current MERVA password **merva6k**. The current system password is then saved as the new MERVA password. For more information on how to log on to a MERVA instance, refer to the *MERVA USE & Branch for Windows NT User's Guide*.
7. The MERVA Main Menu window is displayed.
8. Select **Administration**.
9. Double-click **Users**.
10. The User Administration window is displayed.
11. Update and approve the user **merva1** with all access rights.
For more information on user access rights, refer to the *MERVA USE & Branch for Windows NT User's Guide*.
12. Set an initial password for the user **merva1**. To do this, click **Selected** → **Set user password**.

The next step is to log on as MERVA user administrator. To do this:

1. Log on to Windows NT with the user ID **merva1**.
2. Click **Start** → **Programs** → **MERVA USE & Branch** → **Main menu**.
3. You then get the MERVA Logon window.
4. Type your initial MERVA password.
5. You then get the MERVA Main Menu window.

Your installation is successful if you do not get an error after you perform these steps.

Setting Up the Default Customization for MERVA USE & Branch for Windows NT without SWIFT Link

After successful installation of the MERVA USE & Branch program files, the default customization for MERVA USE & Branch for Windows NT with SWIFT Link is established automatically.

To establish the default customization for MERVA USE & Branch for Windows NT without SWIFT Link:

1. Log on to Windows NT with the MERVA User Administrator ID, for example **merva1**.
2. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
3. Double-click **Local System**.
4. Select the MERVA instance, for example, **merva1**.
5. Right-click the selected MERVA instance.
6. Click **Start customization mode**.

This starts MERVA USE & Branch in customization mode and the message **MERVA successfully started in customization mode** is displayed

7. Click **Start** → **Programs** → **Command Prompt**.
8. Type the following command in the command line:
`rexx enmcxuab.rex <instance name>`

For example, type

```
rexx enmcxuab.rex merva1
```

9. Close the Command Prompt window.

Uninstalling MERVA USE & Branch

This section describes in detail how to uninstall MERVA. This procedure consists of:

- Removing the MERVA databases
- Removing the MERVA instance
- Uninstalling the MERVA program files

Note that you need Windows NT administration authorization for each task.

Removing the MERVA Databases

To remove all MERVA databases:

1. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
2. Double-click **Local System**.
3. Select the MERVA instance for which the databases are to be removed, for example, **merva1**.
4. Right-click the selected MERVA instance.
5. Click **MERVA Databases** → **Drop all**.
6. All MERVA databases are deleted.
7. The DB2 instance is deleted.

Removing a MERVA Instance

To remove a MERVA instance:

1. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
2. Double-click **Local System**.
3. Select the MERVA instance to be removed, for example, **merva1**.
4. Right-click the selected MERVA instance.
5. Click **Delete**.
6. The selected MERVA instance is deleted.
7. The MERVA service is deinstalled.
8. The MERVA configuration file is deleted.

Uninstalling the MERVA Program Files

To remove all MERVA program files:

1. Log on as the Windows NT Administrator
2. Click **Start** → **Programs** → **MERVA USE & Branch** → **Uninstall**.
3. Confirm that you want to remove MERVA.

The uninstallation procedure then removes the following objects:

- MERVA Inetd service
 - Information about MERVA from the Windows NT registry
 - MERVA program files
4. Restart your system.

Chapter 3. Running MERVA under Windows NT Server 4.0, Terminal Server Edition

This chapter tells you what you should take into consideration when you run MERVA under Windows NT Server 4.0, Terminal Server Edition.

Installation of License Use Runtime

After the installation of License Use Runtime Version 4.5.1 is complete, call the command file **reglum.cmd**. The command file is located in the directory **<WINNT>\Application Compatibility Scripts\Install**. **<WINNT>** denotes the installation directory of Windows NT.

Note that the License Use Runtime does not run properly under Windows NT Server 4.0, Terminal Server Edition, if you do not call this command file.

Installation of Java Runtime Environment

After the installation of Java Runtime Environment Version 1.2 is complete, set the environment variable **JAVA_FONT=<WINNT>\FONT**. **<WINNT>** denotes the installation directory of Windows NT.

Configuring Terminal Server Clients

You can customize the client to show only a defined application instead of the entire desktop. This can help you, for example, to ensure security.

To run the MERVA Main Menu under the client software, use the following command:

```
<MERVA>\bin\enmccsta.exe enmcjgme
```

where **<MERVA>** denotes the installation directory of MERVA USE & Branch.

Part 2. Customizing MERVA USE & Branch

Chapter 4. Starting MERVA USE & Branch Customization

Customization Procedure

To start MERVA USE & Branch customization:

1. Log on to Windows NT with the user ID of the MERVA user administrator, for example, **merva1**.
2. Click **Start → Programs → MERVA USE & Branch → Control Center**.

You then get the MERVA Control Center window. The following figure shows an example of this window.

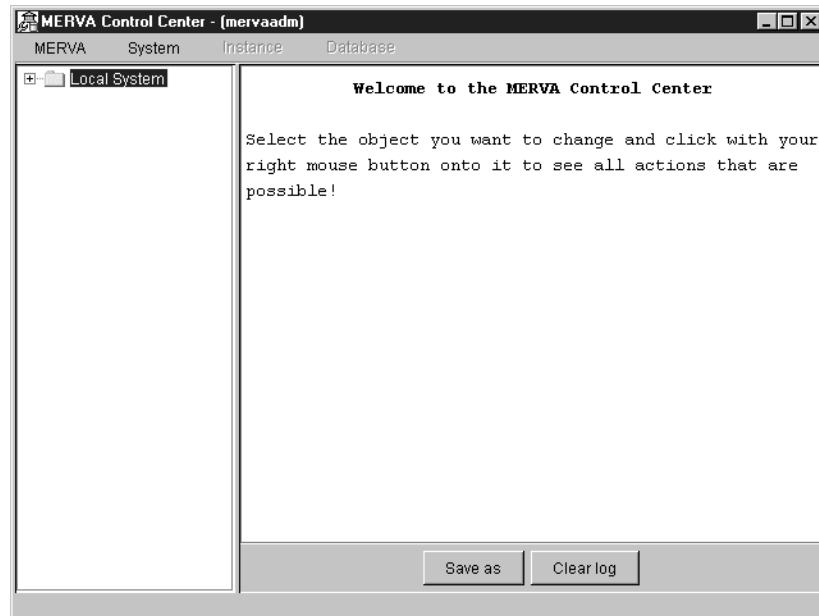


Figure 10. The MERVA Control Center Window

3. Double-click **Local System**.
4. Select the MERVA instance that you want to customize.
5. Right-click the selected MERVA instance.
6. Click **Start customization mode**.
This starts MERVA in customization mode and the message **MERVA successfully started in customization mode** is displayed
7. Log on to Windows NT with a user ID that has Customization access right, for example, **merva1**.
8. Click **Start → Programs → MERVA USE & Branch → Main Menu**.
You then get the MERVA Logon window.
9. Enter your Windows NT password and click **OK**.
You then get the MERVA Main Menu window.
10. Start customization in one of the following ways:
 - In the MERVA Main Menu window, select **Administration and Customization**. Then select **Start** from the **Selected** menu.

- Double-click **Customization** in the **Program** list of the MERVA Main Menu window.

You then get the MERVA Customizer window as shown in Figure 11. In this window, you can change your customization data.

Note: The Customization program can run in two different modes:

Customization Mode

The Customization mode is required to change MERVA according to your requirements. During this mode you must not use any other MERVA functions.

To start the MERVA background processes for this mode, select **Start customization mode**.

Display Mode

With this mode, you can display customization settings without changing them. The Customization program is used in parallel to other MERVA functions.

To start the MERVA background processes for all functions, click **Start multi-user mode**. You can then use the Customization program in Display Mode.

The MERVA Customizer Window

The Customizer window gives you access to the customization data. The data is displayed in a two-level selection list.

The following figure shows an example of the Customizer window.

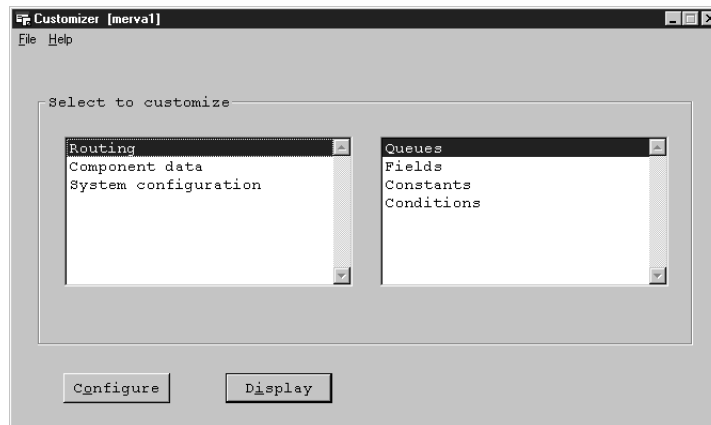


Figure 11. The Customizer Window

To change or display data:

1. Select a customization area from the box on the left.
2. Select a subitem from the box on the right.
3. Click **Configure** to change the selected data or click **Display** to access the data in read-only mode.

Note that the selection list contains only the objects that you are authorized to access.

The following table shows all available customization objects:

Customization Area	Subitem
Routing	Queues (see page 36) Fields (see page 37) Constants (see page 39) Conditions (see page 40)
Component data	MERVA Link (see page 57) SWIFT Link (see page 47) SWIFT USE (see page 75) Automatic print (see page 81)
System configuration	Alarms (see page 83) Currency Codes (see page 84) Date Format (see page 85) Default Printer (see page 82) Destination Checking (see page 86) Message Print Separator (see page 82) Logging Level (see page 86)

Saving Changes and Stopping the Customization Mode

To save your changes and to stop customization:

1. Click **File** → **Save** to save the current changes.

Note that the changes made during a customization program session, such as creating a new routing condition or changing the SWIFT Link configuration, are not saved automatically in the MERVA databases. You must save your changes, before you exit the Customization program.

2. Close the MERVA Main Menu.
3. Click **Start** → **Programs** → **MERVA USE & Branch** → **Control Center**.
4. Double-click **Local System**.
5. Select the MERVA instance that you want to stop.
6. Right-click the selected MERVA instance.
7. Select **Stop**.

Generating Customization Reports

With the Customization program, you can create a report containing the customization settings. This helps you keep a record of the actual message routing table.

To create a report, click **File** → **Report**. You can print the report or write it to a file.

Note: To ensure data integrity, you can only create a report after you save the changes made during a session.

Chapter 5. Defining the Message Routing

MERVA is a message-processing system. A message consists of data in a predefined format and contains all information necessary for processing a single financial transaction. The format of a message depends on the type of network and the type of financial transaction, such as foreign exchange or funds transfer.

MERVA currently supports the following message applications:

- SWIFT (using the SWIFT II network)
- Telex (using the public telex network)
- User-defined (using the MERVA API)

Messages are stored in **queues**. A queue is a logical structure used to hold messages in the same state, for example, waiting for transmission to SWIFT or waiting to be printed.

Each queue belongs to a logical group known as a **purpose group**. The purpose group determines by which functions the queues are accessed. For example, the queue SL_IN belongs to the API purpose group. Messages in queues belonging to this purpose group are processed by the user application programs. For more information refer to the *MERVA USE & Branch for Windows NT Application Programming Guide*.

When a message has been processed by a function, it is routed to the next queue, which normally belongs to a different purpose group.

The next queue is determined by the **routing conditions** that are defined for the current queue. Routing conditions can include tests on the message contents and other message-related data.

Tests are performed on **message fields**. A message field is a named part of a message, that is defined during customization. For a routing decision, the content of a message field is compared with a predefined value using a range of comparison operators. For example, the priority field in SWIFT messages is specified as:

- Scan the message buffer for the SWIFT application header {2:
- Drop the next 16 characters
- Take the next one character

Constants can be used instead of the real values so that the routing conditions can be read more easily. For example, you can define constants for the priority code of SWIFT messages:

- URGENT="U"
- NORMAL="N"

Using these queue, field, and constant definitions, messages are routed from the queue SL_IN to:

- A SWIFT send queue for urgent FIN messages
- A different SWIFT send queue for normal FIN messages
- An error queue if the field contains any other character

A default routing table is supplied containing the message queues, fields, and constants used in the standard routing of messages.

“Appendix B. Standard Routing” on page 169 lists these default parameter names and routing conditions.

Print out a customization report before you begin. See “Generating Customization Reports” on page 33 for further information. This report helps you plan your changes and gives you a record of the current routing table.

To define or change the message routing, first define the required queues, fields, and constants, then define the routing conditions. Queue, field, and constant names must be unique within the system.

Setting up Message Queues

Select **Routing** and **Queues** on the Customizer window shown in Figure 11 on page 32 to display the Queues window shown in Figure 12. This window lists all purpose groups and the corresponding queues for the purpose group that is currently selected.

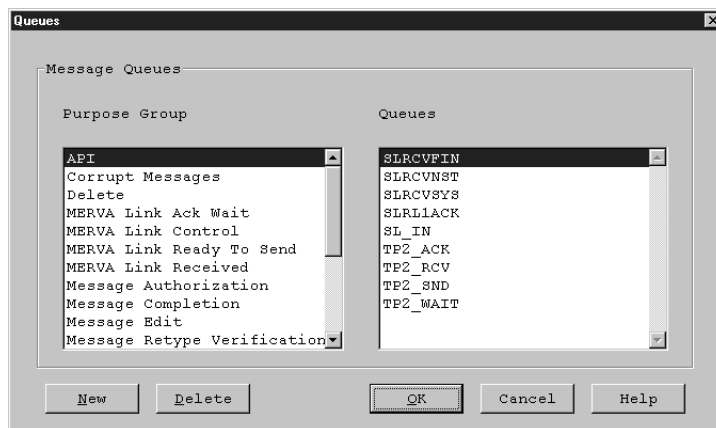


Figure 12. The Queues Window

Setting up a New Queue

To set up a new message queue name, select a **Purpose Group**, click on **New**, and type a name for the new queue. The available purpose groups are listed and explained in “Appendix B. Standard Routing” on page 169.

Deleting a Queue

To delete a selected queue, click on **Delete**.

A queue that is currently used as a target in a routing condition cannot be deleted. In this case, you have to change the **Routing Conditions** (see “Selecting Routing Conditions” on page 40) and remove all occurrences of the queue as a target queue from all routing conditions. All associated routing conditions for the selected queue as a source queue are automatically deleted.

Some message queues have user access rights assigned. If you delete one of these message queues, all user privileges associated with the queue are lost. Subsequently recreating a queue with the same name does not restore these

privileges. Use the User Administration program to restore the user privileges. Refer to the *MERVA USE & Branch for Windows NT User's Guide* for details.

Defining Message Fields

Select **Routing** and **Fields** on the Customizer window shown in Figure 11 on page 32 to display the Fields window shown in Figure 13. This window lists all message parts and the corresponding fields for the currently selected message part.

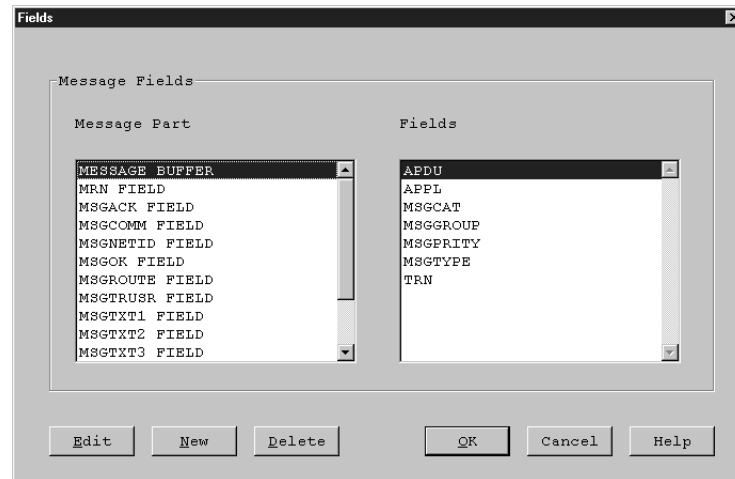


Figure 13. The Fields Window

A message is divided into a number of parts. A message field can be in any of the predefined message parts. The parts include the message text itself and related information, such as acknowledgments, comments, trace data, and routing data. The available parts are listed and explained in "Message Parts" on page 173.

Defining a New Field and Specifying Information for Fields

To add a new field, select a message part and click on **New** to display the New Field window shown in Figure 14 on page 38.

To define a field, specify where the field begins, how long it is, and what type of data it contains (numbers or text).

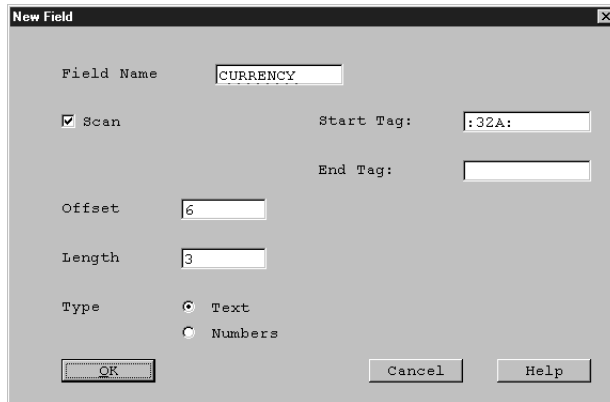


Figure 14. The New Field Window

Select **Scan** if the field does not have a fixed start position in the message part.

In this case you can define a **start tag** that is searched in the message. For example, the value date, the currency code, and the amount of a SWIFT message type 100 can be defined by searching for the tag ':32A:'.

If the field has a variable length, you can define an **end tag**. For example, you can define a comma as an end tag for an amount field without fractional digits.

In the **Offset** field, type the number of characters that are between the beginning of the message part for fixed fields or the start tag for scanned fields and the actual start of the field.

For example, the SWIFT 32A field has three subfields: Value date (6 digits), currency code (3 characters), and amount (up to 15 digits with a comma). The value date is defined with offset 0, the currency code with offset 6, and the amount with offset 9.

In the **Length** field, enter the length of the field in characters. For example, the value date has a length of 6, the currency code a length of 3, and the amount a length of 15.

An example of the use of the offset and length of a field when using scan patterns is shown in Figure 15. The field is defined as:

Start tag {3:
Offset 4
Length 2

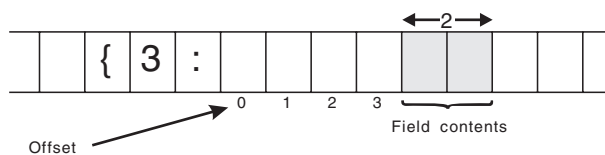


Figure 15. Example for the Use of Offset and Length Parameters

The **Type** of the field determines which type of “padding” the system uses when comparing fields, for example, when the field length is longer than the constant or text entered into it. For a numeric field, the value is padded with zeros from the left. For example:

"41" is defined to "00041"

For a text field, the value is padded with spaces from the right. For example:

"BANK" is defined to "BANK "

Changing Field Definitions

To change the definitions of a selected field, click on **Edit** or double-click on the field name.

Deleting a Field

To delete a selected field, click on **Delete**. You cannot delete fields that are used in any routing condition.

Defining Constants

Constant names are strings of characters that represent numeric values or other strings of characters. It is easier to read and maintain routing conditions if you define constants. For example, the constant names ACK_ACC and ACK_REJ are predefined as “0” and “1” respectively.

To define a constant, select **Routing** and **Constants** on the Customizer window shown in Figure 11 on page 32 to display the Constants window shown in Figure 16.

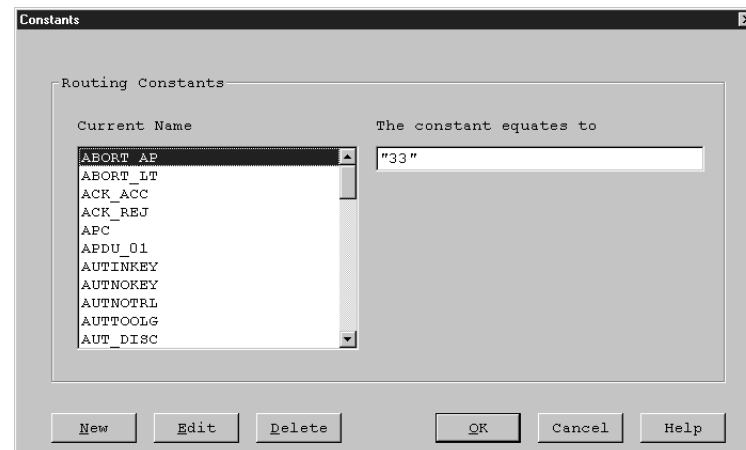


Figure 16. The Constants Window

You can do the following on this window:

- To add a constant, click on **New** and type the name and the value of the constant. The value of a constant must be enclosed in double quotes. The maximum length of a constant value is 30 characters.
- To change a selected constant, click on **Edit** and change the value.
- To delete a selected constant, click on **Delete**. You cannot delete constants that are used in any routing condition.

Setting up Routing Conditions

When you have defined all the message queue, field, and constant names that you need, you can set up routing conditions to route messages according to your requirements.

Routing conditions that define how messages are routed between queues are specified for source queues. Source queues contain the messages that are taken for further routing and determine the target queues for messages according to a series of logical tests. You can test for:

- Presence of a field within a message
- Presence of data within a message field
- Value of the contents of a message field

When a routing condition is satisfied, the message is sent to the target queue. Otherwise, the next routing condition is evaluated. If all defined routing conditions fail, or if no routing has been defined for the queue, the message is routed to a default queue to ensure that no messages are lost.

Note: You can define up to 4096 routing conditions. This number includes the routing to default queues. If you define, for example, 217 message queues, you automatically define 217 routing conditions. You can then add only 3879 new routing conditions because you have to deduct 217 from 4096.

Selecting Routing Conditions

Select **Routing** and **Conditions** on the Customizer window shown in Figure 11 on page 32 to display the Conditions window shown in Figure 17.

This window lists all purpose groups together with the corresponding queues for the selected purpose group.

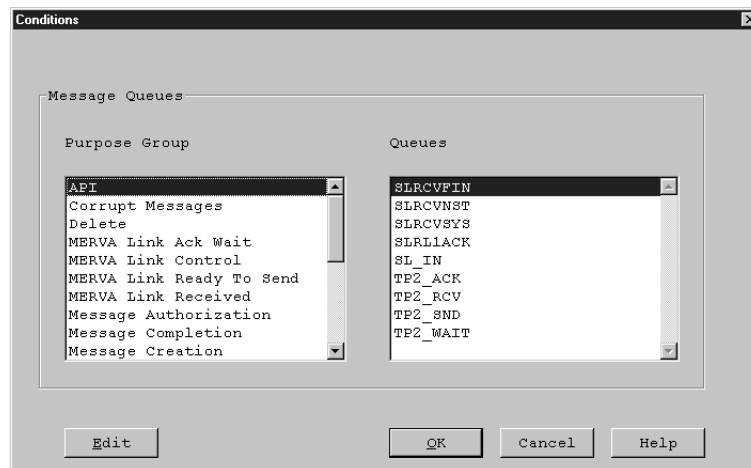


Figure 17. The Conditions Window

Each queue as a source of a routing step has at least one target queue. Newly defined queues have the target queue MBMERROR.

To edit the routing conditions defined for a selected source queue, select **Edit**.

Defining Routing Conditions

The Message Routing Definitions window shown in Figure 18 lists all routing conditions for the selected queue. For example:

SLRURG02 when MSGPRITY=URGENT

When the value of the MSGPRITY field in a message is equal to the value of the constant URGENT, the message is routed to the target queue SLRURG02. If the MSGPRITY field is not equal to URGENT, the next condition in the list is evaluated, until all routing conditions for this source queue have been tested. The text **...then stop** indicates that routing is terminated when this condition is true. Refer to “Creating Routing Conditions” for further details.

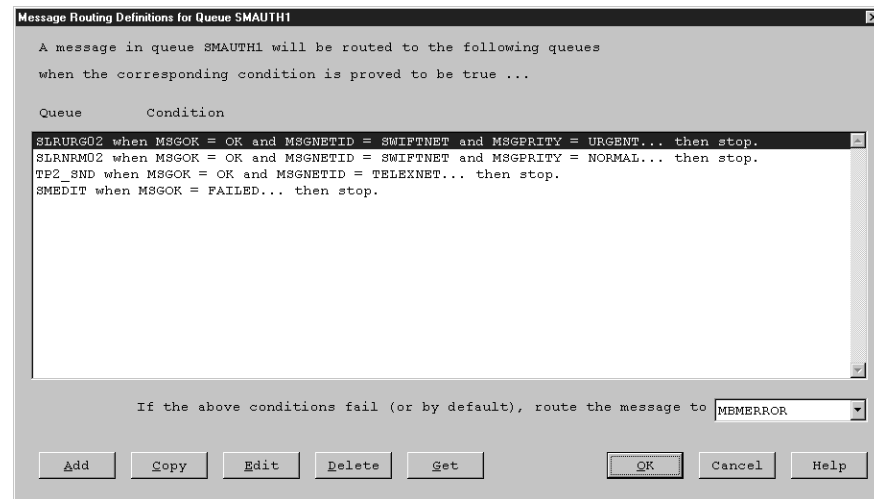


Figure 18. The Message Routing Definitions Window

The list box below the routing conditions shows the target queue to which messages are routed by default or if all routing conditions fail.

You can do the following on this window:

- To change the default target queue, click on the down arrow and select a different target queue.
- To create a new routing condition for the queue, click on **Add**. After completion, the new condition is inserted below the selected condition.
- You can also select a condition that has similar attributes as the one that you want to add. Click on **Copy** to create a copy of this condition. Then select **Edit** to modify this condition as required.
- You can also copy a routing condition from a different queue by clicking on **Get**. See “Copying Routing Conditions from Other Queues” on page 43 for details.
- To change an existing routing condition, select it and click on **Edit**.
- To delete a condition, select it and click on **Delete**.

Creating Routing Conditions

Select **Add** or **Edit** on the Message Routing Definitions window shown in Figure 18 to display the Message Routing Conditions window shown in Figure 19 on page 42.

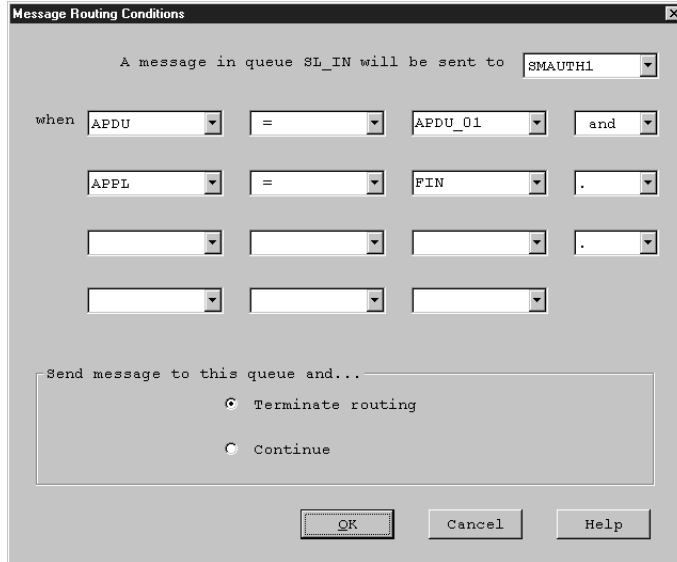


Figure 19. The Message Routing Conditions Window

The information shown in the Message Routing Conditions window describes a routing condition as a complete sentence. The routing condition shown in Figure 19 could therefore be read as:

A message in [source] queue SL_IN will be sent to [target] queue SMAUTH1 when the APDU [field] is equal to [the constant] APDU_01 and when the APPL [field] is equal to [the constant] FIN.

You can define routing conditions only by using the fields and constants that you have already defined.

A routing condition is composed of one or more groups of entry fields. Each group of entry fields must be in the following order:

FIELD OPERATOR OPERAND

Where:

FIELD A previously defined field name.

OPERATOR One of the following logical operators:

- = Is equal to
- > Is greater than
- < Is less than
- >= Is greater than or equal to
- <= Is less than or equal to
- <> Is not equal to

NOT FOUND

To check for the presence of a field within a message. For example:

A message in queue SL_IN will be sent to queue PRINTER1 when AMOUNT is NOT FOUND

sends the message to the verification queue PRINTER1 when the AMOUNT field cannot be found within the message.

This method can be used to check for missing scan patterns, or when a search beyond the end of a message part is attempted. Do not select an OPERAND in this case.

EMPTY

To check for the presence of data within the field. Do not select an OPERAND in this case.

OPERAND Must be either a predefined constant or a string enclosed in double quotes.

Click on the down arrow of an entry field for a list of possible field names, constants, or operators. Select a name or an operator to enter it into the field. You can also type a numeric or a string value, enclosed in double quotes ("), into the OPERAND entry field.

To remove part of a condition, select the **blank** entry.

When defining more complex conditions, that is, those with more than a single group of entry fields, end the first group by selecting the appropriate Boolean operator (AND or OR) from the combination box in the fourth column. Then define the next group of entry fields.

Notes:

1. To avoid ambiguity, you cannot mix AND and OR operators within a single routing condition.
2. You can define any number of target queues. However, messages are only sent to the first 12 target queues whose conditions are found to be true.

To create copies of messages, click on **Continue** at the bottom of the window. If the displayed condition is satisfied and a message is routed to the specified target queue, routing does not stop. The following conditions are evaluated too. This can result in more target queues. When a message is sent to more than one target queue in this way, copies of the message, each with identical Message Reference Number (MRN), are made.

By default messages are sent to this target queue only (**Terminate routing**), and routing terminates when the condition is true.

Copying Routing Conditions from Other Queues

Select **Get** on the Message Routing Definitions window shown in Figure 18 on page 41 to copy routing conditions defined for a different queue into the current queue definition. The Get Routing Definition window shown in Figure 20 on page 44 is displayed. It lists all queues for which routing conditions have been defined. Select the queue from which to copy routing conditions.

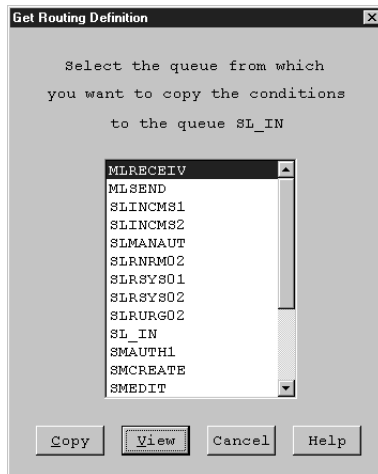


Figure 20. The Get Routing Definition Window

Click on **Copy** to copy all conditions from the selected queue to the current queue. The conditions are inserted below the currently selected routing condition.

Click on **View** to display the shown in Figure 21 to view the routing conditions defined for the selected queue.

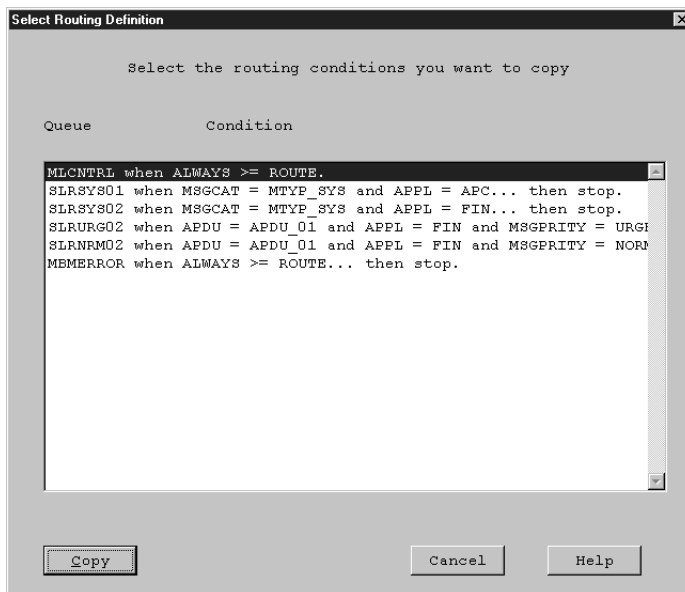


Figure 21. The Select Routing Definition Window

You can select one or more of the listed conditions and click on **Copy**.

Click on **Cancel** to return to the Get Routing Definition window shown in Figure 20 without copying any conditions.

Debugging Routing Conditions

MERVA provides a run-time routing trace facility. This routing trace can be activated with the Logging Level function from the MERVA Main Menu.

The *MERVA USE & Branch for Windows NT Diagnosis Guide* provides further information on using the routing trace facility.

Chapter 6. Customizing the SWIFT Link

Customizing the SWIFT Link comprises the following subtasks:

- Defining your Logical Terminals (LTs)
- Defining queues for the LTs
- Defining network data for the SWIFT Link
- Defining synonym LTs
- Defining the FIN Copy Service

To begin the SWIFT Link customization, select **Component data** and **SWIFT Link** from the list boxes of the Customizer window shown in Figure 11 on page 32 to display the SWIFT Link Data window shown in Figure 22.



Figure 22. The SWIFT Link Data Window

Defining Logical Terminal (LT) Identifiers

The LT uniquely identifies a terminal in the SWIFT network. It can be 9 characters long. Each physical terminal can have one or more LTs. The first LTs you must enter are the **Master LTs** supplied by SWIFT

The default Master LT supplied with MERVA USE & Branch is **IBMMERVAX**. To change it to your own Master LT, click on **Edit** and enter a new name. When changing an LT name, all queue assignments and synonyms are maintained.

Click on **New** to add more Master LTs.

Note that you do not have to change the Master LT for a *USE only* scenario. In this case, you can ignore the warning message **ENM6627A**. This message is shown if you use the sample LT **IBMMERVAX** when you save the changes in your customization.

Defining Synonym LTs

Select **Synonym LTs** from the **Selected** pull-down menu to define synonym LTs for a Master LT.

Synonym LTs are additional names that you can specify for a Master LT. These synonyms enable the system to treat a physical terminal as a logically separate and unique terminal. Similarly, the same LT can be given to more than one physical terminal. This enables the system to treat a group of terminals as a single logical entity.

You can define up to 16 LTs (masters and synonyms together). At least one Master LT must be defined.

Defining Queue Assignments for Logical Terminals

Select **Queue Assignments** on the **Selected** pull-down menu or double click on a Master LT to display the Queue Assignments for Logical Terminals window shown in Figure 23.

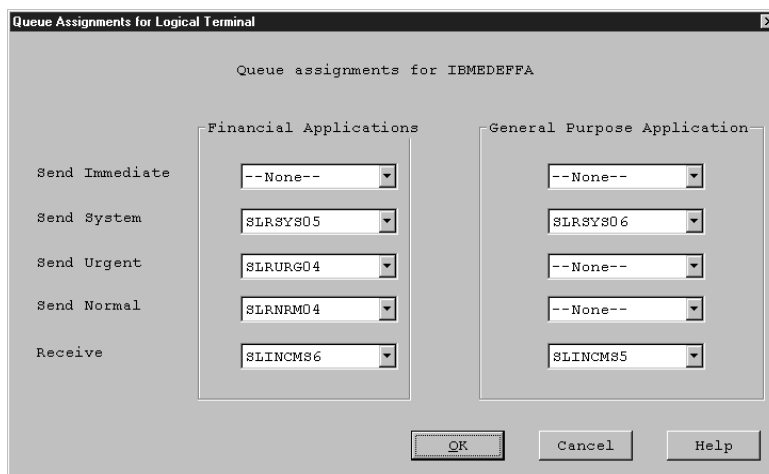


Figure 23. The Queue Assignments for Logical Terminal Window

Each Master LT must be associated with a set of message queues that are used to send and receive messages for this LT. Assign these queues as soon as you have created a Master LT.

Two sets of message queues must be associated with each Master LT, one for the financial application (FIN) and one for the general purpose application (GPA).

For sending messages, you can specify different queues for different priorities:

Send Immediate

Messages in this queue are sent immediately

Send System Used to send SWIFT system messages (priority S)

Send Urgent Used to send SWIFT urgent messages (priority U)

Send Normal Used to send SWIFT normal messages (priority N)

Receive Entry point for incoming messages

Messages are first taken from higher-priority send queues. When the higher-priority queue is empty, messages are taken from the next-highest priority queue.

Enter a different name for each queue. You can also choose **--None--** for an entry, if you do not intend to include the queue in your routing table. You must, however, define at least one send queue and one receive queue for each LT and each application that you define.

Note: For the queues that you define here, you must set up the appropriate routing conditions.

Defining the Network Data for SWIFT Link

Select **Network Data** from the **SWIFT Link** pull-down menu to display the Network Data window shown in Figure 24.

Use this window to configure the SWIFT Link connection to the SWIFT II network.

The screenshot shows the 'Network Data' configuration window. It is divided into several sections:

- Line Type:** Radio buttons for 'leased line', 'dedicated PSTN', 'shared PSTN', and 'PSPDN'. 'shared PSTN' is selected.
- Dial Type:** Radio buttons for 'manual' and 'automatic'. 'automatic' is selected.
- PDU size (bytes):** A dropdown menu showing '4096'.
- Line Details:** Fields for 'Symbolic address' (IEMADEFF), 'Local phone number' (497031287716), 'Local DTE', 'Remote DTE', 'Suspend desired' (No), and 'Link name' (ENMX25L).
- Time-outs (seconds):** Fields for 'Acknowledge' (900) and 'Association' (60).
- Resynchronization:** Fields for 'Number of retries' (3), 'Delay before start (seconds)' (40), and 'Maximum duration of recovery (minutes)' (10).

Buttons for 'OK', 'Cancel', and 'Help' are located at the bottom right.

Figure 24. The Network Data Window

Use the Network Data window to supply the following configuration data:

- Dial Type** Define whether to use automatic or manual dialing.
- PDU Size** You can change the default protocol data unit (PDU) size to be used by the Transport Layer when sending messages via the SWIFT network. The default value is 4096 bytes. Setting a larger data unit size can increase the line throughput, but also increases storage requirements.
- Line Details** The **Symbolic address** is the destination name for your system, delivered to you by SWIFT
The **Local phone number** is required for a shared PSTN.

The **Local DTE** address and the **Remote DTE** are required for a PSPDN network. Enter your own and the SWIFT address, respectively.

Specify whether you want to **suspend** the SWIFT line when the line is not busy.

The **Link name** is the reference to the X.25 link profile.

Time-outs Change these values only on request of SWIFT or IBM.

The acknowledgment timeout is the period of time allowed for the acknowledgment by the SWIFT network that a message has been received. The default value is 900 seconds.

The association timeout is the period of time allowed for establishing a session with the remote partner. The default value is 60 seconds.

Resynchronization

Specifies the:

- Maximum number of retry attempts. The default value is 3. If 0 is specified, the resynchronization is disabled.
- Delay (in seconds) before retry is started. The default value is 40.
- Maximum duration of recovery. When the specified time limit (in minutes) is exceeded, the retry attempts are stopped. The default value is 10.

Line Type Define the type of connection between your computer and the SWIFT transport network.

MERVA USE & Branch supports the following *connection types* between the computer at the bank and the SWIFT transport network:

- Normal phone line (PTT connection)

bank computer ---- PSTN connection ---- PSPDN: SWIFT network

The supported line types are:

- Leased line
- Shared PSTN (Public Switched Telephone Network)
- Dedicated PSTN
- Connection to a PSPDN (Packet Switched Public Data Network)

bank computer ---- PSPDN ---- PSPDN: SWIFT network
private or public

Multiple PSPDNs are supported between the bank computer and the SWIFT PSPDN.

The PSPDN can be a private PSPDN or a public PSPDN. These PSPDNs have gateways to the SWIFT PSPDN.

MERVA USE & Branch supports the following *line types*:

- Leased line (public PTT)

A leased line (sometimes called a non-switched line) is a permanent connection between the computer and the SWIFT network.

- Pseudo-leased line (encryptor box)

The line is a switched line, but it is defined for MERVA USE & Branch as a leased line. When MERVA USE & Branch starts a connection, it sends a call request to the box assuming that there is already a physical connection with the network. The encryptor box gets the call request and establishes a line connection via Data Terminal Ready (DTR) dialing. For details see the description that comes with the encryptor box.

- Shared PSTN - switched line (public PTT)

The connection has to be established by dialing the network computer. The dialing can be done manually or automatically through the modem. The following options are available for automatic dialing (V.25bis procedures):

- The computer sends a dial command with the phone number to the modem. This method is called *addressed*.
- The phone number is stored in the modem and the computer tells the modem via the DTR signal that the connection should be established. This method is called *direct*.

- Dedicated PSTN - switched line (public PTT)

A port at the SWIFT side is reserved for a customer. The line connection is established via automatic or manual dialing. The protocol is according to the definitions for a leased line. The line is never released or reconnected via suspension or resumption.

- PSPDN - Packet Switched Public or Private Data Network

There is a permanent connection between the CBT and the network in most PSPDNs. It can also be a switched line, but the protocol is identical for MERVA USE & Branch in both cases. Two addresses must be provided: the own address (local DTE address) and the SWIFT address (remote DTE address). The PSPDN provider delivers the local DTE address. The remote DTE address is provided by SWIFT

The suspension and resumption process is optional for PSPDN networks.

The following *communication equipment* is required:

- Modems:

- For the PSPDN connections, the modem is delivered by the PSPDN provider.
- For PSTN connections, the modem must support a full duplex bit synchronous communication. If autodialing (V.25bis commands) is used, the modem must be able to handle bit synchronous commands (ASCII character set, CCITT V.22 or V.32).

The maximum line speed is 9600 Baud (kbps). If there is a problem with line quality (too many line errors caused by line noise), reduce the line speed through the modem configuration.

- Encryptor box

For PSPDN access an encryptor box is mandatory. The encryptor box is delivered by SWIFT. The encryptor box is connected between the modem and the computer. For switched lines, only manual dialing and DTR dialing is possible with the encryptor box. The encryptor box cannot pass the V.25bis dial commands to the modem.

Defining the FIN Copy Service

The SWIFT FIN Copy Service is a communication service to facilitate the clearing, netting, and settlement of securities, payments, and other financial transactions.

There are different services available. The services are identified by a 3-character copy service identifier. This service identifier defines the functionality of the service.

If you define a service, you must exactly follow the instructions received from your Central Institution (CID). The CID defines the service and can change it. If you do not follow the instructions, your messages may be NAK'ed.

To define the service, select **FIN Copy** from the **SWIFT Link** pull-down menu to display the FIN Copy Service Definitions window shown in Figure 25.

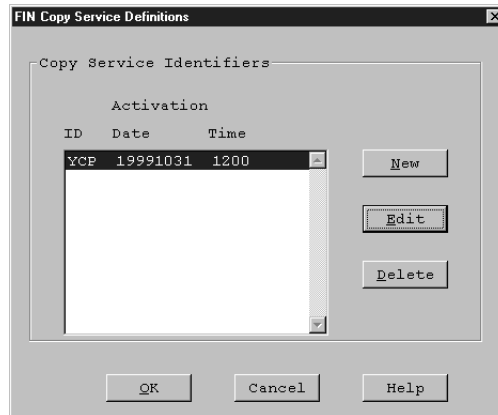


Figure 25. FIN Copy Service Definitions Window

The FIN Copy Service Definitions window shows currently defined services. You can do the following on this window, to:

- Delete an already defined service from the list, select the service from the list and click on **Delete**.
- Define a new service, click on **New**.
- Change an already defined service, click on **Edit**.

If you select **New** or **Edit**, the FIN Copy Service Message Definition window shown in Figure 26 on page 53 is displayed.

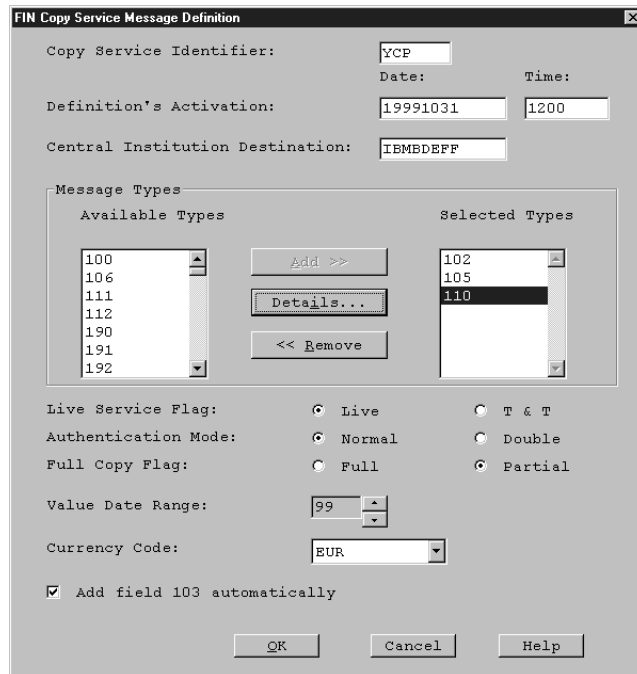


Figure 26. FIN Copy Service Message Definition Window

Use the FIN Copy Service Message Definition window to specify the service definitions.

Copy Service Identifier

Enter the 3-character service identifier.

Definition's Activation

You can specify the definitions in advance. You must specify when the service should be activated. The activation date has the format *YYYYMMDD* and the activation time has the format *HHMM*.

YYYY four-digit year

MM two-digit month

DD two-digit day

HH two-digit hour (from 00 to 23)

MM two-digit minutes (from 00 to 59)

This definition can also be used to deactivate (set to dormant state) a service. You can specify a date in the future and reset the date to activate it again. It is not necessary to delete and to add a definition for a service, if the service is not available for any reason.

Note: The service used depends on the date and time when SWIFT Link is started. This means that copy services are not exchanged automatically. Restart SWIFT Link to deactivate the old service and to activate the new service.

Central Institution Destination

The BIC code (8-character code) of the central institution that handles the copy service.

Note: If the authentication mode **double** is selected, and if you do not specify the **CID** field in the Message Fields window, the value of the **Central Institution Destination** field is used for the Proprietary Authentication Code (PAC) calculation. For an example of the Message Fields window, refer to Figure 27 on page 55.

Message Types

Only some message types are handled by the service. You can specify all message types that should be handled by the copy service. Select the message types from the list of **Available Types** and click on **Add** to add them to the list of **Selected Types**. To delete a message type, select it and click on **Remove**.

Live Service Flag

Specify the usage of the service for Live or Test and Training (T&T).

Authentication Mode

Specify the authentication mode:

- | | |
|---------------|--|
| Normal | Only the MAC trailer is added (as usual). |
| Double | A PAC trailer is added too. The PAC trailer is used for the authentication between the sender/receiver bank and the CID. |

Full Copy Flag

Specify full copy or partial copy. If you specify partial copy, you must make a list of the fields for each message type and send this list to the CID.

Value Date Range

Specify the value date range. It indicates if the copy service needs to make a value-date check on the message to be sent.

- | | |
|-------------|-----------------------|
| 0 | Same day |
| 1-98 | Range of days |
| 99 | No check will be done |

Currency Code

The copy of the messages can depend on a specific currency code. In this case the field tag that contains the currency code must be specified for all selected message types. See the Message Fields window shown in Figure 27 on page 55 for details.

Add field 103 automatically

All FIN Copy messages must contain a service identifier (field 103: Service Code) in the User Header. You can choose to let the MERVA USE & Branch system add this field for you, by selecting **Add field 103 automatically**. Before sending the message to SWIFT, MERVA USE & Branch compares the message content to the service definition and adds field 103 if all criteria are met. If the message already contains a field 103, it will be replaced. If you do not select this check box, MERVA USE & Branch will neither add nor replace the field.

Notes:

1. MERVA USE & Branch does not remove a service identifier that was added by the user.
2. If you do not specify the **CID** field in the Message Fields window, field 103 is not filled automatically. In this case, the message has to provide this field.

To define the fields for a selected message type, select the message type in the **Selected Types** list, and click on **Details** to display the Message Fields window shown in Figure 27.

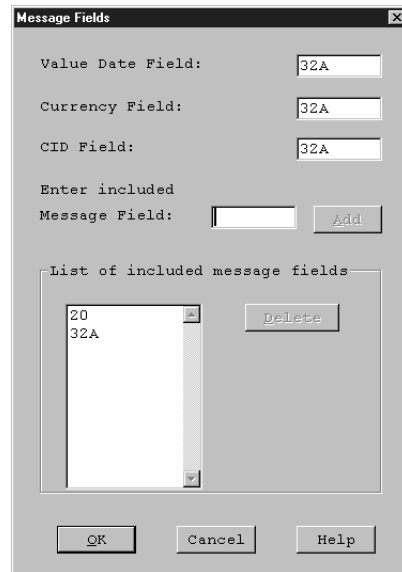


Figure 27. Message Fields Window

The Message Fields window allows you to specify the fields (field tags) for the value date, the currency code, the CID, and those message fields of the selected message type that should be copied and sent to the CID. It depends on the FIN copy definitions whether the **CID** field is required. If you select double authentication in the FIN Copy Service Message Definition window, and if you do not specify the **CID** field, the value of the **Central Institution Destination** field is used for the PAC calculation. For an example of the FIN Copy Service Message Definition window, refer to Figure 26 on page 53.

Enter a field and click on **Add** to add the field to the list. To delete a field from the current list, select it and click on **Delete**.

Hints to Set up Routing Related to FIN Copy Service

The routing field **MSGACK** has a length of 127 characters. It is used to keep the diagnostic information written by the SWIFT Link during send and receive of SWIFT messages. This diagnostic information can be used for routing.

The **MSGACK** field can contain one of the following information:

- SWIFT ACK or NAK
- MERVA error or information message

The **MSGACK** field contains:

- For input messages:
 - The ACK received from SWIFT if the authentication is correct or not necessary
 - The MERVA error message describing the authentication error if the message authentication fails
 - PAC information if a MAC and a PAC must be calculated, and the MAC calculation is correct but the PAC calculation fails

- For output messages:
 - The MERVA error message **ENN9128** if the authentication fails for the MAC
 - PAC information if the MAC information is correct but the PAC calculation fails

Additionally, the information about authentication is stored in the fields **MSGMAC** (MAC information) and **MSGPAC** (PAC information). You can use these fields for a more detailed routing.

For example, to route a SWIFT message based on the PAC information, you can use the field **MSGPAC**. This field can contain a MERVA message with one of the following message identifiers:

- ENM9985
- ENM9986
- ENN9128
- ENN9129
- ENN9130
- ENN9131
- ENN9132
- ENN9133
- ENN9134

Chapter 7. Customizing the MERVA Link

Introduction to MERVA Link

The MERVA Link component is the means to communicate between several MERVA installations using System Network Architecture (SNA) connections (applicable for all MERVA products), or between MERVA installations using TCP/IP connections. MERVA Link can be used either when routing a message from one MERVA system to another, or to access a SWIFT Link component residing on a remote MERVA system.

When sending and receiving messages, MERVA Link ensures that any loss of a message is detected and reported. It also ensures that no message is stored twice in the queues of the receiving MERVA system.

Under normal circumstances, MERVA Link works automatically without operator intervention. Control facilities are provided in every MERVA product to supervise message processing or to restart processing after failures.

MERVA Link Structure and Resources

The following sections provide a short introduction into the basic MERVA Link structures, and introduce the terminology used by MERVA Link. For detailed information about the MERVA Link architecture refer to *MERVA ESA V4 Advanced MERVA Link*.

Physical Processes

MERVA Link provides the means to communicate between two MERVA systems. In a specific instance of that communication one of the two MERVA systems is the sending system (also called the originator system or the client system). The other MERVA system is the receiving system (also called the recipient system or the server system). This communication is performed by a process in the sending system (a MERVA Link sending process) and a process in the receiving system (a MERVA Link receiving process). Both processes run at the same time.

A MERVA system can be the client at one time, and assume the role of a server at another time. A MERVA system can even be the client and the server at the same time (using parallel connections).

Logical Processes

Unique names are assigned to MERVA Link processes within the network of interconnected MERVA systems. A fully qualified MERVA Link process name consists of the MERVA Link node name and the Application Support Process (ASP) name. ASP names are unique within a MERVA installation. The MERVA Link node name applies to one MERVA installation. Any two MERVA installations in a network of interconnected MERVA systems must have different MERVA Link node names.

A MERVA Link ASP that has a unique name in the MERVA network (node_name.asp_name) is a logical MERVA Link resource. Parameters such as the name of the partner ASP, the send queue names, and the receive queue name are associated with an ASP when it is defined in the MERVA Link customization database.

MERVA Link Parameters

The MERVA Link parameters can be divided into subsets that apply to the following:

- Local MERVA Link system
- ASP (sending and receiving)
- Sending ASP
- Receiving ASP
- Connection to a partner MERVA Link system

Local System Parameters: The Local System Parameters describe the local MERVA Link system. The local MERVA Link node name is the most important element of this parameter subset. It is the node name of all ASPs defined in the MERVA Link component of the MERVA instance. If there is another MERVA instance installed on the same host, you must assign a different node name to the MERVA Link component of the MERVA instance.

ASP Parameters: An ASP represents the logical end of a sending or receiving MERVA Link connection in a MERVA Link node. A MERVA Link node can host many ASPs. Every ASP is associated with its partner ASP in the partner MERVA Link system. The name of the local ASP and the fully qualified name of the partner ASP (node_name.ASP_name) are the parameters that apply to a sending and to a receiving ASP.

The Message Transfer Process (MTP) names are additional ASP parameters that must be provided in all connections to MERVA ESA systems. MTP names are optional in connections between MERVA Workstation systems.

Sending ASP Parameters: A sending ASP can handle messages in up to three send queues. A MERVA application routes a message to one of the send queues of an ASP to get the message transferred to the applicable partner MERVA system.

Receiving ASP Parameters: A receiving ASP can route an incoming message to a receive queue. The receive queue name is a receiving ASP parameter.

A receiving ASP can correlate an incoming status report with a message that was previously sent. The name of the queue (Ack wait queue) that contains the previously sent message is a receiving ASP parameter.

Intersystem Connection Parameters: All parameters that apply to the intersystem connection to a specific partner system are collected in the set of ISC parameters. The set of ISC parameters is divided in the subset of the SNA APPC parameters, the TCP/IP parameters, and the ISC security parameters.

The subset of SNA APPC parameters contains the symbolic destination name. It is the name of a Side Information profile defined in Communications Server for Windows NT, the SNA service used by MERVA Link of MERVA.

The subset of TCP/IP parameters consists of the partner host name and the TCP/IP port number in the partner Windows NT host.

The subset of ISC security parameters consists of the client user name and the password that must be passed to the partner system for server access authorization.

Scheduling of MERVA Link

The MERVA Link sending process is started:

1. When MERVA is started, if the following is true:
 - Automatic Start is specified for the MERVA Link partner definition.
 - A send queue belonging to this ASP contains a message.
2. When Automatic Start is selected for a partner definition, and a message arrives in one of the defined send queues.
3. When you start an ASP using the operator function.
4. When you use the Kickoff operator function.

The MERVA Link sending process terminates:

1. When there are no more queues defined for the specific partner that contain messages and are in the nohold state. The MERVA Link sending process for this partner terminates normally.
2. After each non-recoverable error.

The MERVA Link receiving process is started by the Attach Manager of the applicable SNA services or by the MERVA Inetd service, if TCP/IP is used. The MERVA Link receiving process terminates:

1. When the sending partner has no more messages to send or an error occurs at the sending side. The session is deallocated and the receiving process subsequently terminates normally.
2. When an error occurs at the receiving side. The receiving process sends an error report to the sending partner and terminates.

Defining Details for MERVA Link

Select **Component data** and **MERVA Link** in the list boxes on the Customizer window shown in Figure 11 on page 32 to display the MERVA Link window shown in Figure 28 on page 60.

To customize MERVA Link:

1. Specify your **Local Node Name**
2. Define the partner node
3. Define ASP information

Note: You must define a partner node before you can refer to it in an ASP definition.

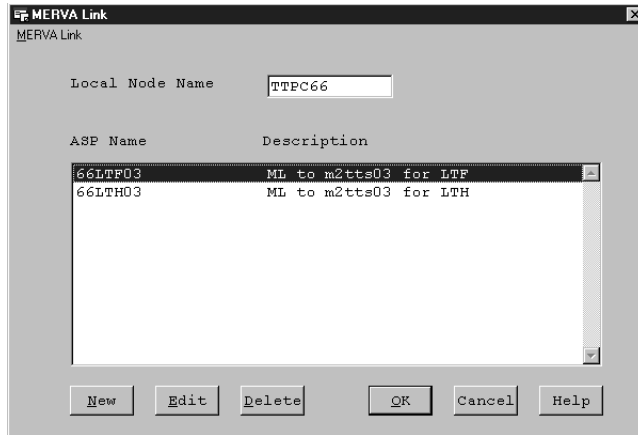


Figure 28. The MERVA Link Window

Click on:

- **New** to enter new ASP information on the ASP Information window shown in Figure 31 on page 63
- **Edit** or double-click an ASP to edit the ASP information on the ASP Information window shown in Figure 31 on page 63
- Click on **Delete** to delete an ASP

Defining Details for Partner Nodes

Select **Partner Nodes** on the **MERVA Link** pull-down menu of the MERVA Link window shown in Figure 28 to display the Partner Nodes window shown in Figure 29.

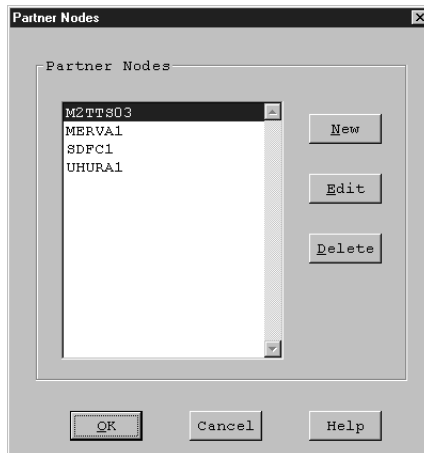


Figure 29. The Partner Nodes Window

Click on:

- **New** to create a new partner definition
- **Edit** to change the details of a partner definition
- **Delete** to delete a selected partner definition

The ISC Information window shown in Figure 30 on page 61 is displayed, if you select **New** or **Edit**.

Figure 30. The ISC Information Window

Use the ISC Information window to specify or change the required SNA APPC, TCP/IP, or conversation security information for the selected partner node.

- **Partner Node** is the name of the applicable MERVA Link partner node.
- SNA APPC Information

Symbolic Destination

Is the name of the Side Information Profile defined in the Communications Server for an SNA APPC connection to the partner system.

Transaction Program Name

Is the name of the Transaction Program. The Remote Transaction Program name in the Side Information Profile applies if this parameter is empty.

- TCP/IP Information

Host Name

Is the TCP/IP host name of the partner system. It can consist of up to 64 characters.

Port Number

Is the TCP/IP port number assigned to the MERVA Link message transfer server in the partner system.

- Conversation Security Information

The conversation security information is used when establishing a connection and must be valid at the partner system.

User ID

Is the client user ID that is passed to the partner system for client user authorization. The user ID is optional for an SNA APPC connection. It is mandatory for a TCP/IP connection.

Password

Is the client user password. It is optional for an SNA APPC connection, but it is mandatory for a TCP/IP connection.

Password Encryption Method

MERVA Link uses the selected password encryption method to encrypt the client user password before it is sent through the network.

The password encryption method **basic** is a MERVA proprietary algorithm.

- **Gateway**

You can use the gateway information to establish a link to another ISC information set.

Alternate

Is the name of a node that is used as an alternate gateway. If a connection to the partner node fails, MERVA Link tries to establish a connection by using ISC information represented by the node that is selected in the **Alternate** list.

Note: You can select a defined node name or type any other node name. If you type a node name, you must customize the required ISC information for this node.

Defining ASP Information

On the MERVA Link window shown in Figure 28 on page 60 you can do one of the following to display the ASP Information window shown in Figure 31 on page 63.

- **New** to fill in the new ASP information
- **Edit** or double-click an ASP to edit the ASP information

By default the status of the receiving ASP is enabled for new ASPs.

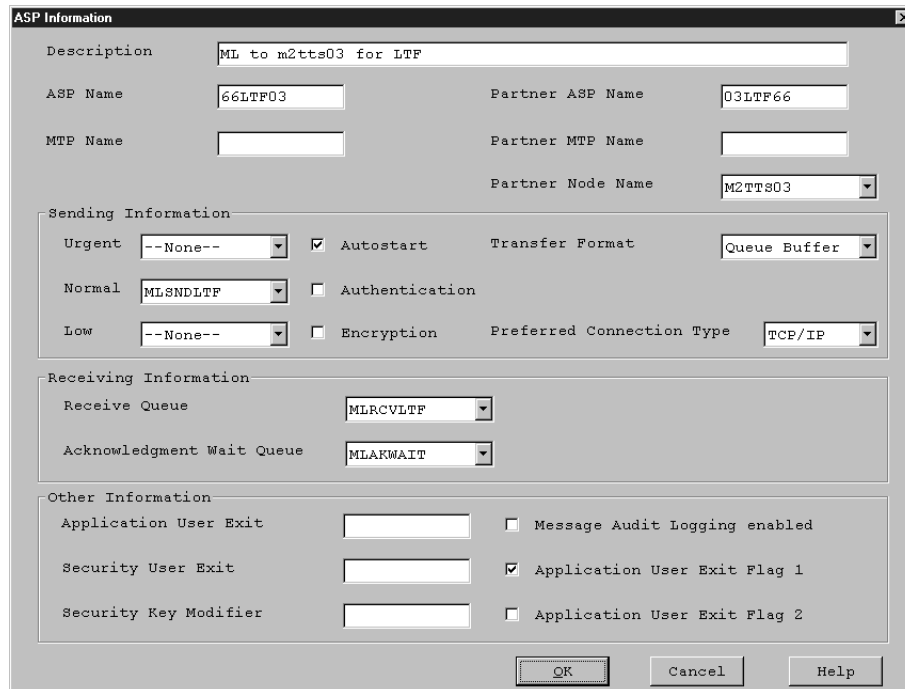


Figure 31. The ASP Information Window

The ASP Information window contains the following information:

- “Fields for Specifying the Description and Partner Names”
- “Fields for Defining the Sending ASP Information” on page 64
- “Fields for Defining the Receiving ASP Information” on page 65
- “Fields for Defining Other Information” on page 66

Fields for Specifying the Description and Partner Names

At the top of the ASP Information window shown in Figure 31 you can specify the following:

Description

A descriptive text for the ASP (ASP free form name).

ASP Name

The Name of a local ASP. This ASP must be defined when you create a new ASP.

Partner ASP Name

The name of the ASP in the partner system that is associated with the local ASP. This ASP must be defined as a local ASP in the partner system.

MTP Name

The name of the local Message Transfer Process (MTP) that is associated with the local ASP.

A MERVA Link MTP is the lower layer of the process that provides services for the sending Application Support Process (ASP) and requests services from the ASP when messages are received. This MTP must be defined as a Partner MTP in the partner system.

Partner MTP Name

MTP name of the partner system. This MTP must be defined as a local MTP in the partner system.

Partner Node Name

The MERVA Link node name of the partner system.

Fields for Defining the Sending ASP Information

You can specify the following information:

Queues (Urgent/Normal/Low)

These are the queues from where “ready-to-send” messages are sent to the partner ASP. Up to three send queues with different priorities can be selected.

The messages are sent in sequence according to the priority of the send queues: all messages in the Urgent queue are sent first, then those in the Normal queue, and finally those in the Low priority queue.

To select a queue, click on the down arrow in the combination box. Only queue names of the purpose group **MERVA Link Ready to Send** that are not already used for another ASP are listed.

For further information on MERVA Link queues and routing, refer to “Chapter 5. Defining the Message Routing” on page 35.

Autostart

Specifies if the sending process is started automatically when a message is routed to one of its “ready-to-send” queues, or only if the status of a send queue is explicitly changed from **hold** to **nohold**:

- Automatic activation of the sending process (flag is set)
When a message arrives in a send queue of an ASP, MERVA Link establishes the session with the appropriate partner and sends the messages from all send queues of this ASP. When all messages have been sent, the session is deallocated and the sending process ends.
- Sending process is started on user request (flag not set)
MERVA Link does not automatically activate a session. The user must start the sending process. When there are no more messages in send queues of this ASP, the ASP is set to **hold** status, the session is de-allocated, and the sending process ends.

Authentication

Defines whether authentication is required.

To ensure that changes made to a message during a transfer can be detected, you can request that each message that is transferred to the specified partner is authenticated.

During authentication, the sender generates a key derived from the contents of the message. This key is sent with the message to the receiver. The receiver then uses the same algorithm to generate a key from the message, and compares it to the key received with the message. The receiving process can then determine whether changes have been made to the message during the message transfer.

Encryption

Defines whether the message is to be encrypted before sending.

If selected, outgoing messages are encrypted. The encryption and decryption algorithms used are the same as those provided with MERVA ESA.

Note: If you select encryption, you must also select **Authentication** to check that the message text conversion is correct.

Transfer Format

MERVA Link can send and receive messages in a number of formats.

Select the format in which application messages are sent to the partner system. The format of the message is indicated in the message heading, so that the receiving process of the partner can respond accordingly. Selecting the **Transfer Format** only indicates the format in which messages are sent to the partner system, not the format in which received messages are expected.

The possible formats are Line and Queue Buffer:

- **Line**

When MERVA Link receives a message in line format from a partner system, it creates a new queue buffer with a new message reference number. The received message body is copied to the new queue buffer.

This transfer format is called “line format”, because the message body normally contains those parts of a SWIFT or telex message that are also sent to the SWIFT or telex network.

You can choose between the following:

EBCDIC The encoding of the message in line format is EBCDIC. For connections to MERVA ESA, this is the only format that can be selected.

ASCII The encoding of the message in line format is ASCII. This can only be selected for connections to MERVA Workstation systems.

- **Queue Buffer**

The entire queue buffer of a message is transferred to the partner system. Using this format, you can implement a form of transparent intersystem routing. The message appears in the receiving system as if it had been routed there locally. Only the message entry time and some MERVA Link control fields are changed in the buffer. This format can only be selected for connections to MERVA Workstation systems.

Note: Using this format the received message retains the original message reference number. Information held in fields such as the MSGACK field is also transferred.

To transfer a message in telex message processing format for further processing on the partner system select **Queue Buffer** as the transfer format.

Preferred Connection Type

Select SNA APPC or TCP/IP.

Fields for Defining the Receiving ASP Information

You can specify the following information:

Receive Queue

The name of the queue where MERVA Link places messages received from the specified partner node.

To select the receive queue, click on the down arrow in the combination box. The same receive queue can be specified for more than one ASP. Only queue names that were previously assigned to the purpose group **MERVA Link Received** are listed. The receive queue is the source queue for the

routing of messages received by a specific partner. For more information on MERVA Link queues and routing, refer to “Chapter 5. Defining the Message Routing” on page 35.

With the Message Routing Conditions window (see Figure 19 on page 42), you can define the target queue or queues that the received messages are routed to, and the conditions for routing messages to the appropriate queues.

Acknowledgment Wait Queue

The MERVA Link receiving process searches in this queue for messages with which a Status Report received from the specified partner is to be merged. After adding the information contained in the Status Report, the message is routed. For more information refer to “MERVA Link Remote Front-End Scenario” on page 67. In this scenario, messages are created on one system (called the “creating system”), and then sent to a second system (called the “front-end system”), where the Link component corresponding to the destination network is installed.

To select the queue, click on the down arrow in the combination. Only queue names that were previously assigned to the purpose group **MERVA Link Ack Wait** are listed. For further information refer to “Chapter 5. Defining the Message Routing” on page 35.

Fields for Defining Other Information

You can specify the following information:

Application User Exit

You can specify the name of the application user exit. For more information refer to “Appendix A. MERVA Link Application User Exits” on page 145.

Security User Exit

You can specify the name of the security user exit. This exit is executed to encrypt and authenticate the message text.

If you do not specify the name of the security user exit, but set the encryption flag, the standard encryption routine of MERVA is executed.

Security Key Modifier

You can specify a security key modifier. It is used to modify the message text encryption algorithm. Cooperating ASPs must use the same Security Key Modifier to exchange encrypted messages.

Application User Exit Flag (1 and 2)

When you use the default user exit modules, these flags enable the generation of status reports for a “remote front-end application”. See “MERVA Link Remote Front-End Scenario” on page 67 and “Application user exit” for details.

- Flag 1 enables the status report for SWIFT messages.
- Flag 2 enables the status report for telex messages.

Message Audit Logging Enabled

Select this item to log all incoming and outgoing messages via MERVA Link in the Message Audit Log.

MERVA Link Remote Front-End Scenario

MERVA Link allows you to generate a message on one system, send it to another MERVA system where the SWIFT Link component is installed, and from there forward it to the SWIFT network. You can customize MERVA Link to process SWIFT acknowledgements in this so-called remote front-end scenario.

In the following, the system named CREATING is the system where the message is generated, and FRONTEND is the system where the SWIFT Link is installed.

The CREATING System

The following steps are carried out on the CREATING system:

1. A message is created or loaded from a file. The SWIFT Link component is not used on the local system.
2. The message is routed to a MERVA Link send queue defined in the partner definitions for the FRONTEND system.
3. MERVA Link sends the message to the FRONTEND system and then it triggers the routing of the message. The routing is customized in such a way that the message arrives in the queue defined as the ACK Wait queue in the partner definition for the FRONTEND system.

The FRONTEND System

The following steps are carried out on the FRONTEND system:

1. The message is received by MERVA Link and routed to the SWIFT Link send queue.
2. SWIFT Link adds acknowledgment information to a message when it has been delivered to the network. Part of the acknowledgment information is copied to the MSGACK field.

This field is not part of the message in line-format, which resides in the message buffer of the MERVA queue buffer. SWIFT Link also changes some fields in the message buffer of the queue buffer.

3. In most cases, this information is required on the CREATING system and not on the remote FRONTEND system. Therefore, those parts of the message buffer that have been changed by SWIFT Link and the acknowledgment information contained in the MSGACK field are transferred back to the originating CREATING system, where it is received by MERVA Link.
4. If required, the MERVA Link on the CREATING system correlates the acknowledgment information contained in a message returning from the FRONTEND system with the corresponding message waiting in the ACK Wait queue. For this correlation, the unique IAM message identifier is used.

Returning the Acknowledged Message

When transferring the acknowledged message from the FRONTEND system to the CREATING system, several different types of transfer are possible:

- Returning the message in line-format
- Returning the entire queue buffer
- Returning a status report

Returning the Message in Line-Format

What Is Transferred: The following information is transferred:

- Acknowledgment information (MSGACK field)

- Information about authentication (MSGMAC and MSGPAC field)
- Message envelope data

Partner Definitions in the FRONTEND System: The following selections must be made in the partner definitions of the FRONTEND system, using the MERVA Customization program:

- Transfer Type is **Line ASCII** or **Line EBCDIC**
- Do not select **userexit control flag 1**

The Result: The contents of the MSGACK field is transferred back with the message. No correlation with the original message is performed by MERVA Link. A new message, containing the message-text part of the original message modified by the remote application (such as SWIFT Link) on the FRONTEND system, appears in the CREATING system. The original message still resides in the ACK Wait queue if it has previously been routed there.

Returning the Queue Buffer

What Is Transferred: When the entire queue buffer is being transferred, all acknowledgment information is transferred automatically.

Partner Definitions in the FRONTEND System: The following selections must be made in the partner definitions of the FRONTEND system using the MERVA Customization program:

- Transfer Type is **Queue Buffer**
- Do not select **userexit control flag 1**

The Result: The message is transferred and received as a normal application message. When it arrives in the CREATING system, it contains all information that was received by the remote application (such as SWIFT Link) on the FRONTEND system. No correlation with the original message is performed by MERVA Link.

Returning a Status Report

Using the terminology of the MERVA Link protocol, this status report is called an SR ASPDU (Status Report Application Support Protocol Data Unit). The status report contains only the acknowledgment information added to the message by the SWIFT Link on the FRONTEND system.

Partner Definitions in the FRONTEND System: The following selections must be made in the partner definitions of the FRONTEND system using the MERVA Customization program:

- Select **userexit control flag 1**

Status Report for SWIFT Messages

The following applies to status reports for SWIFT messages:

- Acknowledgment information is contained in the following SWIFT message elements:
 - MSGACK field
 - SWIFT basic header
 - SWIFT application header
 - SWIFT trailers

The SWIFT fields are extracted from the message and are added, together with the MSGACK field, to the status report.

- When merging the acknowledgment information with the original message, the MSGACK field is copied to the MERVA queue buffer and the SWIFT fields in the original message are replaced by those received with the status report.
- The acknowledgment code is extracted from the <accept-reject> tag ({451:<accept-reject>}) of the APDU 21 which resides in the MSGACK field.

Conditions for Generating and Sending a Status Report

The MERVA Link sends a status report when the following conditions are true:

- The message contains acknowledgment information from MERVA Link. A check is made whether the MSGACK field is filled.
- The message in the send queue contains an IAM Message Identifier. This is true when the message has been received by MERVA Link.
- The **userexit control flag 1** in the specific partner definition is set. This means that if it is a SWIFT message, the **userexit control flag 1** has been turned on.

Merging the Acknowledgment Information with the Original Message

When the original message on the creating system is to be correlated with the status report arriving from the FRONTEND system, it awaits correlation in the ACK Wait queue. This is the queue that was specified as the **ACK Wait queue** when the partner definition for the FRONTEND system was entered in the MERVA Customization program on the CREATING system. The message must be routed from the send queues defined for the partner FRONTEND to the ACK Wait queue.

When the MERVA Link on the CREATING system receives the status report from the FRONTEND system, the original message is retrieved from the ACK Wait queue. The IAM message identifier received with the status report is used to identify the original message.

The acknowledgment information contained in the status report is correlated and merged with the original message. The SWIFT Basic Header, the SWIFT Application Header, and the SWIFT Trailers are also replaced by those delivered with the status report. The message is then routed further, as determined by the routing definitions of the Customization program.

Routing Conditions for the CREATING System

Table 3. MERVA Link Queues Used on System A

Queue Name	Purpose Group	Already defined in sample routing?
MLSNDURG	MERVA Link Ready to Send	No
MLSNDNRM	MERVA Link Ready to Send	No
MLSNDLOW	MERVA Link Ready to Send	No
MLRECEIV	MERVA Link Received	Yes
MLAKWAIT	MERVA Link Ack Wait	Yes
MLCNTRL	MERVA Link Control	Yes

Table 4. Routing Conditions for the CREATING System

Messages in queue...	Are routed to...	When...
SMCREATE	MBDELETE	MSGOK = CANCEL
	MLSNDURG	MSGOK = OK and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBMERROR	Otherwise
SMINCMPT	MLSNDURG	MSGOK = OK and MSGCAT = MTYP_SYS
	MBDELETE	MSGOK = DELETE or MSGOK = CANCEL
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBMERROR	Otherwise
SMVERIFY	SMAUTH1	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMAUTH1	MLSNDNRM	MSGOK = OK and MSGNETID = SWIFNET and MSGPRITY = URGENT
	MLSNDLOW	MSGOK = OK and MSGNETID = SWIFNET and MSGPRITY = NORMAL
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMEDIT	SMVERIFY	MSGOK = OK
	MBDELETE	MSGOK = DELETE
	SMEDIT	MSGOK = INCOMPLT
	MBMERROR	Otherwise
MLAKWAIT	SLPRINT2	MSGCAT = MTYP_SYS
	SLRL1ACK	ACKINFO = ACK_ACC
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise

Table 4. Routing Conditions for the CREATING System (continued)

Messages in queue...	Are routed to...	When...
MLRECEIV	MLCNTRL	ALWAYS>= ROUTE and continue routing
	SLRCVSY	MSGTYPE = MTYP_010 or MSGTYPE = MTYP_011 or MSGTYPE = MTYP_015
	SLRCVSY	MSGTYPE = MTYP_066 or MSGTYPE = MTYP_082 or MSGTYPE = MTYP_083
	SLRCVSY	MSGTYPE = MTYP_021 and EMB_TYP1 <> MTYP_SYS
	SLPRINT2	MSGCAT = MTYP_SYS
	SLRCVNST	MSGGROUP = MTYP_19X or MSGGROUP = MTYP_29X
	SLRCVFIN	Otherwise
MLSNDURG	MLAKWAIT	MSGNETID = SWIFTNET
	MBMERROR	Otherwise
MLSNDNRM	MLAKWAIT	MSGNETID = SWIFTNET
	MBMERROR	Otherwise
MLSNDLOW	MLAKWAIT	MSGNETID = SWIFTNET
	MBMERROR	Otherwise
MLCNTRL	MBDELETE	

Routing Conditions for the FRONTEND System

Table 5. MERVA Link Queues Used on the FRONTEND System

Queue Name	Purpose Group	Already defined in sample routing?
MLSNDURG	Message Ready to Send	No
MLRECEIV	Message Received	Yes
MLCNTRL	Message Control	Yes

Table 6. Additional Fields for the FRONTEND System

Field Name	Explanation	Message Part	Scan Patterns		Offset	Length	Type
			Start Tag	End Tag			
BRANCH	Branch code of the address in the SWIFT Basic Header Block	MESSAGE BUFFER	{1:		12	3	Text

In the following table, **SAMP2A** denotes the node name of the CREATING system.

Table 7. Routing Conditions for the FRONTEND System

Messages in queue...	Are routed to...	When...
SLRSYS01	SLPRINT1	APDU = LOGIN or APDU = LOGOUT or APDU = ABORT_LT
	MLSNDURG	MLORGLU = "SAMP2A"
	SLPRINT2	ACKINFO = ACK_ACC or ACKINFO = ACK_REJ
	MBMERROR	Otherwise
SLRSYS02	MBMERROR	MSGNETID<> SWIFNET
	SLPRINT1	APDU = SELECT or APDU = QUIT or APDU = ABORT_AP
	MLSNDURG	MLORGLU = "SAMP2A"
	SLPRINT2	ACKINFO = ACK_ACC or ACKINFO = ACK_REJ
	MBMERROR	Otherwise
SLRURG02	MLSNDURG	MLORGLU = "SAMP2A"
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SLRL1ACK	ACKINFO = ACK_ACC
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise
SLRNRM02	MLSNDURG	MLORGLU = "SAMP2A"
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SLRL1ACK	ACKINFO = ACK_ACC
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise
SLINCMS1	UFROMSWF	MSGTYPE = MTYP_092
	SLPRINT1	APDU<>APDU_01
	MLSNDURG	BRANCH = "ABC"
	SLPRINT2	MSGCAT = MTYP_SYS
	MBMERROR	Otherwise

Table 7. Routing Conditions for the FRONTEND System (continued)

Messages in queue...	Are routed to...	When...
SLINCMS2	SLPRINT1	APDU<>APDU_01
	UFROMSWF	MSGGROUP = MTYP_96X or MSGTYPE = MTYP_087 or MSGTYPE = MTYP_076
	SLMANAUT	MSGINFO = AUTNOKEY or MSGINFO = AUT_FAIL
	SLMANAUT	MSGINFO = AUT_SUSP or MSGINFO = AUT_DISC
	MLSNDURG	BRANCH = "ABC"
	SLRCVSY	MSGTYPE = MTYP_010 or MSGTYPE = MTYP_011 or MSGTYPE = MTYP_015
	SLRCVSY	MSGTYPE = MTYP_066 or MSGTYPE = MTYP_082 or MSGTYPE = MTYP_083
	SLRCVSY	MSGTYPE = MTYP_021 and EMB_TYP1<>MTYP_SYS
	SLPRINT2	MSGCAT = MTYP_SYS
	SLRCVNST	MSGGROUP = MTYP_19X or MSGGROUP = MTYP_29X
SLRCVFIN	Otherwise	
MLSNDURG	MBDELETE	
MLRECEIV	MLCNTRL	ALWAYS >= ROUTE and continue routing.
	SLRSYS01	MSGCAT = MTYP_SYS and APPL = APC
	SLRSYS01	MSGCAT = MTYP_SYS and APPL = LTC
	SLRSYS02	MSGCAT = MTYP_SYS and APPL = FIN
	SLRURG02	MSGPRITY = URGENT
	SLRNRM02	MSGPRITY = NORMAL
	MBMERROR	ALWAYS >= ROUTE
<p>Note: If multiple branches (System A) are to be supported, the routing statements from SLINCMS1 and SLINCMS2 to MLSNDURG have to be differentiated by some criteria, for example, the branch code of the receiver of a SWIFT message.</p>		

Chapter 8. Customizing SWIFT USE

Select **Component data** and **SWIFT USE** on the Customizer window shown in Figure 11 on page 32 to display the USE Definitions window shown in Figure 32.

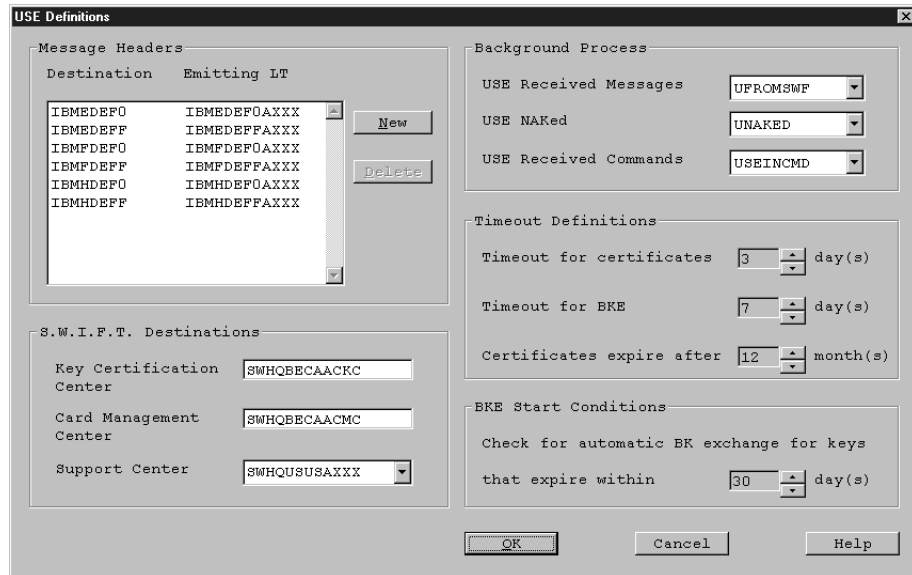


Figure 32. The USE Definitions Window

Use the USE Definitions window to define the following information:

- “Defining Message Headers”
- “Defining SWIFT Destinations” on page 76
- “Understanding the USE Background Process” on page 76
- “Defining Timeout” on page 77
- “Defining BKE Start Conditions” on page 78

Defining Message Headers

Define the emitting LTs for USE-related SWIFT messages. When you display this window for the first time, click on **New** to display the USE Destination window shown in Figure 33 on page 76 to define a new USE Destination.

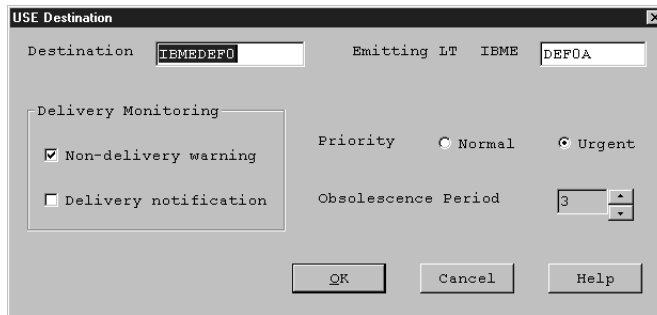


Figure 33. The USE Destination Window

Use the **Destination** and **Emitting LT** fields to define the emitting address of USE messages (the address of the sending LT). Define one LT for each destination. See the *MERVA USE Administration Guide* for further details.

In the **Destination** field, enter the following home destinations:

- Your own Key Management Authorities (KMA) used in all BKE pre-agreements
- For which the User Security Officer (USOF) maintains ICC cards and sets
- For which the User Key Management Officer (UKMO) maintains certificates

In the **Emitting LT** field, enter the 9-character LT identifier of the address to use for all USE messages. Do not change the 4-character bank code. You can, however, change the country and location codes.

The USE Destination window shown in Figure 33 also allows you to change the defaults of **Delivery Monitoring**, **Priority**, and **Obsolescence Period** for the SWIFT header.

Repeat this procedure until all home destinations have been defined.

Once you have defined a destination, you can change the information by selecting the entry from the list on the USE Definitions window shown in Figure 32 on page 75 and click on **Edit**.

Note: The **Edit** push-button appears when you select a destination in the list, and the **New** push-button appears when nothing is selected.

Defining SWIFT Destinations

You can change the destinations for USE-related SWIFT messages. Specify the U.S., Netherlands, or Hongkong Support Center. See the *S.W.I.F.T. Use Planning Guide* for details.

Understanding the USE Background Process

Define the queues from which the USE Background Process reads USE messages.

The USE Background Process:

- Receives and responds to Bilateral Key Exchange (BKE) messages. It also processes all USE-related messages from other MERVA systems.
- Cleans up the certificate blacklist by deleting all certificates from the database that have passed their expiry date.

- Reads messages from the following queues:
 - USE Received Messages:
 - For BKE messages (MT 96x) the appropriate response is created based on the BKE message protocol. The status of the key exchange and any data is stored in the database. A newly established bilateral key is sent to distribution if it is defined in the related pre-agreement.
 - BKE Initiation Requests (MT 960), for which no pre-agreement exists in the database, are routed to the queue for the Incoming MT 960 program. After processing, the Incoming MT 960 program routes the messages back together with the information to continue or cancel the key exchange. Refer to the *MERVA USE Administration Guide* for details on the Incoming MT 960 program.
 - Certification Responses (MT 087) are processed to store the new certificates in a Secure Card Reader (SCR). Certification Errors (MT 076) are used to update the status of a requested certificate.
 - Certificate Blacklist Response Messages (MT 092) are used to build the certificate blacklist.
 - USE NAKed:

Messages in this purpose group have been negatively acknowledged by the SWIFT network. They are processed to update the status of the bilateral key exchange in the database.
 - USE Received Commands:

This purpose group is for free-format messages (MT 999) that were created by another MERVA system for MERVA-internal USE processing:

 - For Session Key Requests a session key is read from a card reader and sent back within a new free-format message.
 - BK Update Messages are used to store or update a bilateral key in the database.
 - For Pre-Agreement Requests, the corresponding pre-agreement is sent back as a free-format message.
 - Pre-Agreement Update Messages are used to update the pre-agreement in the database.

By default, the USE Background Process is not activated. In this case, --None-- is specified for all queue fields. You can define whether or not the process is activated and which tasks are performed by specifying the related queues.

Defining Timeout

You can customize a number of USE-related timeout and expiry definitions:

- Timeout for Certificates

Defines the number of days the USE Background Process waits for a response to a Certification Request (MT 075) before setting a timeout condition.
- Timeout for BKE

Defines the number of days the USE Background Process waits for a response to a BKE Message (MT 96X) before setting a timeout condition.
- Certificates expire after

Defines the certificate expiry date for a newly requested certificate.

Defining BKE Start Conditions

You can customize the number of days a BKE is initiated prior to the planned effective date of a new future key.

The BKE Background Process automatically initiates bilateral key exchange for all correspondent pairs that fulfill the following conditions:

- The pre-agreement is approved.
- Automatic exchange has been defined in the pre-agreement.
- For bidirectional keys, the home destination is defined as initiator.
- No future key exists for which the home destination is defined as the initiator.
- The current key reaches the validity date defined in the pre-agreement, or no current key exists and the effective date defined in the pre-agreement is reached within the customized number of days. The effective date is always the date in the Greenwich Mean Time (GMT) time zone.
- No BKE is already in process for future keys for which the home destination is defined as the initiator.

The USE Background Process performs these tasks at each MERVA startup and at midnight.

The USE Background Process also checks the number of days between the last and the current automatic start check. This avoids an accidental automatic start of keys when the system date changes to a date in the year 2000. For example, if the system is used as a test system and is therefore not started regularly, the number of days can be greater than 30. In this case, ENNCBAT abends with abend code 332. To avoid this, increase the value of the environment variable ENM_DATE_DIFF. The maximum value of this variable is 999.

Supporting the SWIFT USE Card Reader

MERVA supports a USE card reader attached to:

- The local workstation. For detailed information about the card reader maintenance program, refer to the *MERVA USE Administration Guide*.
- Any remote workstation that is connected to the local workstation via TCP/IP.

Customizing Devices for the SWIFT USE Card Reader

To attach a SWIFT USE Card Reader to a local or remote workstation, you must define a COM port. To do this:

1. Click **Start** → **Settings** → **Control Panel**.
2. Double-click the **Ports** icon.
3. Check if a COM port is displayed, for example, **COM1**.
If no COM port is displayed, click **Add** and define a new port.
4. Double-click the COM port or click **Settings** to get the settings of the port.
5. You then get the Settings for COM Port window. The following figure shows an example of this window.

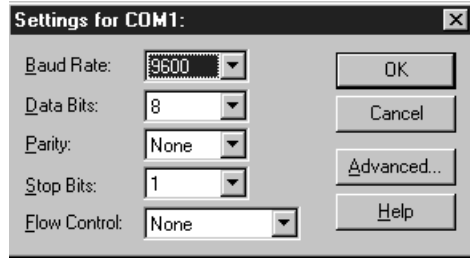


Figure 34. Settings for COM Port Window

6. Click **OK** to save the default settings.
7. Restart your workstation.

The COM port is defined with the default settings as shown in Figure 34. For more information about SWIFT Card Readers refer to the *S.W.I.F.T. Card Readers User Guide*.

Customizing TCP/IP for the Remote Card Reader Client

The Remote Card Reader Client of MERVA runs on the local MERVA host that does not have a USE card reader attached. It is contained in the library **enncreq.dll**.

You do not have to install MERVA on the remote host to get USE Card Reader services. These services are available after you install the MERVA Remote Card Reader Server and several programs that use TCP/IP to communicate with the MERVA Remote Card Reader Client.

You have to customize TCP/IP to activate the MERVA Remote Card Reader Server.

The Remote Card Reader Client

The Remote Card Reader Client of MERVA transfers function calls to its partner program, the Remote Card Reader Server. The address of the server is the TCP/IP host name and TCP port number. The address is part of the first function-call parameters.

The response data contains the returned parameters of the function call.

Customizing Hosts Table

For a detailed description of how to customize hosts table, refer to “Customizing Hosts Table” on page 114.

To customize the hosts table:

1. Change to the directory `%SystemRoot%\system32\drivers\etc`
2. Edit the file **hosts** by adding a text line, such as


```
1.123.1.235    mervar
```

where **1.123.1.235** is the IP address of the remote host with a USE card reader attached and **mervar** is the host name.

MERVA Card Reader Server

A MERVA Card Reader Server runs on the remote host that has a USE card reader attached. It is implemented in the program **enncrs.exe**.

The Remote Card Reader Server Program `enncrs`

The Remote Card Reader Server is started as a subservice of the MERVA Inetd service when an inbound call is received via the TCP port defined for that subservice. The attached card reader is referred to as the remote host.

The Remote Card Reader Server calls the TCP/IP socket services to receive the card reader request parameters and data from the partner Windows NT system. It also calls to return the card reader response parameters and data to the partner Windows NT system.

Customizing TCP/IP for the Remote Card Reader Server

You have to define the Remote Card Reader Server of MERVA as a Client Network Service on the remote host. The definition specifies the connection of the TCP port number and the symbolic service name for the Remote Card Reader Server of MERVA.

Customizing Client Network Services: To customize the Client Network Service:

1. Change to the directory `%SystemRoot%\system32\drivers\etc\`.
2. Edit the file `%SystemRoot%\system32\drivers\etc\services` by adding a text line, such as:

```
rcr1      7119/tcp
```

where:

- **rcr1** is a symbolic service name for the Remote Card Reader connection.
- **7119/tcp** defines the IP port number 7119 for the connection and specifies that the TCP protocol is required.

Customizing the MERVA Inetd Service: To customize the MERVA Inetd service:

1. Change to the directory `%SystemRoot%\system32\drivers\etc\`.
2. Edit the existing configuration file `enminetd.cfg` by adding a text line, such as

```
rcr1  stream tcp  nowait root d:\merva_ws_rcrs\bin\enncrs.exe COM1
```

where:

- **rcr1** is a symbolic service name for the Remote Card Reader connection. This name must be identical to the name defined in the file `%SystemRoot%\system32\drivers\etc\services`.
- The executable file `d:\merva_ws_rcrs\bin\enncrs.exe` is the Remote Card Reader server program.
- **COM1** denotes the COM port to which the card reader is connected.

Installing the MERVA Card Reader Server: To install the MERVA Remote Card Reader Server on the remote host:

1. Log on with the user ID that has Windows NT administration authorization.
2. Insert the CD labeled MERVA in the CD-ROM drive.
3. Select the CD-ROM drive.
4. Change to the directory `MERVA_Features\Remote_Cardreader_Server`.
5. Double-click the file `setup.exe` to start the setup program.

Note: MERVA USE & Branch for Windows NT has the same functions as the MERVA Card Reader Server. If MERVA USE & Branch for Windows NT is installed on a remote host, you do not have to install the MERVA Card Reader Server additionally on this remote host.

Chapter 9. Customizing Other MERVA Parameters

This chapter describes how to:

- Customize printer conditions
- Maintain alarms
- Maintain currency codes
- Set the date format
- Activate destination checking
- Set logging levels

Customizing Printer Conditions

Automatic Printing

Select **Component data** and **Automatic Print** on the Customizer window shown in Figure 11 on page 32 to display the Automatic Print window shown in Figure 35.

Note: You can only change the parameters. To create or delete a print queue refer to “Setting up Message Queues” on page 36.

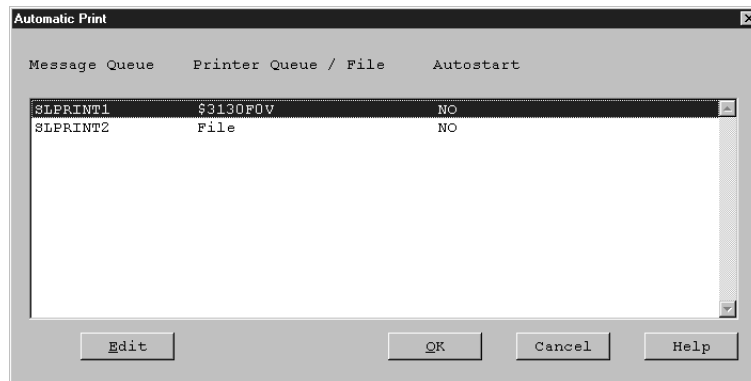


Figure 35. The Automatic Print Window

MERVA Automatic Print provides a Background Process that spools messages from MERVA queues to operating system printer queues or files.

The Automatic Print window shown in Figure 35 lists all MERVA print queues together with their defined settings. Assign each queue belonging to the Print purpose group to an operating system printer queue or file, and specify whether printing starts immediately after the MERVA startup or whether it is initiated by the operator.

Initially, all printing is directed to files with no automatic start.

Select a queue and click on **Edit**, or double-click on a queue to display the Print Parameters window shown in Figure 36 on page 82 and to change the message printing settings.

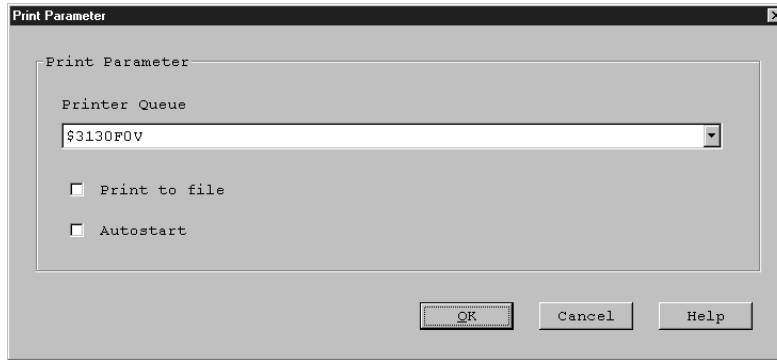


Figure 36. The Print Parameters Window

On the Print Parameter window select the **Printer Queue** from the combination list box or select the **Print to File** check box. Note that only printer queues with a name of up to 64 characters are shown.

If you select **Print to File**, messages are printed to a file named **autoprt.nnn**, where *nnn* is a unique 3-digit number that begins with **001** and is incremented by 1 for each print queue created.

The print files are located in the MERVA instance home directory. See “Creating a MERVA Instance” on page 14 for details.

Select the **Autostart** check box if you want to start printing messages from this queue immediately after the MERVA startup.

Select the **Automatic Print** user function in the MERVA Main Menu to start and stop the printing process for selected queues. See the *MERVA USE & Branch for Windows NT User's Guide* for details.

Selecting the Default Printer

Select **System configuration** and **Default Printer** on the Customizer window shown in Figure 11 on page 32 to define the default print destination for all user-initiated printing.

This default is used as long as the users do not select their own specific printer from the **Setup** pull-down menu in the MERVA Main Menu.

Notes:

1. After installation, the default printer is undefined. Select a default printer from the list of printers defined in your operating system.
2. Only printer queues with a name of up to 64 characters are shown.

The Message Print Separator

Select the **System configuration** and **Message Print Separator** on the Customizer window shown in Figure 11 on page 32 to display the Message Print Separator window shown in Figure 37 on page 83.

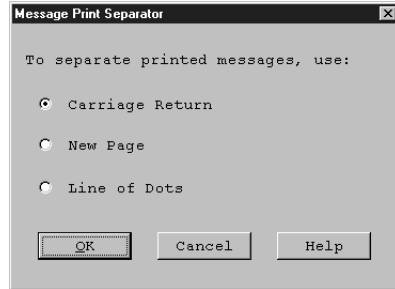


Figure 37. The Message Print Separator Window

To customize the message separator, you must first set the Form Feed option of the printer to **None**.

Select one of the following:

Carriage Return

Separate printed messages with a blank line. This is the initial setting.

New Page

Print each message on a new page.

Line of Dots

Print a dotted line across the page after each message.

Note: Messages are printed with expanded address information.

Maintaining Alarms

Select **System configuration** and **Alarms** on the Customizer window shown in Figure 11 on page 32 to display the Alarms window shown in Figure 38.

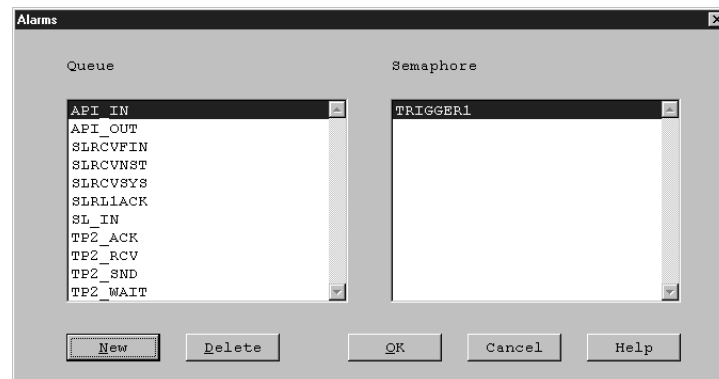


Figure 38. The Alarms Window

The Alarms window lists all queues of the purpose group API and the corresponding signals (semaphores). Click on **New** to create a new alarm. Click on **Delete** to delete an alarm.

An alarm is a signal (semaphore) used to trigger the execution of an application program. You can associate an alarm with queues belonging to the purpose group API. By associating an alarm with a message queue, you cause the program associated with the alarm to be notified whenever a message arrives in that queue.

Alarms are normally used for application programs that unload messages from MERVA for processing by another software package. Using alarms, these programs do not have to continuously query for new messages, thus reducing the system load.

You can, however, use the alarm facility to trigger any type of event for any type of message arriving in any queue of MERVA.

For example, to send an electronic note to a person when a message arrives in the queue for Manual Authentication of SWIFT messages. Route a copy of the message to an API queue by specifying the appropriate routing condition. Then write an application program that is triggered by the alarm, reads, and deletes the message copy, and sends the note to the user. Following this example, you can trigger all types of events using the appropriate routing condition to select the desired messages and an application program that can interpret the message content to decide on the action to take.

Refer to the *MERVA USE & Branch for Windows NT Application Programming Guide* for details on how to use alarms in application programs.

Currency Code Customization

Using the Currency Codes window shown in Figure 39 you can add, change, or delete currency codes. This allows you to update currency data immediately after receiving the information from SWIFT Select **System configuration** and **Currency Codes** on the Customizer window shown in Figure 11 on page 32 to display the Currency Codes window shown in Figure 39.

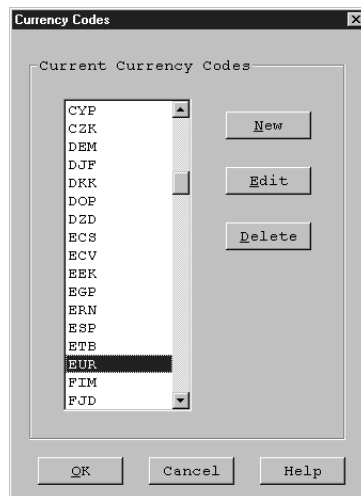


Figure 39. The Currency Codes Window

You can select to do the following:

- To create a new currency code, click on **New** to display the window shown in Figure 40 on page 85.

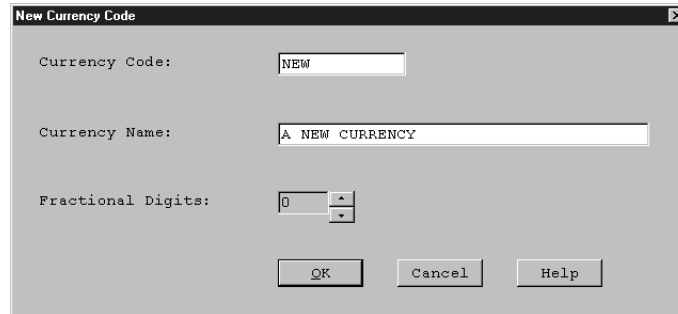


Figure 40. The New Currency Code Window

Enter the new currency code, its name (optional), and the number of fractional digits allowed for it. The default for fractional digits is 0.

Click **OK** to save the new currency code.

- To edit a currency code, select an item from the list and click **Edit**. You can then change the name of the currency code and the number of fractional digits. Click **OK** to update the currency code.
- To delete a currency code, select one or more items from the list and click **Delete**.

Setting the Date Format

Select **System configuration** and **Date Format** on the Customizer window shown in Figure 11 on page 32 to display the Date Format Select window shown in Figure 41.

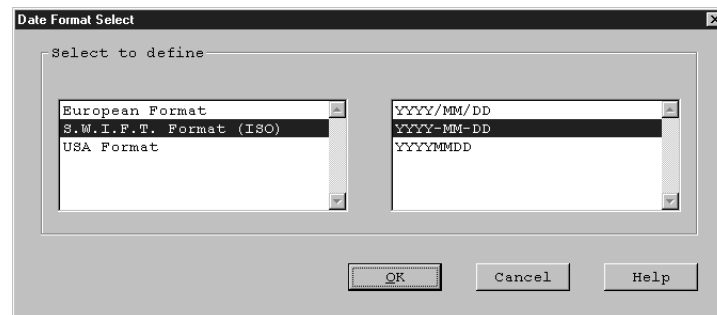


Figure 41. The Date Format Select Window

Select the desired format to define the order of the day (*DD*), month (*MM*), and year (*YYYY*) fields, and the delimiter used to separate the date fields; a forward slash (/), a dash (—), or no delimiter.

MERVA is supplied with the SWIFT format (*YYYY-MM-DD*) with dashes as the defaults. Select the layout and the delimiter to use and click on **OK**.

Note: Dates in SWIFT messages are always in SWIFT format.

Destination Checking

Select **System configuration** and **Destination Checking** on the Customizer window shown in Figure 11 on page 32 to activate checks for destinations entered in the Create Message and Edit Message functions.

When you activate authentication key checking, the system will check if a valid authentication key exists for the entered destination. If it does not exist a warning is displayed.

Setting Logging Levels

Select **System configuration** and **Logging Level** on the Customizer window shown in Figure 11 on page 32 to display the Logging Level window shown in Figure 42. The **Programmer's Trace Log** level is represented by an integer between 1 and 4.

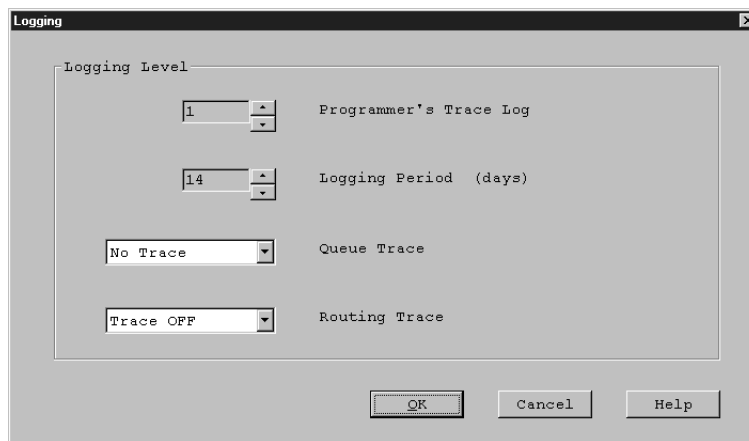


Figure 42. The Logging Level Window

All log messages at or below the chosen logging level are written to the trace log file. A logging level of 1 causes only severe error messages to be written to the trace log file, while a logging level of 4 causes all messages to be written to the trace log file.

For a detailed description refer to the *MERVA USE & Branch for Windows NT Diagnosis Guide*.

The **Logging Period** is the number of days that entries in the programmer's trace and diagnosis log are kept before the file is renamed to become a backup file and a new empty file is started. A previously existing backup file is deleted.

The **Queue Trace** log level is initially switched off (**No Trace**). You can also specify a small or large trace log level to generate a queue trace log:

- With a **Small Trace** log level the queue trace record lists parameters and return codes of queue management requests.
- With a **Large Trace** log level the queue trace record contains, in addition to the information for a small trace, the contents of the message and all related information.

When a **Routing Trace** is active (**Trace On**), the execution of each routing table entry and the parameter values determining routing are recorded in the diagnosis log. This information can be useful when routing table problems occur.

Note: MERVA is supplied with a minimum programmer's trace log, a 14-days log period, and no queue and routing trace log as default. Setting high log levels can affect system performance and requires large amounts of disk storage space.

Important:

As the information recorded in the trace logs is used for error diagnosis, the logging levels should only be changed at the request of an IBM representative or when you are testing your own routing tables.

These log levels can also be set temporarily with the **Setup** pull-down menu in the MERVA Main Menu. See the *MERVA USE & Branch for Windows NT User's Guide* for further details.

Chapter 10. Exporting and Importing Customization Data

You can export and import customization data to ease the update, distribution, or exchange of this data. During export, the customization data is written from the MERVA control database to ASCII files with a fixed format. The file extensions identify which component they refer to, as shown in Table 8 on page 91. You can edit these files or copy them to a diskette for later update or customization-data transfer. During import the files are loaded into the MERVA control database.

This reduces the time-consuming and maybe redundant online update using the window dialog described in “Customization Procedure” on page 31, and ensures similar working conditions in different MERVA environments.

Note: Before using the import and export commands, you should be familiar with the MERVA customization requirements described in:

- “Customization Procedure” on page 31
- The chapter on defining user access rights in the *MERVA USE & Branch for Windows NT User's Guide*.

Using the Export and Import Commands

Prerequisites to export or import customization data are:

- MERVA is running in customization mode.
For information on how to start MERVA in customization mode, refer to “Chapter 4. Starting MERVA USE & Branch Customization” on page 31.
- No MERVA Main Menu is started.
- You are a member of the MERVA administrator group **mervasys**.

The following sections describe the export and import commands in detail.

Export Command

To export customization data, type the following command in a command prompt window:

```
enmcxexp -f filename -c component_value
```

The export command creates separate files for each component, identified by their extensions as shown in Table 8 on page 91.

You can specify the following parameters for this command:

-f filename Specifies the ASCII file name of the customization data you want to export.

If the specified file name does not exist, a file is created for the defined component. If the specified file exists, its content is overwritten.

Note: You must not specify an extension, because it is predefined as shown in Table 8 on page 91.

-c component_value Specifies a number for the MERVA components you want to

export. This number is the sum of the values defined for the MERVA components in Table 8 on page 91.

Example: To export customization data for the MERVA Link component (a value of 2) and automatic print (a value of 16) specify the following command in a command prompt window:

```
enmcxexp -f oldinfo -c 18
```

As a result the following ASCII files are created:

- oldinfo.MLI for MERVA Link
- oldinfo.APR for automatic print.

Import Command

To import customization data, type the following command in a command prompt window:

```
enmcximp -f filename -c component_value [-t]
```

You can specify the following parameters for this command:

- f *filename*** Specifies the ASCII file name of the customization data you want to import.
- c *component_value*** Specifies a number for the MERVA components you want to import. This number is the sum of the values defined for the MERVA components in Table 8 on page 91.
- t** Indicates that the customization data should not be imported, but tested. All data is read and checked as described in “Importing Customization Data” on page 91, but the MERVA control database is not updated.

Example: To test the import of the previously exported customization data for the MERVA Link component (a value of 2) and automatic print (a value of 16) specify the following command in a command prompt window:

```
enmcximp -f oldinfo -c 18 -t
```

The data of the following ASCII files is read and validated, but not loaded into the MERVA control database:

- oldinfo.MLI for MERVA Link
- oldinfo.APR for automatic print.

Components and Files

Table 8 on page 91 defines the values for the components you specify when entering the export or import commands. It also shows the predefined extensions provided for each component.

Table 8. Overview of Components: Predefined Extensions and Values

MERVA Component	ASCII File Name	Value
Routing (names and conditions)	<i>filename.ROU</i>	1
MERVA Link	<i>filename.MLI</i>	2
SWIFT Link	<i>filename.SLI</i>	4
SWIFT USE	<i>filename.USE</i>	8
Automatic Print	<i>filename.APR</i>	16
Alarm Maintenance	<i>filename.ALR</i>	32
User Access Rights	<i>filename.USR</i>	64

Exporting Customization Data

The export command writes the currently valid customization data to ASCII files as specified in Table 8.

The exported customization data for user access rights contains all access rights, including those access rights that are in **update** status. Passwords are not exported.

Importing Customization Data

During import the customization data defined in the ASCII files is loaded to the MERVA control database. Before data is loaded to the MERVA control database, it is checked if the:

- Syntax of the customization data specified in the ASCII files is correct
- Customization data is consistent

If one of these checks fails, the data is not loaded to the MERVA control database.

To import customization data, it is recommended to:

1. Back up the MERVA control database.
2. Export all components of the customization data.
3. Update the export files according to your requirements.
4. Check the import files first by specifying the import command **enmxcimp** with the **-t** parameter.
5. If item 4 completed successfully, you can import your customization data. It is recommended to import the customization data for all components at the same time.

Importing Data for Alarm Maintenance

Existing user-defined alarms are deleted. It is checked if the queues listed in the ASCII file are defined in the correct MERVA purpose group.

Importing Data for Automatic Print

Import is done in update mode. It is checked if the:

- System printers exist in the operating system
- Queues are defined in the correct MERVA purpose group
- Queues exist already in the MERVA instance

Importing Data for Routing

The following data is read and checked:

- Routing Names

For queues it is checked, if the specified purpose group is defined in MERVA.

Note: It is not checked, if the maximum number of queues allowed for a purpose group is exceeded. For message creation, for example, only one queue can be defined.

For fields it is checked, if the sum of offset, length, and length of the start tag exceeds the length of a message part, to which this field was assigned.

- Routing Conditions

If the default target queue has not been defined, MBMERROR is used as the default target queue. It is also checked, if the defined queue is an allowed target queue. SMCREATE, for example, must not be a target queue.

A condition must have been defined for each target queue. This condition specifies the criteria for routing messages to this queue.

Note: You must not mix AND and OR operators within a single routing condition.

Importing Data for MERVA Link

The following data is read and checked:

- Data for the local node
- Data for the partner nodes (ISC)
- Data for the Application Support Processes (ASP)
 - The ASP block must contain a parameter that relates to the partner node.
 - Each send or receive queue must exist in the names table and must be defined for the correct purpose group (Send to Send, Receive to Receive, ACK-Wait to ACK-Wait).
 - Each send, receive, or ACK wait queue must be defined only once.
 - You must define at least one send queue and one receive queue for each ASP.
 - Within the definition of one ASP, queues must be unambiguously defined. (For example, you must not define a queue as Urgent and Normal within one ASP definition.)
 - The partner node password is not specified.

Importing Data for SWIFT Link

The following data is read and checked:

- Data for the Logical Terminals including the information for the queues of the GPA and FIN applications
- Data for the network
- Data for the FIN copy services including the defined data for the message types

Importing Data for User Access Rights

The following data is read and checked:

- Data for the general access rights in the MERVA system including the queues that can be assigned to these rights
- Data for the users defined in MERVA

The imported user access rights have not yet been approved. They are in **added** status.

Note: After import, users cannot work with MERVA. The updated access rights of the users must first be approved.

During import the initial MERVA user ID **merva** is created with the initial MERVA password **merva6k**, as provided during the installation. All MERVA passwords are reset to the initial password **merva6k**.

To approve the access rights of the users after import, you must:

1. Log on to Windows NT with the user ID **merva**
2. Log on to MERVA
3. Select **Administration** and **Users** on the MERVA Main Menu window to approve the access rights defined for users as described in the *MERVA USE & Branch for Windows NT User's Guide*.

Importing Data for SWIFT USE

The following data is read and checked:

- Data for the message header for the destinations. Priority and Deliv_Notification must be defined for each destination.
- Data for the queues for the BKE Background Process.
- Data for the SWIFT definitions.

Support_Center must select only one of the support centers defined by MERVA.

Timeout definitions must be within the following limits:

- 1 ≤ Timeout_for_Cert lower than or equal to 7
- 1 ≤ Timeout_for_BKE lower than or equal to 14
- 1 ≤ Cert_exp_after lower than or equal to 24

The queues of the Background Process must be defined in MERVA as queues in the correct purpose groups.

Understanding the Keywords Used in the ASCII Files

The export and import commands process ASCII files that contain the customization data for the specified components. Refer to Table 8 on page 91 for an overview of the supported components and the file naming conventions. Here you find detailed information on the keywords and values used in the customization definitions of these ASCII files. References to the corresponding window dialog described in "Customization Procedure" on page 31 and the *MERVA USE & Branch for Windows NT User's Guide* are also provided.

Notational Convention

The syntax is based on a reduced IBM Configuration, Installation, and Distribution (CID) response file format. Response files are ASCII flat files that contain sequences of keyword=value pairs that are interpreted for customization purposes. The response files contain predefined answers to questions that you must normally specify in a window dialog. For more information on response files refer to the *CID Enablement Guidelines*.

The following line formats are allowed:

- Comment lines, starting with a semicolon (;).
- Response lines, such as:

```
keyword = (  
    keyword = "value"  
    ...  
    [ keyword = "value" ]  
)
```

The *keywords* allowed for the different components are listed in the following sections. Keyword expressions are only valid if they are specified in the correct context and positions within the parenthesis. Bracketing parenthesis must be specified on separate lines.

Value strings must be enclosed in double quotes. Table 9 specifies the valid entries for value strings. The value length is identified by the number of characters specified in the symbol string (such as *uuuuuuuuu*).

Table 9. Valid Entries for Keywords

Character Symbols	Changed into Uppercase	Valid Entries
<i>0</i>	-	Numbers, hyphen (-)
<i>1</i>	-	Numbers, colon (:)
<i>9</i>	-	Numbers (unsigned)
<i>a</i>	No	Alphanumeric characters, blank, period (.)
<i>A</i>	No	Alphanumeric characters
<i>b</i>	No	Alphanumeric characters, underscore (_)
<i>B</i>	-	Binary (0 or 1)
<i>e</i>	No	Letters
<i>E</i>	Yes	Letters
<i>G</i>	Yes	Alphanumeric characters
<i>h</i>	No	Hexadecimal (0-9, A-F)
<i>H</i>	Yes	Hexadecimal (0-9, A-F)
<i>i</i>	-	Numbers
<i>l</i>	No	Only the length is checked
<i>L</i>	Yes	Only the length is checked
<i>n</i>	Yes	Alphanumeric characters, number sign (#), dollar sign (\$), at sign (@)
<i>u</i>	Yes	Alphanumeric characters, underscore (_)
<i>U</i>	Yes	Alphanumeric characters
<i>x</i>	No	Any entries, except for blank
<i>X</i>	Yes	Any entries, except for blank
<i>Y</i>	No	Alphanumeric characters, percent (%)

Alarm Maintenance: Keyword List

Table 10 lists the keywords allowed for customizing data in the ASCII file (*filename.ALR*) of the alarm maintenance component.

Table 10. Keyword List and Description for Alarm Maintenance

Keyword	Meaning	Values	Window Dialog Reference
Alarms	Header of the structure for Alarms	-	"Maintaining Alarms" on page 83
Alarm_ <i>i</i>	Header of the structure for alarm number <i>i</i>	-	"Maintaining Alarms" on page 83
Alarmname	Semaphore name of the queue	A character string of <i>bbbbbbb</i>	"Maintaining Alarms" on page 83
Queue_Name	Queue name. This queue name must be a defined queue in the purpose group API.	A character string of <i>uuuuuuuu</i>	"Maintaining Alarms" on page 83

A sample definition of customization data in the ASCII file of the alarm maintenance component is:

```
Alarms = (  
    Alarm_1 = (  
        Alarmname = "Ala_Test"  
        Queue_Name = "SLRCVFIN"  
    )  
    ...  
)  
; End Alarms;
```

Automatic Print: Keyword List

Table 11 lists the keywords allowed for customizing data in the ASCII file (*filename.APR*) of the automatic print component.

Table 11. Keyword List and Description for Automatic Print

Keyword	Meaning	Values	Window Dialog Reference
Automatic_Print	Header of the structure for Automatic_Print	-	"Automatic Printing" on page 81
Print_Queue_ <i>i</i>	Header of the structure for print queue number <i>i</i>	-	"Automatic Printing" on page 81
Name	Name of the message queue	A character string of <i>uuuuuuuu</i>	Figure 35 on page 81
Print_to_file	Specifies, if Print to File is selected	"YES" or "NO"	Figure 36 on page 82
Printer	Name of the printer (queue)	A character string as defined in the operating system	Figure 36 on page 82
Autostart	Specifies, if Autostart is selected	"YES" or "NO"	Figure 36 on page 82

A sample definition of customization data in the ASCII file of the automatic print component is:

```
Automatic_Print = (
    Print_Queue_1 = (
        Name = "SLPRINT1"
        Print_to_file = "YES"
        Autostart = "NO"
    )
    Print_Queue_2 = (
        Name = "SLPRINT2"
        Print_to_file = "NO"
        Printer = "pr_001"
        Autostart = "NO"
    )
    ...
)
; End Automatic Print;
```

Routing: Keyword List

Table 12 lists the keywords allowed for customizing data in the ASCII file (*filename.ROU*) of the routing component:

Table 12. Keyword List and Description for Routing

Keyword	Meaning	Values	Window Dialog Reference
Names_Table	Header of the structure and the names defined in MERVA	-	"Setting up Message Queues" on page 36

Defining Message Queues: Keyword List

Queue	Header of the structure for defining a name of type QUEUE	-	"Setting up Message Queues" on page 36
Name	Name of the defined queue	A character string of <i>uuuuuuuu</i>	"Setting up a New Queue" on page 36
Purpose_Group	The purpose group to which the queue is assigned	A character string as defined by MERVA	Figure 12 on page 36

Defining Message Fields: Keyword List

Field	Header of the structure for defining a name of type FIELD	-	"Defining Message Fields" on page 37
Name	Name of the defined field	A character string of <i>uuuuuuuu</i>	"Setting up Message Queues" on page 36 Figure 14 on page 38
Offset	Field offset	A character string of <i>99999</i>	"Offset" on page 38
Start_Tag	String as a start tag	A character string of <i>LLLLLL</i>	"Start Tag" on page 38
End_Tag	String as an end tag	A character string of <i>LLLLLL</i>	"End Tag" on page 38
Length	Length of the defined field	A character string of <i>99999</i>	"Length" on page 38
Message_Part	Specifies to which message part the defined field belongs	One of the following character strings: "MSGOK FIELD" "MSGACK FIELD" "MSGROUTE FIELD" "MSGCOMM FIELD" "MSGUSER FIELD" "MRN FIELD" "MSGTXT1 FIELD" "MSGTXT2 FIELD" "MSGTXT3 FIELD" "MSGTXT4 FIELD" "MSGNETID FIELD" "MESSAGE BUFFER" "SWIFT TEXT" "MESSAGE TEXT"	"Message Part" on page 37
Scan	Specifies if scan was selected	"YES" or "NO"	"Scan" on page 38
Datatype	Data type TYPE	"TEXT" or "NUMERIC"	"Type" on page 39

Defining Constants: Keyword List

Constant	Header of the structure for defining a name of type CONSTANT	-	"Defining Constants" on page 39
Name	Name of the defined constant	A character string of <i>uuuuuuuu</i>	"Defining Constants" on page 39
Actual_Value	Specifies the defined value of a constant (The constant equates to list box)	A string of up to 30 characters. This string must be enclosed in double quotes.	Figure 16 on page 39

Defining Routing Conditions: Keyword List

Routing_Conditions	Header of the structure containing all routing conditions defined in MERVA	-	"Setting up Routing Conditions" on page 40
Messages_from_Queue	Header of the structure for a routing condition	-	"Setting up Routing Conditions" on page 40
Source_Queue	Specifies the source queue for a routing condition	A character string of <i>uuuuuuuu</i>	"Source Queue" on page 40
Default_Queue	Specifies the default (send) queue for a routing condition	A character string of <i>uuuuuuuu</i>	Figure 18 on page 41, field Failing all above ...
Target_Queue_i	Header of a target queue in a routing condition. Can be defined several times for a source queue.	-	"Target Queue" on page 41
Queue_Name	Queue name of the specified target queue	A character string of <i>uuuuuuuu</i>	Figure 19 on page 42, field A message in queue ...
Condition1 ... Condition4	Header of the structure for defining a condition for a send queue	-	41
Field	Comparison FIELD in the defined condition	A character string of <i>uuuuuuuu</i>	"FIELD" on page 42
Operator	Specifies a logical operator in a condition	Can be one of the following: " = " " > " " < " " >= " " <= " " <> " " is not FOUND" " is EMPTY"	"OPERATOR" on page 42
Value	Comparison constant or value in a condition definition	A character string of <i>uuuuuuuu</i> . Numeric or string values must be enclosed in double quotes.	"OPERAND" on page 43

Concatenation	Concatenation of several conditions (structures) defined for a target queue	One of the following character strings: " and " " or " Do not mix these operators.	"Boolean Operators" on page 43
Further_Routing	Specifies if Terminate routing or Continue is selected	Specify "YES" for Continue or "NO" for Terminate routing	"Continue" on page 43

A sample definition of customization data in the ASCII file of the routing component is:

```

Names_Table = (
  Queue = (
    Name = "FIN_I_1"
    Purpose_Group = "SWIFT Ready to Send"
  )
  ...
  Field = (
    Name = "APDU"
    Start_Tag = "{1:"
    Offset = "1"
    Length = "2"
    Message_Part = "MESSAGE BUFFER"
    Scan = "YES"
    Datatype = "NUMERIC"
  )
  ...
  Constant = (
    Name = "AUTNOKEY"
    Actual_Value = ""ENM9979""
  )
  ...
)
Routing_Conditions = (
  Messages_from_Queue = (
    Source_Queue = "SL_IN"
    Default_Queue = "MBMERROR"
    Target_Queue_1 = (
      Queue_Name = "SLRSYS01"
      Condition1 = (
        Field = "MSGCAT"
        Operator = " = "
        Value = "MTYP_SYS"
      )
      Concatenation = " and "
      Condition2 = (
        ...
      )
      Further_Routing = "NO"
    )
  )
; End Target Queue SLRSYS01
  ...
)
; End Source Queue
  ...
)
; End Routing;

```

MERVA Link Component: Keyword List

Table 13 lists the keywords allowed for customizing data in the ASCII file (*filename.MLI*) of the MERVA Link component:

Table 13. Keyword List and Description for MERVA Link

Keyword	Meaning	Values	Window Dialog Reference
MERVA_Link	Header of the structure for the MERVA Link data	-	"Chapter 7. Customizing the MERVA Link" on page 57
Local_Node	Header of data for the local node	-	"Chapter 7. Customizing the MERVA Link" on page 57
Nodename	Name of the local node	A character string of <i>nnnnnnnn</i>	Figure 28 on page 60

Defining Details for Partner Nodes: Keyword List

Partner_Nodes	Header of data for the partner nodes (ISC)	-	"Defining Details for Partner Nodes" on page 60
Partner_Node	Header of the structure for a partner node	-	"Defining Details for Partner Nodes" on page 60
Nodename	Name of the partner node	A character string of <i>nnnnnnnn</i>	Figure 29 on page 60 and "Partner Node Name" on page 61
SNA_Symbolic_Destination	Name of the Side Information Profile defined for an SNA APPC connection to the partner system	A character string of <i>bbbbbbbb</i>	"Symbolic Destination" on page 61
SNA_TP_Name	Name of the Transaction Program	A character string of <i>bbbbbbbb</i>	"Transaction Program Name" on page 61
TCP/IP_Host_Name	Host name, if TCP/IP is used	"xx-64x-xx"	"Host Name" on page 61
TCP/IP_Port_Number	Port number in partner host, if TCP/IP is used	A number of 0 to 65535	"Port Number" on page 61
User_ID	User ID on the partner node	A character string of <i>xxxxxxx</i>	"User ID" on page 61
Pswd_Encrypt_Method	Partner Password Encryption Method	In MERVA USE & Branch for Windows NT, you can specify only "basic"	"Password Encryption Method" on page 62
Gateway_Alternate	Alternate Gateway	A character string of <i>nnnnnnnn</i>	"Alternate" on page 62

Defining ASP Information: Keyword List

Application_Support	Header of data for Application Support Processes (ASP)	-	"Defining ASP Information" on page 62
ASP	Header of data for an Application Support Process	-	"Defining ASP Information" on page 62
Local_ASP_Name	Name of the local ASP	A character string of <i>uuuuuuuu</i>	"ASP Name" on page 63

Local_MTP_Name	Name of the local MTP	A character string of <i>UUUUUUUUU</i>	"MTP Name" on page 63
Partner_ASP_Name	Name of the partner ASP	A character string of <i>UUUUUUUUU</i>	"Partner ASP Name" on page 63
Partner_MTP_Name	Name of the partner MTP	A character string of <i>UUUUUUUUU</i>	"Partner MTP Name" on page 63
Partner_Node_Name	Name of the partner node	A character string of <i>UUUUUUUUU</i>	"Partner Node Name" on page 64
Application_User_Exit	Name of the application user exits	A character string of <i>AAAAAAAAA</i>	"Application user exit" on page 66
Security_User_Exit	Name of the security user exits	A character string of <i>AAAAAAAAA</i>	"Security user exit" on page 66
Security_Key_Modifier	Name of the security key modifiers	A character string of <i>AAAAAAAAA</i>	"Security key modifier" on page 66
Message_Audit_Logging	Specifies, if message audit logging is selected	"YES" or "NO"	"Message audit logging enabled" on page 66
Urgent_Send_Queue	Name of the Urgent queue	A character string of <i>UUUUUUUUU</i>	"Queues (Urgent/Normal/Low)" on page 64
Normal_Send_Queue	Name of the Normal queue	A character string of <i>UUUUUUUUU</i>	"Queues (Urgent/Normal/Low)" on page 64
Low_Send_Queue	Name of the Low queue	A character string of <i>UUUUUUUUU</i>	"Queues (Urgent/Normal/Low)" on page 64
Transfer_Format	The Transfer Format	One of the following: Line ASCII Queue Buffer Line EBCDIC	"Transfer Format" on page 65
Preferred_Connection	The Preferred Connection Type	One of the following: SNA APPC TCP/IP	"Preferred Connection Type" on page 65
Autostart	Specifies, if Autostart is selected	"YES" or "NO"	"Autostart" on page 64
Authentication	Specifies, if Authentication is selected	"YES" or "NO"	"Authentication" on page 64
Encryption	Specifies, if Encryption is selected	"YES" or "NO"	"Encryption" on page 64
Receive_Queue	Name of the Receive Queue	A character string of <i>UUUUUUUUU</i>	"Receive Queue" on page 65
ACK_Wait_Queue	Name of the (Receive) Acknowledgment Wait Queue	A character string of <i>UUUUUUUUU</i>	"Ack Wait Queue" on page 66
State_of_Receiving_Process	Specifies the status of the receiving process	One of the following: Enabled Disabled	"Operating the MERV Link" in the <i>MERVA USE & Branch for Windows NT User's Guide</i>
Description	Description	A string of up to 60 characters	"Description" on page 63
User_Exit_Flag_1	Specifies, if Application User exit flag 1 is selected	"YES" or "NO"	"Application user exit flag (1 and 2)" on page 66

User_Exit_Flag_2	Specifies, if Application User exit flag 2 is selected	"YES" or "NO"	"Application user exit flag (1 and 2)" on page 66
------------------	---	---------------	---

A sample definition of customization data in the ASCII file of the MERVA Link component is:

```

MERVA_Link = (
  Local_Node = (
    Nodename = "LN1"
  )
  Partner_Nodes = (
    Partner_Node = (
      Nodename = "P1"
      SNA_Symbolic_Destination = "SymDest1"
      User_ID = "UserID_1"
      TCP/IP_Host_Name = "Host_Name_1"
      TCP/IP_Port_Number = "11111"
    )
    ...
  )
; End (Nodename=P1)
; End Partner_Nodes
  Application_Support = (
    ASP = (
      Local_ASP_Name = "A1"
      Local_MTP_Name = "M1"
      Partner_ASP_Name = "PA1"
      Partner_MTP_Name = "MP1"
      Partner_Node_Name = "P1"
      Urgent_Send_Queue = "S_11"
      Normal_Send_Queue = "S_12"
      Low_Send_Queue = ""
      ACK_Wait_Queue = ""
      Receive_Queue = "MLRECEIV"
      Preferred_Connection = "TCP/IP"
      User_Exit_Flag_1 = "YES"
      User_Exit_Flag_2 = "NO"
      Transfer_Format = "Queue Buffer"
      Encryption = "NO"
      Autostart = "YES"
      State_of_Receiving_Process = "Disabled"
      Authentication = "NO"
      Description = "1st ASP"
    )
; End A1
    ...
; End Application_Support
  )
; End MERVA_Link;

```


SWIFT Link Component: Keyword List

Table 14 lists the keywords allowed for customizing data in the ASCII file (*filename.SLI*) of the SWIFT Link component:

Table 14. Keyword List and Description for SWIFT Link

Keyword	Meaning	Values	Window Dialog Reference
SWIFT_Link	Header of the structure for all SWIFT Link data	-	"Chapter 6. Customizing the SWIFT Link" on page 47
Logical_Terminals	Header of the structure for all Logical Terminals	-	"Chapter 6. Customizing the SWIFT Link" on page 47
LT_i	Header of the structure for a Logical Terminal	i can be 1 to 16	"Defining Logical Terminal (LT) Identifiers" on page 47
LT_Name	Name of the Logical Terminal	A character string of GGGGGUUU	"Defining Logical Terminal (LT) Identifiers" on page 47
Synonym_i	Synonym name of the Logical Terminal	A character string of GGGGGUUU	"Defining Synonym LTs" on page 48

Defining Queue Assignments for Logical Terminals: Keyword List

Assigned_GPA_queues	Header of the structure for the assigned GPA queues	-	"Defining Queue Assignments for Logical Terminals" on page 48
Assigned_FIN_queues	Header of the structure for the assigned FIN queues	-	"Defining Queue Assignments for Logical Terminals" on page 48
Send_Immediate	Name of the send immediate queue for FIN or GPA	A character string of uuuuuuuu	"Send Immediate" on page 48
Send_Normal	Name of the send normal queue for FIN or GPA	A character string of uuuuuuuu	"Send Normal" on page 48
Send_System	Name of the send system queue for FIN or GPA	A character string of uuuuuuuu	"Send System" on page 48
Send_Urgent	Name of the send urgent queue for FIN or GPA	A character string of uuuuuuuu	"Send Urgent" on page 48
Receive	Name of the receive queue for FIN or GPA	A character string of uuuuuuuu	"Receive" on page 48

Defining the Network Data for SWIFT Link: Keyword List

Network_Data	Header of the structure for network data	-	"Defining the Network Data for SWIFT Link" on page 49
Line_Type	Specifies the line type	One of the following: leased_line shared_PSTN dedicated_PSTN PSPDN	"Line Type" on page 50

Dial_Type	Specifies the dial type	One of the following: automatic manual	"Dial Type" on page 49
Symbolic_Address	The BIC part of the symbolic address	A character string of <i>GGGGUUUU</i>	"Line Details" on page 49
Connection_ID	The ID part of the symbolic address	A character string of 99	"Line Details" on page 49
Local_Phone_Number	Local phone number	A character string of 99999999999999	"Line Details" on page 49
Local_DTE	Local DTE	A character string of 99999999999999	"Line Details" on page 49
Remote_DTE	Remote DTE	A character string of 99999999999999	"Line Details" on page 49
Suspend_Desired	Specifies, if suspend is desired	"YES" or "NO"	"Line Details" on page 49
Link_Name	Link name	A character string of <i>bbbbbb</i>	"Line Details" on page 49
Ack_Timeout_Period	Specifies the size of the acknowledgment timeout	A number of 0 to 9999	"Time-outs" on page 50
Ass_Timeout_Period	Specifies the size of the association timeout	A number of 0 to 9999	"Time-outs" on page 50
TPDU_Size	Specifies the size of the Protocol Data Unit	Multiples of 128; maximum: 8192 (i.e. 128, 256, ..., 8192)	"PDU Size" on page 49
Retries	Maximum number of retry attempts	A number of 0 to 40	"Resynchronization" on page 50
Delay	Delay (in seconds) before retry is started	A number of 20 to 300	"Resynchronization" on page 50
Duration	Maximum duration of recovery (in minutes)	A number of 0 to 60	"Resynchronization" on page 50

Defining the FIN Copy Service: Keyword List

FIN_Copy_Data	Header of the structure for FIN copy data	-	"Defining the FIN Copy Service" on page 51
Copy_Service_i	Header of the structure for a copy service	-	"Defining the FIN Copy Service" on page 51
Name	Copy service identifier	A character string of <i>XXX</i>	"Copy Service Identifier" on page 53
Activation_Date	Activation date	A character string of 99999999 (<i>YYYYMMDD</i>)	"Definition's Activation" on page 53
Activation_Time	Activation time	A character string of 9999 (<i>HHMM</i>)	"Definition's Activation" on page 53
Central_Institution	BIC code of the central institution destination (CID)	A character string of <i>GGGGUU</i>	"Central Institution Destination" on page 53
Live_Service_Flag	Live service flag	Live or T&T	"Live Service Flag" on page 54
Authentication_Mode	Authentication mode	Normal or Double	"Authentication Mode" on page 54

Full_Copy_Flag	Full copy flag	Full or Partial	"Full Copy Flag" on page 54
Date_Range	Value date range	A character string of 99 (0 to 99)	"Value Date Range" on page 54
Currency_Code	Currency code	A character string of XXX	"Currency Code" on page 54
Add_103	Specifies if field 103 is added automatically	"YES" or "NO"	"Add field 103 automatically" on page 54

Specifying Message Fields: Keyword List

Message_Type_i	Header of the structure for selected message types	-	Figure 27 on page 55
Name	Name of the selected message type	A character string of XXX	"Message Types" on page 54
Value_Date	SWIFT tag identifying the value date field	A character string of GGG	Figure 27 on page 55
Currency	SWIFT tag identifying the currency code field	A character string of XXX	"Currency Code" on page 54
CID	SWIFT tag identifying the CID field	A character string of XXX	"Central Institution Destination" on page 53
included_MSG_Field_i	SWIFT tags identifying additional message fields to be included	A character string of XXX	Figure 27 on page 55

A sample definition of customization data in the ASCII file of the SWIFT Link component is:

```

SWIFT_Link = (
  Logical_Terminals = (
    LT_1 = (
      LT_Name = "MASTER001"
      Assigned_GPA_queues = (
        Send_Immediate = "--None--"
        Send_System = "GPA_S_1"
        Send_Urgent = "GPA_U_1"
        Send_Normal = "GPA_N_1"
        Receive = "GPA_R_1"
      )
      Assigned_FIN_queues = (
        ...
      )
    )
    ...
  )
;
Network_Data = (
  Line_Type = "PSPDN"
  Dial_Type = "manual"
  Symbolic_Address = "HUG00000"
  Connection_ID = "99"
  Local_Phone_Number = ""
  Local_DTE = "1111111111111111"
  Remote_DTE = "2222222222222222"
  Suspend_Desired = "NO"
  Link_Name = "RECHTERN"
  Ack_Timeout_Period = "9"
  Ass_Timeout_Period = "6"

```

```

        TPDU_Size = "128"
        Retries = "3"
        Delay = "40"
        Duration = "10"
    )
;   End Network Data
    FIN_Copy_Data = (
        Copy_Service_1 = (
            Name = "BBB"
            Activation_Date = "19991231"
            Activation_Time = "2130"
            Central_Institution = "CIDXXXXX"
            Live_Service_Flag = "Live"
            Authentication_Mode = "Normal"
            Full_Copy_Flag = "Partial"
            Date_Range = "99"
            Currency_Code = ""
            Add_103 = "YES"
            Message_Type_1 = (
                Name = "105"
                Value_Date = ""
                Currency = "32A"
                CID = ""
                Included_MSG_Field_1 = "454"
            )
            ...
        )
        ...
    )
;   End FIN Copy Data
;   End SWIFT Link
)

```

User Access Rights: Keyword List

Table 15 lists the keywords allowed for customizing data in the ASCII file (*filename.USR*) of user access rights.

Note: The window dialog for user access rights is described in the *MERVA USE & Branch for Windows NT User's Guide*.

Table 15. Keyword List and Description for User Access Rights

Keyword	Meaning	Values	Window Dialog Reference
User_Rights	Header of the structure for defining all user access rights	-	<i>MERVA USE & Branch for Windows NT User's Guide</i>
All_Access_Rights	Header of the structure for all rights (in general)	Note: Definitions contained in this part must not be changed.	<i>MERVA USE & Branch for Windows NT User's Guide</i>
Right_nnnn	Internal definition of access rights (specified as Right_nnnn) for the export or import commands	Note: You must not change the definitions.	Available Access Rights on the Access Rights - Update window
;Associated Queue	Comment line for specifying the name of the access rights assigned to a queue	Comment line	Details on the Access Rights - Update window

Updating User Access Rights: Keyword List

All_Users	Header of the structure for defining all access rights (user-specific)	-	<i>MERVA USE & Branch for Windows NT User's Guide</i>
User	Header of the structure for a user	-	The User Administration - List Users window
UserID	Name of the users in MERVA	A character string as allowed by Windows NT	The User Administration - List Users window
Assigned_LT	Logical Terminal of the user	A character string of <i>GGGGGIIUUU</i>	The Access Rights - Update window
Granted_Right	Access right assigned to the user	A character string of <i>Right_nnnn</i>	Current Access Rights on the Access Rights - Update window

Restricting Access Rights to Queues: Keyword List

Details	Header of the structure for assigned queues	-	<i>MERVA USE & Branch for Windows NT User's Guide</i>
Queue	Name of the queue that is assigned to the access right	A character string of <i>uuuuuuuu</i>	Details on the Access Rights - Update window

Restricting Access Rights to Create Specific Message Types: Keyword List

Message_Types	Header of the structure for assigned message types	-	<i>MERVA USE & Branch for Windows NT User's Guide</i>
Application	Header of the structure for an application	-	Applications on the Access Rights - Details window

Name	The short name of the application the messages belong to	A character string of eAAAAAAAA	Applications on the Access Rights - Details window
Category	Header of the structure for the categories	-	Categories on the Access Rights - Details window
Name	The short name of the category the messages belong to	A character string of eAAAAAAAA	Categories on the Access Rights - Details window
Message_Type	Name of the message type the user can access	Any character string	Message Types on the Access Rights - Details window
Filter	Specifies message groups for the user, instead of explicit message types defined in Message_Type	Any character string	Filter on the Access Rights - Details window

A sample definition of customization data in the ASCII file of the user access rights definitions is:

```

User_Rights = (
    All_Access_Rights = (
        Right_0001 = "API - Without password"
    ;
        Associated Queue = SLRCVFIN    (only as comment)
        ...
        Right_0002 = "API - With password"
        ...
    )
    All_Users = (
        User = (
            UserID = "m3"
            Assigned_LT = "IBMMERVAX"
            Granted_Right = "Right_0001"
            Details = (
                Queue = "SL_IN"
                ...
            )
        ;
            End Queue Details
        )
    ;
        End User m3
    )
    ...
;
End All Users
)
;
End User Rights;

```

The SWIFT USE Component: Keyword List

Table 16 lists the keywords allowed for customizing data in the ASCII file (*filename.USE*) of the SWIFT USE component.

Table 16. Keyword List and Description for the SWIFT USE Component

Keyword	Meaning	Values	Window Dialog Reference
SWIFT_USE	Header of the structure for defining all SWIFT USE data	-	"Chapter 8. Customizing SWIFT USE" on page 75
Message_Headers	Header of the structure for defining data of all SWIFT Message Headers	-	"Defining Message Headers" on page 75
Destination	Header of the structure for a destination	-	"Defining Message Headers" on page 75
Name	Name of the destination	A character string of <i>GGGGGGUU</i>	"Defining Message Headers" on page 75
Emitting_LT	Emitting Logical Terminal	A character string of <i>GGGGGGUUUUUU</i>	"Defining Message Headers" on page 75
Priority	Priority	"Normal" or "Urgent"	"Defining Message Headers" on page 75
Deliv_Notification	Specifies, if delivery notification is selected	"YES" or "NO"	"Defining Message Headers" on page 75
Obsolesc_Period	Obsolescence period	A number of 2 to 999.	"Defining Message Headers" on page 75

Background Processes: Keyword List

Processes	Header of the structure for defining all data of the BKE Background Process	-	"Understanding the USE Background Process" on page 76
Received_Messages	Name of the queue from which the BKE Background Process reads SWIFT USE messages	A character string of <i>uuuuuuuuu</i>	"Understanding the USE Background Process" on page 76
NAKed	Name of the queue for messages that have been negatively acknowledged by the SWIFT network	A character string of <i>uuuuuuuuu</i>	"Understanding the USE Background Process" on page 76
Received_Commands	Name of the queue for internal BKE commands	A character string of <i>uuuuuuuuu</i>	"Understanding the USE Background Process" on page 76

Further SWIFT USE Definitions: Keyword List

Definitions	Header of the structure for defining all data of the USE definitions	-	"Defining Timeout" on page 77
Timeout_for_Cert	Timeout for certification request (in days)	A number of 1 to 7	"Defining Timeout" on page 77
Timeout_for_BKE	Timeout for a BKE message response (in days)	A number of 1 to 14	"Defining Timeout" on page 77

Cert_exp_after	Expiry date for a certificate (in months)	A number of 1 to 24	"Defining Timeout" on page 77
BKE_Start	Specifies the number of days a BKE is initiated prior to the planned effective date of a new future key	A number of 1 to 999.	"Defining BKE Start Conditions" on page 78
Key_certification	Key certification	A character string of AAAAAAAAAAAAAA	Figure 32 on page 75
Card_Management	Card management	A character string of AAAAAAAAAAAAAA	Figure 32 on page 75
Support_Center	Support center	A character string of AAAAAAAAAAAAAA	Figure 32 on page 75

A sample definition of customization data in the ASCII file of the SWIFT USE component is:

```

SWIFT_USE = (
;
    Message_Headers = (
        Destination = (
            Name = "IBMMERVA"
            Emitting_LT = "IBMMABCDEXXX"
            Priority = "Normal"
            Deliv_Notification = "YES"
            Obsolesc_Period = "003"
        )
    )
    Processes = (
        Received_Messages = "USERCV02"
        NAKed = "UNNAKED"
        Received_Commands = "INCOMMCO"
    )
    Definitions = (
        Timeout_for_Cert = "5"
        Timeout_for_BKE = "14"
        Cert_exp_after = "12"
        BKE_Start = "30"
        Key_certification = "SWHQBECAACKC"
        Card_Management = "SWHQBECAACMC"
        Support_Center = "SWHQUSUSAXXX"
    )
;
)
; End SWIFT USE ;

```

Part 3. Establishing Connections in MERVA

Chapter 11. Customizing TCP/IP for MERVA Applications

TCP/IP is a communication method that is mostly used in a network of interconnected Windows NT systems. A set of hosts that is interconnected by TCP/IP is called an **internet**. MERVA components, such as MERVA Link or MERVA Message Processing Client for Windows NT, can execute in the hosts of an internet and use TCP/IP services for communication.

This chapter describes all aspects related to the TCP/IP support in MERVA.

TCP/IP Services

This section explains the subset of TCP/IP services that is used by MERVA.

TCP/IP Sockets Interface

The TCP/IP Sockets Interface is an Application Program Interface (API) to request TCP/IP services. A TCP/IP socket is the connection point of an application program to TCP/IP communication services.

The MERVA Link socket application, for example, is based on the connection-oriented client-server model using TCP/IP as its communication protocol. In this model, first a server, then a client is started. The client connects to the server and sends data to the server.

TCP/IP Host Name and Port Number

A TCP/IP socket program that starts a conversation specifies the location of its partner Transaction Program (TP) via partner host name or TCP/IP address. The TP in the partner system is identified by its TCP/IP port number.

Host Name

A partner TCP/IP system is identified in MERVA by its TCP/IP host name. The host names used by MERVA should be defined in the host table (%SystemRoot%\system32\drivers\etc\hosts) of a sending Windows NT system. A TCP/IP name server can be called automatically if a host name is not defined in the host table.

An entry of the host table maps the name of a partner host to its internet address and the transfer protocol, such as TCP.

Port Number

MERVA supports a set of TCP/IP port numbers for its inbound TPs. Each MERVA instance has a port number. The port number specifies the MERVA instance in a Windows NT host to which the TP must connect.

MERVA Inetd Service

The internet service **Inetd** is a Windows NT service that can handle several connection requests.

To configure the MERVA Inetd service:

1. Click **Start** → **Settings** → **Control Panel**.
2. Click **Services**.

3. Select **MERVA-Inetd Daemon**.
You then get the Service window.
4. Click **Startup**.
5. In the field **Startup Type**, select **Automatic**.
6. From the list **Log On As**, select **This Account**.
7. In the field **This Account**, type the user ID of your MERVA system administrator, for example, **mervaadm**.
8. In the field **Password**, type the correct password.
9. Click **OK**, then close the Service window.

To use the MERVA Inetd service for a specific server application has the following advantages:

- The application server does not have to run permanently. It does not add another process to the permanently active processes in a Windows NT system.
- It makes the writing of the program that handles inbound requests easier because MERVA Inetd handles many startup details.

The configuration file **enminetd.cfg** determines the services that MERVA Inetd is to use. To get the sample configuration file:

1. Click **Start** → **Programs** → **Command Prompt**.
2. Type the following command in the command line:
`copy %ENM_PATH%\samples\InetD\enminetd.cfg %SystemRoot%\system32\drivers\etc\`
3. Close the Command Prompt window.

The sample configuration file includes the following services:

- MERVA Message Processing Client for Windows NT
- MERVA Link
- Remote API connection
- Remote Card Reader

Customizing TCP/IP Services

TCP/IP customization parameters that are related to MERVA component parameters must be provided in the following places:

- TCP/IP Hosts Table (**%SystemRoot%\system32\drivers\etc\hosts**)
Note that you do not have to provide hosts parameters if you use a TCP/IP name server.
- TCP/IP Client Network Services
(**%SystemRoot%\system32\drivers\etc\services**)
- MERVA Inetd services (**%SystemRoot%\system32\drivers\etc\enminetd.cfg**)

Edit the corresponding system files and add the corresponding parameter entries.

Customizing Hosts Table

The host name of every partner host that a MERVA component must use as a partner system has to be defined in the TCP/IP network. Usually, a name server that can map a partner host name to the applicable TCP/IP address is used. If a name server is not available, or if you do not want to use the services of a name server for MERVA partner hosts, you must define the partner hosts in the Hosts

Table of your Windows NT system. The name of the Hosts Table is **%SystemRoot%\system32\drivers\etc\hosts**.

Add a text line to this file. The syntax of each text line is:

```
ip_address      name
```

The following line shows you an example for a text line.

```
1.123.1.235     mervas
```

Customizing Client Network Services

To customize the Client Network Services, add a text line to the file **%SystemRoot%\system32\drivers\etc\services**. The syntax of each text line is:

```
symbolic_subservice_name  Portnumber/TCP_protocol
```

The following line shows you an example for a text line.

```
subservice1      7110/tcp
```

Note: To avoid duplicate port numbers, ensure that the lines are sorted by port numbers.

Customizing MERVA Inetd Subservices

To customize the Superserver Services, add a text line to the file **%SystemRoot%\system32\drivers\etc\eninetd.cfg**. The syntax of each text line is:

```
S_S_N  S_T  T_P  W_F_S  U_N  P_P  P_A
```

The following list shows the meaning of the abbreviations.

S_S_N	Symbolic Subservice Name
S_T	Socket Type
T_P	Transport Protocol
W_F_S	Wait for Server to Release Socket
U_N	User Name
P_P	Program Path
P_A	Program Arguments

The following line shows you an example for a text line.

```
subservice1  stream tcp  nowait merval C:\merva\bin\ekatci arg1 arg2
```

Chapter 12. MERVA Link

Customizing TCP/IP Services for MERVA Link

You have to provide TCP/IP customization parameters that are related to MERVA Link in the following places:

- TCP/IP Hosts Table
- TCP/IP Client Network Services
- MERVA Inetd Service

For a detailed description of these items, refer to “TCP/IP Services” on page 113.

Customizing Hosts Table

To customize the hosts table:

1. Change to the directory `%SystemRoot%\system32\drivers\etc`.
2. Edit the file `hosts` by adding a text line, such as
`1.123.2.234 mervas`

where `1.123.2.234` is the IP address of the MERVA Link partner host `mervas`.

Note: You can use the local host entry defined in any hosts table as a valid partner host name in a MERVA Link sending process. If you specify `localhost` as partner host name, the local receiving TP (`ekatci`), receives the outbound message immediately.

The IP address of `localhost` is `127.0.0.1`. The TCP/IP architecture defines it as an address that identifies always the local host.

Customizing Client Network Services

To customize the client network services:

1. Change to the directory `%SystemRoot%\system32\drivers\etc`.
2. Edit the file `services` by adding a text line, such as
`ekaras 7110/tcp`

where:

- `ekaras` is the symbolic service name for the MERVA Link message transfer server.
 - `7110/tcp` defines the IP port number 7110 for the connection and specifies that the TCP protocol is required.
3. Ensure that the last line in the file `services` is empty.
 4. Close the file.

Customizing MERVA Inetd Service

To customize the MERVA Inetd Service:

1. Change to the directory `%SystemRoot%\system32\drivers\etc`.
2. Edit the configuration file `enminetd.cfg` by adding a text line, such as
`ekaras stream tcp nowait root C:\MERVA\USE_Branch\bin\ekatci merval`

where **merva1** is the name of the MERVA instance.

TCP/IP Conversation Security

Cooperative processing allows application programs to establish communications with partner programs on other systems. It also allows to share work, data, and services between systems and across networks. This makes it necessary that installations using cooperative processing follow specific security requirements.

In a client-server environment, the server sets the security requirements for a conversation with a specific client or for all clients. A client must follow the security requirements and must provide the applicable access security information.

MERVA Link can be considered as a client when it establishes a conversation to a partner system to send messages to a partner application. MERVA Link can be considered as a server when it accepts an inbound conversation from a partner system to receive messages and to pass the messages to a receiving application.

MERVA Link Outbound Conversation Security Information

TCP/IP Conversation Security Information is a Conversation Security User ID (CSU) and a Conversation Security User Password (CSP). The CSU identifies a user in a server system. The CSU and the CSP must be provided by a client application when it requests the services of a server application.

TCP/IP does not provide facilities to manage and to transmit conversation security information. Therefore, MERVA Link provides functions to handle conversation security information, and transmits this information as part of its control data.

Specify Conversation Security Information

Conversation-security information is part of the MERVA Link Intersystem Connection (ISC) information that describes the connection to a partner system (partner MERVA Link node). The MERVA Customizer is used to specify the parameters of a MERVA Link connection to a partner MERVA Link node. This information is saved in the MERVA customization database.

The conversation-security user password is encrypted before it is stored in the MERVA customization database.

Modify Conversation Security Information

The conversation-security information, especially the password, can require a modification more often than the other customization parameters. Therefore, the MERVA Link operating function allows you to change a conversation-security user ID and a password easily. The changed conversation-security information comes immediately into effect. You do not have to restart MERVA.

Encrypting the Conversation Security Password

Before a conversation security password is stored in the MERVA customization database, it is encrypted with a MERVA proprietary algorithm.

Conversation Security Information in the ACT

The MERVA Link customization parameters that are used by MERVA Link processes are contained in the MERVA Link ACT. When the ACT is generated, it is initialized with information from the MERVA customization database. When a conversation security password must be copied from the customization database to the ACT, it is first decrypted with the appropriate functions. It is then encrypted again and stored in the ACT.

The encrypted forms of a password in the customization database and in the ACT are always different.

The base vector that is used to generate the encryption key is unique for every ACT password encryption process. The encrypted passwords in the ACT are changed after a MERVA startup and after a refresh of the ACT from the MERVA customization database.

MERVA Link Outbound Conversation Security

A MERVA Link sending process based on TCP/IP services must provide a client user ID and a password. This is a requirement of the MERVA Link receiving process (server). Using conversation security is mandatory in MERVA Link conversations based on TCP/IP services.

MERVA Link Inbound Conversation Security Information

A MERVA Link receiving process based on TCP/IP services expects conversation security information in the PROBE that is the first PDU received in a MERVA Link conversation. The mandatory conversation security information data element contains the following mandatory data elements:

- Client user ID
- Client user password (encrypted)
- Encryption control information
- Encryption method identifier

The MERVA Link inbound TP supporting TCP/IP (EKATCI) rejects a conversation if any of the conversation security information data elements is missing from the inbound PROBE.

MERVA Link Inbound Conversation Security

After the MERVA Link inbound TP supporting TCP/IP (EKATCI) finds all mandatory conversation security information in the PROBE envelope, it takes the following actions to ensure conversation security.

1. The encryption control information is used to decrypt the client user password as specified by the applicable encryption method.
2. Client user information is retrieved from the Windows NT user database. If the specified user ID is not defined in Windows NT, the received client user ID is translated to lower case characters. If that user ID is also not defined in the Windows NT user database, the inbound conversation is rejected.
3. The received client user password is translated to the Windows NT user database format and compared with the password in the Windows NT user database. If the passwords do not match, the received client user password is translated to lower case characters. If that password is also not correct, the inbound conversation is rejected.

To ensure confidentiality of user passwords, the MERVA Link password encryption rules are considered as confidential. Plain text passwords are available in Windows NT main storage only as long as they are needed by MERVA Link programs. A plain text password is erased in the main storage as soon as possible.

Customizing a Windows NT Communications Server for MERVA Link

MERVA Link supports the interconnection of MERVA systems within an SNA network using the LU 6.2 services of the IBM program products Communications Server or Personal Communication.

A configuration profile must be defined in the Communications Server or Personal Communication to connect a MERVA Link node to a MERVA Link SNA APPC network. First, your PC must be physically connected to the SNA network (via a token ring attachment, for example), and a logical unit (LU 6.2) must be defined in the SNA control point that supports the Communications Server.

A sample set of VTAM[®] and Windows NT SNA definitions is shown in the following. The samples apply to the IBM Communications Server for Windows NT Version 5.0.1. For more detailed information, refer to the *Communications Server NT Tutorial*.

VTAM APPC LU Definition

If you are connecting the PC to an SNA host (PU type 4 or 5), the PC is defined as a physical unit (PU) in VTAM. The type of the physical unit is PU 2.1. It supports independent LUs.

A PU can host a number of logical units (LUs). One of these LUs, the APPC LU, is regarded in this example. It is an independent LU that supports parallel sessions (LOCADDR=0).

The following sample shows a VTAM PU and LU 6.2 definition sample:

```
*
FD070C  PU  ADDR=01,ANS=CONT,MAXDATA=265,MAXOUT=7,PASSLIM=7,PUTYPE=2,  *
          IDBLK=071,DISCNT=NO,ISTATUS=ACTIVE,IDNUM=FE00C,PACING=7,*
          SSCPFM=USSSCS
FD070C0A LU  LOCADDR=0,DLOGMOD=LU62,MODETAB=MTX7LU6,USSTAB=USSSNA
*
```

The values of the IDBLK and IDNUM parameters of the PU definition are important parameters in the SNA Server Control Point profile (node_cp). The LU name (the label of the LU macro) is also referred to in follow on profiles.

The name of the network that houses the PU is also an important parameter in the follow on profiles. It is specified in the NETID parameter in the VTAM start statement.

Node Configuration

This section describes how to configure the Communications Server to support MERVA Link.

To define a new configuration:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **New** and **CPI-C, APPC, or 5250 Emulation**.

You then get the Define the Node window.

The following figure shows the Define the Node window with sample values.

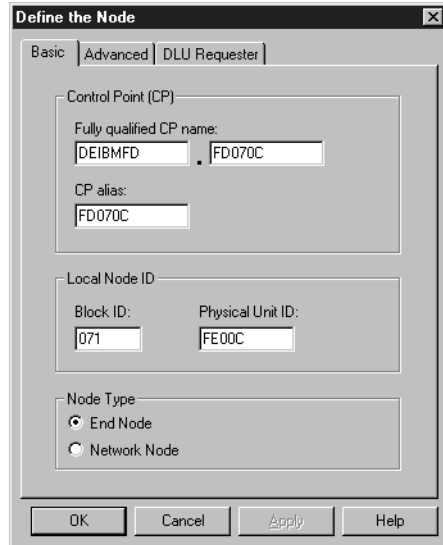


Figure 43. The Define the Node Window

The fully qualified unique Control Point (CP) name identifies the node within the network environment. The first part is the name of the network, the second part is the CP name. The sample uses the local PU name as CP name as shown in the example on page 120.

The **Local Node ID** is the concatenation of the values specified for the IDBLK and IDNUM parameters in the PU definition as shown in the example on page 120.

The **Node Type** must be **end_node** because APPN[®] services are not used in this example.

The defaults are used for the remaining parameters of the Define the Node window.

Device Configuration

To define a supported communication device on the PC:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **LAN** as **DLC** type, then select **New**.

You then get the Define a LAN Device window.

The following figure shows an example of this window.

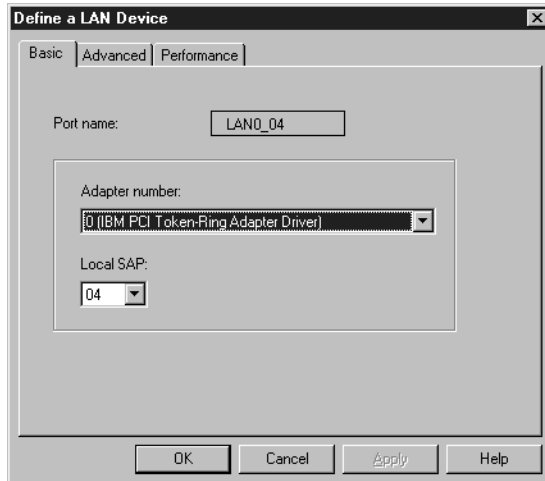


Figure 44. The Define a LAN Device Window

In the **Adapter number** field, specify the adapter number. If an adapter number is not shown by default, the IBM LLC2 LDC interface was not installed during Communications Server installation.

Connection Configuration

To define links to other nodes in the SNA network:

1. Click **Start** → **Programs** → **IBM Communications Server** → **Configure Connections**.
2. Select **LAN** as **DLC** type, then select **New**.

You then get the Define a LAN Connection window.

The following figure shows an example of this window.



Figure 45. The Define a LAN Connection Window

In the **Destination address** field, you specify the address of the remote node to which the connection is to be established. Ask your SNA network administrator for the address to be specified.

To activate the link:

1. Click **Advanced**.
2. Select **Activate link at start**.

LU 6.2 Sessions

The setup of LU 6.2 sessions is configured in a Local LU profile, in Side Information profiles, in Partner LU profiles, in Mode profiles, in a Transaction Program Name profile, and in a Partner LU 6.2 Location profile. Each profile must be contained in the sample configuration. You can also use Conversation Security profiles (Conversation Security Access List and Resource Security Access List).

The Session Security, Session Timeout, and the Class of Service (COS) profiles provided by Communications Server are not modified for MERVA Link purposes.

Local LU 6.2 Configuration

You have to define a local LU. The local LU handles the local side of the APPC session for the transaction program.

To define a local LU:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **Configure Local LU 6.2**, then select **New**.

You then get the Define a Local LU 6.2 window. The following figure shows an example of this window.

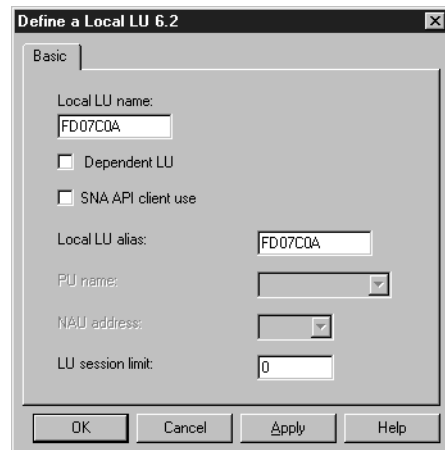


Figure 46. The Define a Local LU 6.2 Window

The **Local LU name** is the name of the APPC LU as defined in the label of the LU macro that describes the LU in VTAM. For details refer to the example shown on page 120.

Define the local LU as an independent LU. To do this, select **No** in the field **Local LU is dependent?**.

You can choose any name for the Local LU alias. The default LU alias is the LU name.

Partner LU 6.2 Configuration

The name of the partner LU is the name of the LU in which the partner program is located. The local LU recognizes the name of the remote LU to allocate a conversation.

To define a partner LU:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **New** and **Configure Partner LU 6.2**.

You then get the Define a Partner LU 6.2 window. The following figure shows a sample LU 6.2 partner LU definition that defines a partner CICS/ESA[®] system.

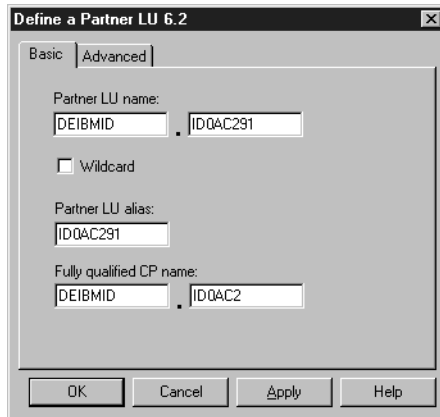


Figure 47. The Define a Partner LU 6.2 Window

Specify the fully qualified partner LU name to which you want to connect. The fully qualified name is the network name and the LU name. You also have to specify the fully qualified control point name.

You can choose any name for the partner LU alias. The default partner LU alias is the partner LU name.

LU 6.2 Mode Configuration

To use an SNA logon mode for MERVA Link sessions to different partner systems, you have to define it.

To define a LU 6.2 Mode:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **New** and **Configure Modes**.

You then get the Define a Mode window. The following figure shows a sample LU 6.2 Mode (APPCLU62) definition.

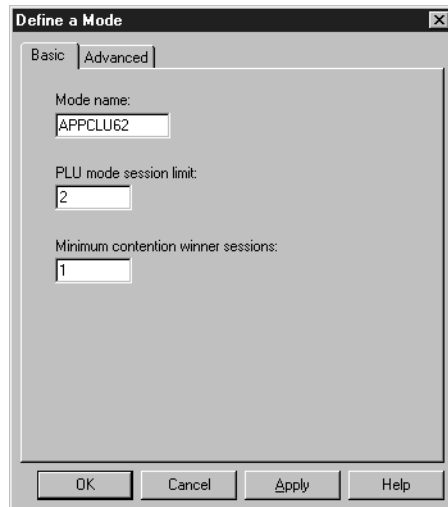


Figure 48. The Define a Mode Window

Two sessions are sufficient for MERVA Link operations. Each system, the local system and the partner system, gets a session of its own as a contention-winner session. Each system can start a conversation without asking for permission from the partner system.

MERVA Link does not need a specific RU size. You can specify any valid number. Note that this number affects the resources of the local and the partner system. The sample maximum RU size of 16384 ensures the highest possible message transfer rate. Depending on your system resources, you might specify a lower number.

The RU size that actually applies to a LU 6.2 session is negotiated between the partner systems at the time when the session is bound. It depends on the RU sizes that are specified in the log mode table of the partner SNA node. It also depends on the mechanism that binds the LU 6.2 session.

CPI-C Side Information Configuration

You have to define the receiving processes in the partner LUs of MERVA Link in a CPI-C Side Information. The definition of a Side Information is called *symbolic destination name*.

To define a CPI-C Side Information:

1. Click **Start** → **Programs** → **IBM Communications Server** → **SNA Node Configuration**.
2. Select **New** and **Configure CPI-C Side Information**.

You then get the Define CPI-C Side Information window. The following figure shows a sample configuration of a LU 6.2 Side Information.

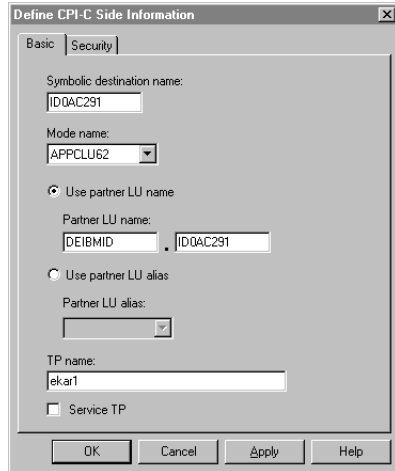


Figure 49. The Define CPI-C Side Information Window

The sample LU 6.2 Side Information configuration defines the symbolic destination ID0AC291. The symbolic destination is the MERVA Link receiving process in a partner CICS/ESA system. The sample destination name matches the LU name of the partner CICS/ESA system.

The **Partner LU alias** must specify the Partner LU alias defined in the Partner LU definition. For information of the Partner LU definition refer to “Partner LU 6.2 Configuration” on page 124. This sample specifies the Partner LU alias but not the fully qualified Partner LU name.

The **Mode name** specifies the VTAM logon mode that is to be used for the LU sessions to the partner system. The mode is defined in the SNA configuration and must be defined in the partner SNA node. For information on the Mode configuration refer to “LU 6.2 Mode Configuration” on page 124.

The **TP name** identifies the MERVA Link receiving transaction in the partner system. It is specified by the partner MERVA Link installation.

The sample TP name in a MERVA Link Windows NT system is **ekar1**.

Transaction Program Configuration

You have to define a transaction program for the MERVA Link receiving program.

To define a LU 6.2 transaction program:

1. Click **Start → Programs → IBM Communications Server → SNA Node Configuration**.
2. Select **New** and **Configure Transaction Programs**.

You then get the Define a Transaction Program window. The following figure shows a sample LU 6.2 TP name definition (**ekar1**).

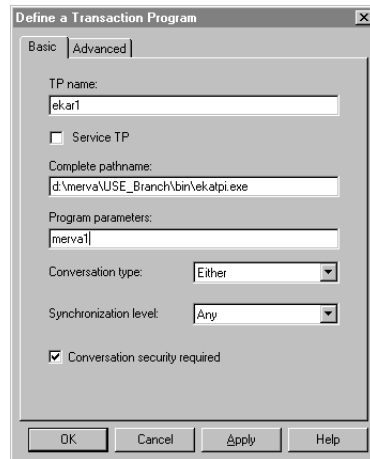


Figure 50. The Define a Transaction Program Window

The sample LU 6.2 TP definition for **ekar1** defines the MERVA Link receiving TP that is associated with the MERVA instance **merva1**. For each MERVA instance in a MERVA host, you have to generate a unique TP name profile for the MERVA Link receiving TP. Only the TP name and the program parameter name must be different because the MERVA instance is specified as a program parameter.

In the field **Complete pathname**, specify the Windows NT path of the MERVA Link receiving program **ekatpi.exe**. The program is located in the subdirectory **bin** of the MERVA installation directory, for example, **D:\merva\USE_Branch\bin**.

Note that the MERVA Link receiving TP supports only mapped conversations for which the synchronization level is set to **Confirm** or to **Any**.

The flag **Conversation security required** is selected by default. The following prerequisites apply:

- To start the TP, conversation security information is required. Incoming allocation requests for this TP without this information is rejected.
- The supplied user must be defined in the Windows NT system and in the LU 6.2 conversation security.

To define user IDs and passwords, use **Configure User ID Passwords**.

In the Advanced panel, ensure that the fields **Dynamically loaded**, **Full duplex support**, and **Background process** are checked.

LU 6.2 Conversation Security Profiles

A MERVA Link inbound conversation and a MERVA Link receiving TP can be protected by the Communications Server conversation security function. An inbound conversation is protected by selecting **Conversation security required** in the Define a Transaction Program window. Select this option if you want that the conversation security is checked when you create a transaction program definition.

To define conversation security for a TP, do the following in the SNA Node Configuration program:

On the server side

1. Select **Conversation security required** in the Define a Transaction Program window.

2. In the Configure User ID Passwords window, type the user ID and the corresponding password for each user.

On the client side

1. In the Partner LU 6.2 window, select **Advanced** → **Conversation security support**.
2. Provide the user ID and password in **Security** of the CPI-C side information or via the MERVA Link Partner Node Customization.
To change user ID and password during a session, select **ASP** → **Detailed Information** → **ISC Info** in the MERVA Link Operating window.

Automatically Starting Communications Server Resources

You can specify that the SNA Communication Server starts automatically when the Windows NT system is started. Use the **CSSTART** command to start the Communications Server with a specific configuration. This command is part of the **Command Line Utilities** for command line programs.

For more information about the parameters of the command line programs refer to the corresponding documentation.

SNA APPC Conversation Security

Cooperative processing allows application programs to establish communications with partner programs on other systems and to share work, data, and services between systems and across networks. This ability to access other programs and all the resources at their disposal poses special security considerations for installations that use cooperative processing.

In a client-server environment it is the server that sets the conversation security requirements for a conversation with a specific client (or for all clients). A client must be aware of the server's security requirements and must provide the applicable access security information.

MERVA Link NT can be considered as a client when it allocates a conversation to a partner system to send messages to a partner application. MERVA Link NT can be considered as a server when it accepts an inbound conversation from a partner system to receive messages and to pass the messages to a receiving application.

Conversation security for a MERVA Link client and for a MERVA Link server is discussed separately in the following sections.

In addition to conversation level security, session level security (also referred to as bind security) can be implemented when two APPC LUs are interconnected.

MERVA Link NT Outbound Conversation Security Information

APPC Conversation Security Information is a Conversation Security User ID (CSU) and an optional Conversation Security User Password (CSP). The CSU identifies a user in a server system. The CSU and the optional CSP must be provided by a client application when it requests the services of a server application.

Specify Conversation Security Information: Conversation security information is part of the MERVA Link NT Intersystem Connection (ISC) information that describes the SNA APPC connection to a partner system (partner MERVA Link

node). The MERVA Customizer is used to specify the parameters of a MERVA Link connection to a partner MERVA Link node. This information is saved in the MERVA Customization Data Base.

The optional conversation-security user-password is encrypted before it is stored in the Customization database.

Modify Conversation Security Information: The conversation-security information, especially the password, may require a modification more often than the other customization parameters. This is why the MERVA Link Operating function provides a means to change a conversation-security user ID and password easily. The changed conversation-security information becomes immediately effective. You need not restart MERVA to activate the changed conversation-security information.

Encrypting the Conversation Security Password: Before a conversation-security password is stored in the MERVA Customization database, it is encrypted. A MERVA proprietary algorithm is used to encrypt the password.

Conversation Security Information in the ACT: The MERVA Link customization parameters that are actually used by MERVA Link processes are contained in the MERVA Link ACT. When the ACT is generated, it is initialized with information from the MERVA Customization database. When a conversation security password must be copied from the Customization database to the ACT, it is first decrypted using the appropriate functions. It is then encrypted again and stored in the ACT.

The encrypted forms of a password in the Customization database and in the ACT are always different. The rules to generate the encryption key are different in the two encryption processes.

The base vector for generating the encryption key is unique for every ACT password-encryption process. The encrypted passwords in the ACT will be changed after a MERVA startup and after a refresh of the ACT from the MERVA Customization database.

MERVA Link NT Outbound Conversation Security

A MERVA Link sending process (client) must follow the conversation security rules that have been setup by the partner system for accessing a MERVA Link receiving process (server). Following the conversation-security rules means in this case to provide the applicable access-security information when the conversation to the partner system is allocated.

MERVA Link NT Outbound Security Types: The security type of a MERVA Link outbound allocate request can be NONE, SAME, PROGRAM, or STRONG. MERVA Link supports the security types NONE, SAME, and PROGRAM. The security information in the MERVA Link customization determines the security type used. Security information is the user ID, or the user ID and password.

Note: The security information of MERVA Link overwrites the security settings in the CPI-C side information window.

Security Type NONE: If a user ID is not part of the ISC parameters associated with a specific MERVA Link partner node, MERVA Link allocates a conversation to that partner with security type NONE. The Access Security Information in the SNA Attach Header (FMH 5) that is sent to the partner system will be empty. Depending on the security mechanisms implemented in the partner system, a

conversation with the partner process can be established or the allocate request is rejected because of insufficient access authority.

Security Type SAME: If a user ID is part of the ISC parameters associated with a specific MERVA Link partner node but a password is not specified, MERVA Link allocates a conversation to that partner with the security type SAME. Depending on the requirements of the partner system, the requested security type (SAME) can be downgraded to NONE. This is, however, not the case if the partner system is a Windows NT system. If you try to use the security type SAME in a Windows NT partner system, you get the parameter check **TS2A19**.

This type of error is not necessarily reported as a security violation. It may be reported as a permanent APPC resource failure. The corresponding MERVA Link diagnostic code can be **TS2E1A**.

If the requested security type (SAME) is accepted by the partner system, the Access Security Information in the SNA Attach Header (FMH 5) that is sent to the partner system will contain the user ID and the Already Verified (AV) indicator. This indicator shows that the user ID can be trusted and that it was verified on the outbound side. It is checked whether the user ID exists in the partner system. A conversation with the partner process is established only if the user is defined in the receiving system.

Any partner system of MERVA Link NT is advised not to accept conversation security type SAME from the Communications Server LU. A Windows NT system is not considered to be a trusted system, because a single user can have the full control of all user and security-administrator functions.

Security Type PROGRAM: If a user ID and a password are part of the ISC parameters associated with a specific MERVA Link partner node, MERVA Link allocates a conversation to that partner with security type PROGRAM. Depending on the requirements of the partner system, the requested security type (PROGRAM) can be downgraded to NONE.

If the requested security type (PROGRAM) is accepted by the partner system, the Access Security Information in the SNA Attach Header (FMH 5) that is sent to the partner system will contain the user ID and the password. The user ID and the password are checked in the partner system. A conversation with the partner process is established only if the Access Security Information is acceptable.

Communications Server NT Version 6.0 or higher offers an additional conversation security type called STRONG. MERVA Link does not support this type.

CICS/ESA V4 Inbound Conversation Security: The conversation-security requirements of a MERVA Link server (MERVA Link receiving process) in the CICS/ESA V4 environment is specified in the CICS CONNECTION definitions and in the CICS® startup parameters. The level of CICS inbound conversation security is specified through the ATTACHSEC parameter of the CONNECTION resource definition. Attach security for inbound transactions is enabled by the CICS startup parameter XTRN. Transaction-attach security is controlled by the RACF® classes TCICSTRN and GCICSTRN.

A short discussion of the CICS attach security parameters follows in the following sections. For a detailed description of the CICS/ESA V4 security facilities refer to the *CICS-RACF Security Guide*.

Conversation Security: The ATTACHSEC parameter can have the following values:

- LOCAL** Disables conversation security. The partner system will not send any security information to CICS. The conversation fails if transaction attach security applies.
- IDENTIFY** Asks for at least a user ID as part of the Access Security Information (with the AV indicator) from the partner system. User ID and password is also accepted. The conversation fails if the user ID is not defined in the receiving system.
- VERIFY** User ID and password are required on incoming attach requests. MERV Link allocate requests with security SAME are downgraded to NONE when the CICS ATTACHSEC is VERIFY. The conversation is accepted unless transaction attach security is implemented in the receiving system.
- PERSISTENT** See CICS documentation.
- MIXIDPE** See CICS documentation.

Transaction-Attach Security: CICS/ESA transaction security is enabled by specifying the CICS startup parameter XTRN=YES or XTRN="resource_class_name". The authorization to start a particular transaction, for example, the MERV Link receiving transaction EKAR, is controlled by the RACF class TCICSTRN. CICS performs an authorization check for every transaction-attach request that is received over an APPC link.

If XTRN=NO is specified, CICS/ESA transaction security is disabled. In this case, the specification of ATTACHSEC VERIFY may not have the desired effect. The conversation security may be downgraded to NONE and the receiving transaction is not protected against unauthorized access.

A secure MERV Link CICS environment must implement transaction-attach security.

APPC/MVS Inbound Conversation Security: The conversation-security requirements of a MERV Link server (MERVA Link receiving process) in the APPC/MVS environment is specified in the SECACPT parameter of the VTAM definition of the receiving APPC/MVS LU. This parameter can be modified via the CONVSEC parameter of a RACF profile for the APPC/MVS LU in RACF class APPCLU.

The access to TP profile data sets and the execution of APPC transaction programs can be controlled via RACF profiles of the class APPCTP. Other RACF classes are defined to support APPC/MVS security.

A short discussion of the VTAM SECACPT parameter (or RACF APPCLU profile parameter CONVSEC) follows. For a detailed description of the APPC/MVS security facilities refer to *MVS/ESA Planning: APPC Management* and the *ITSC redbook APPC Security: MVS/ESA, CICS/ESA, and OS/2*.

SECACPT=NONE or CONVSEC(NONE): SECACPT=NONE means that this LU does not accept security elements from any partner LU in the inbound SNA Attach Header (FMH 5). Any inbound allocate request is downgraded to NONE. No security environment is built for the inbound transaction program that will execute in an APPC/MVS initiator. The transaction program can only access resources that are either not protected by RACF or have a universal access level (UACC) that is sufficiently high.

SECACPT=CONV or CONVSEC(CONV): SECACPT=CONV means that the LU expects a user ID and the corresponding password in an inbound FMH 5. If the outbound allocate request specifies conversation security NONE, the inbound conversation security is downgraded to **no security**. APPC/MVS will allow the conversation to run without a security environment as if SECACPT=NONE was specified.

If the outbound allocate request specifies conversation security PROGRAM, the inbound FMH 5 contains a user ID and the corresponding password. The user ID must be defined in RACF and the password is validated. The inbound conversation is rejected if either of these checks fails.

If the outbound allocate request specifies conversation security SAME, it can send only a user ID without a password. The inbound conversation security is therefore again downgraded to **no security**. APPC/MVS will allow the conversation to run without a security environment as if SECACPT=NONE was specified.

SECACPT=ALREADYV or CONVSEC(ALREADYV): SECACPT=ALREADYV means that the LU uses conversation security but is willing to accept a user ID that has already been verified (authenticated) by the security product in the requesting system. If the outbound allocate request specifies conversation security NONE, the inbound conversation security is downgraded to **no security**. APPC/MVS will allow the conversation to run without a security environment as if SECACPT=NONE was specified.

If the outbound allocate request specifies conversation security PROGRAM, the inbound FMH 5 contains a user ID and the corresponding password. The user ID must be defined in RACF and the password is validated. The inbound conversation is rejected if either of these checks fails.

If the outbound allocate request specifies conversation security SAME, the inbound FMH 5 contains a user ID and the AV indicator. If the user ID is defined in RACF, a security environment will be setup by RACF without an authentication of the user ID. The inbound conversation is rejected if the user ID is not defined in RACF.

SECACPT=ALREADYV should not be used for an LU-LU connection to an untrusted partner system. CICS/ESA, APPC/MVS, and APPC/IMS systems are considered as trusted systems with all user access controlled by RACF and a trusted security administrator making RACF definitions. A Windows NT system is, however, considered as an untrusted system because user and security administrator functions are often performed by the same person.

The SECACPT=ALREADYV parameter that applies for all connections to trusted systems (especially for connections to CICS/ESA) should be overwritten by the CONVSEC(CONV) parameter in RACF APPC LU profiles for all connections to untrusted partner systems.

APPC/IMS Inbound Conversation Security: An APPC/IMS inbound allocate request is handled by APPC/MVS until the receiving TP must be scheduled. APPC/MVS passes an inbound allocate request to the APPC/MVS scheduler ASCH if the receiving LU is associated with the APPC/MVS scheduler. APPC/MVS passes an inbound allocate request to the APPC/IMS scheduler if the receiving LU is the APPC/IMS LU.

All inbound security facilities of APPC/MVS apply for APPC/IMS as well. Additional considerations apply for APPC/IMS because of its means to schedule the inbound TP. For a detailed description of the APPC/IMS security facilities refer to *MVS/ESA Planning: APPC Management* and *IMS/ESA Version 4 Data Communication Administration Guide*.

Inbound TP Scheduling: An inbound APPC/IMS TP is scheduled by the APPC/IMS scheduler in an IMS/ESA[®] Message Processing Region (MPR) when the inbound allocate request has passed all security checks. The MPR selection is based on the parameters CLASS and MAXRGN of the APPC/IMS TP profile definition.

If the inbound security-type is NONE (as requested by the partner or downgraded from any other requested security-type value) and the inbound TP is not protected by a RACF profile in the RACF resource class APPCTP, the inbound TP is scheduled in an MPR. The TP executes in the security environment that has been setup when the MPR was started. Access Security Information has been provided as part of the MVS/ESA (TM) job control information associated with the MPR startup job.

A secure MERVA Link APPC/IMS environment must implement transaction-attach security using the RACF resource class APPCTP.

Inbound TP Security Environment: The security environment that applies for an inbound TP scheduled in an MPR can be controlled by the parameter RACF of the APPC/IMS TP profile definition.

RACF=NONE

causes IMS[™] to call DFSCTRNO Transaction Authorization exit routine. The security environment of the MPR startup job applies for the executing transaction.

RACF=CHECK

causes IMS to call RACF for security checking when IMS receives a transaction, but does not clone the security environment into the dependent region (MPR) when the transaction executes.

RACF=FULL

causes IMS to call RACF for security checking when IMS receives a transaction and clones the security environment into the dependent region (MPR) at execution time.

MERVA Link NT Inbound Conversation Security

The MERVA Link NT Inbound Conversation Security is specified and handled by Communications Server outside the scope of MERVA Link. When the MERVA Link SNA APPC TP is given control by the Communications Server attach manager, all security checks have been successfully passed. This is why this chapter discusses LU 6.2 security options of Communications Server for an inbound allocate request and the outbound security options of partner systems (CICS/ESA, APPC/MVS, APPC/IMS, and Windows NT CS).

Communications Server Security Information: Communications Server checks the user ID and password - as far as applicable - before a transaction program is started. User IDs and passwords in the Windows NT environment are not case-sensitive.

Communications Server LU 6.2 Security Options: Communications Server provides two kinds of security for LU 6.2 configurations:

Session security confirms that the correct local and partner LUs are being connected in an LU-LU session. This kind of security is configured in the configuration option **Configure LU-LU Passwords** in the **SNA Node Configuration** program. For more information about this kind of LU 6.2 security, refer to the Communications Server documentation.

Conversation security is used to ensure that only properly authorized TPs are allowed to communicate. This kind of LU 6.2 security permits only those TPs with the proper security parameters to establish a conversation. You can configure the conversation security options by using the **SNA NODE Configuration** program from the Communications Server program group.

The option **Configure User ID Passwords** contains a list of all user IDs and passwords that are allowed to start a conversation between a TP on this Windows NT system and a partner TP. Click **New** to create a new user entry in this list.

The option **Configure Transaction Programs** let you activate conversation security for a designated TP. To do this, open the Configure Transaction Programs window. Then select **Conversation security required**.

If conversation security is not required, and if the partner LU tries to allocate a conversation with one or more security parameters, you get an error. The error is called **State Parameter Check** and the error number is **TS2A19**. To correct the error, the partner LU must allocate the conversation security without any security parameters set.

CICS/ESA V4 Outbound Conversation Security: CICS/ESA has a good reason to consider itself as a trusted system. The conversation-security type of an outbound allocate request is therefore always **already_verified**. This means, CICS/ESA always sends a user ID and the AV indicator in an outbound FMH 5 if the conversation security is not downgraded because of definitions in the partner system. CICS/ESA cannot send an access security password.

A sending MERVA Link task in the CICS/ESA V4 environment is associated with one of the user IDs available in the CICS/ESA environment. It is one of the following:

PLTPIUSR

The user ID specified in the CICS/ESA startup parameter PLTPIUSR applies if the MERVA ESA nucleus (sample transaction name is DSL) has been started from the CICS/ESA Program List Table (PLT).

DFLTUSER

The CICS/ESA default user ID specified in the startup parameter DFLTUSER applies if no user ID is associated with the MERVA ESA nucleus task, and no other user ID applies.

The applicable user ID must be defined as a Windows NT user if the connection to the CICS system must have the conversation-security type **already_verified**. An inbound allocate request from CICS is rejected otherwise.

If conversation security type **none** or **conversation** is specified in the Logical Connection profile for a partner CICS system, the effective conversation security type is (downgraded to) **none**. CICS/ESA does not send an access security user ID in this case. The inbound allocate request is accepted by Communications Server only if the receiving TP is unprotected (resource security type **none**).

The recommended Communications Server parameters for a connection to a CICS/ESA system are:

- Conversation security level **already_verified**
- Resource security level **conversation**
- CICS/ESA V4 DFLTUSER defined in `/etc/passwd`
- CICS/ESA V4 DFLTUSER defined in CS-ALP (if applicable)

APPC/MVS Outbound Conversation Security: APPC/MVS supports any conversation security that is required by a partner system. APPC/MVS can send a user ID and a password, a user ID and the AV indicator, or no access security information in an FMH 5.

MERVA Link IMS considers its environment as a secure environment (IMS/ESA is considered as a trusted system). This is why a MERVA Link IMS outbound allocate request specifies SECURITY=SAME as a fix parameter. The user password cannot be contained in an FMH 5 that is based on a MERVA Link IMS outbound allocate request.

The conversation security type of an outbound allocate request is **already_verified**. This means, APPC/MVS sends a user ID and the AV indicator in an outbound FMH 5 if the conversation security is not downgraded because of definitions in the partner system.

A sending MERVA Link IMS task executes in an IMS MPR. The security environment of the IMS MPR includes a user ID that identifies an MVS user. Access to the local resources (for example, the MERVA ESA libraries) is controlled via this user ID. It is this user ID that is sent in an outbound FMH 5 to the partner system.

The MVS user must be defined as a Windows NT user if the connection to the APPC/MVS system must have the conversation security type **already_verified**. An inbound allocate request from APPC/MVS is rejected otherwise.

If conversation security type **none** or **conversation** is specified in the Partner LU profile for a partner APPC/MVS system, the effective conversation security type is (downgraded to) **none**. APPC/MVS does not send an access security user ID in this case. The inbound allocate request is accepted by Communications Server only if the receiving TP is unprotected (resource security type **none**).

The recommended Communications Server parameters for a connection to an APPC/MVS system are the same as for a connection to a CICS/ESA system. Instead of the CICS system identifier (SYSIDNT) the MVS user ID that defines the security environment of the IMS MPR must be defined in `/etc/passwd`.

APPC/IMS Outbound Conversation Security: The outbound conversation security considerations for APPC/MVS apply - unless otherwise specified - for APPC/IMS as well.

Currently, no differences between the outbound conversation security rules of APPC/MVS and APPC/IMS are known.

Defining Client Users in Windows NT

If the conversation security of a MERVA Link inbound conversation is other than NONE, the inbound SNA Function Management Header (FMH 5) contains a user

ID. This user ID must be defined as a Windows NT user. You can define it in the same case as on the host or in lower case.

The FMH 5 contains a password if the inbound conversation security is PROGRAM. The password must be identical to the Windows NT password that is defined for the corresponding user. Windows NT passwords and user IDs are not case-sensitive.

Chapter 13. SWIFT Link

This chapter describes the X.25 adapter configuration for SWIFT Link and how to define the logical link between the SWIFT Link application and the X.25 hardware component.

X.25 Adapter Configuration

Prerequisite for the following steps is that the ARTIC Co-Processor Card drivers and the latest X.25 Support on Windows NT are installed. The following sections tell you how to do this.

To install the latest ARTIC Co-Processor Card drivers and the latest X.25 Support for X.25:

1. Open the MERV A homepage <http://www.ibm.com/solutions/finance/merva> with any Web browser.
2. Click **Support** → **MERVA Migration & Installation** → **X.25 Adapter Configuration**.
3. The page **MERVA for Windows NT – X.25 Adapter Configuration** is displayed. Follow the instructions on this page.

Installing the ARTIC Support for Windows NT

To install the ARTIC Support for Windows NT:

1. If an older version of the ARTIC Support is already installed, deinstall the older version.
2. If the ARTIC Support for X.25 on Windows is already installed, deinstall the X.25 Support.
3. Start the executable **wnt###w.exe** with a parameter that points to a temporary directory, where **###** is a placeholder for the version number. For example, open a Command Prompt window and type the following command:

```
wnt###w c:\temp
```

where **###** is a placeholder for the version number.

The files are stored in the directory **c:\temp**.

4. Change to the temporary directory and start the program **setup.exe**.

Note: Before you run **setup.exe** for the first time, you must create a directory for the ARTIC Support files, for example, **c:\artic**.

5. Change to the **bin** directory of the installed ARTIC Support and type the following command:

```
icaldric 0 icaaim.com 0 -reset
```

Note: This command is also described in the installation instructions for the ARTIC Support. The command line applies to systems with only one ARTIC Co-Processor card installed. If you have more than one ARTIC Co-Processor card installed, refer to the installation instructions for the ARTIC Support software.

6. You can also get the diagnostic disks from the **IBM Personal Computing Support** Web page. For information on how to download these disks, refer to "X.25 Adapter Configuration" on page 137.
Use these files to check the ARTIC Co-Processor card installation for hardware and software.
7. Remove the contents of the temporary directory.
8. Restart your system.

Installing the ARTIC Support for X.25 on Windows NT

To install the ARTIC Support for X.25 on Windows NT:

1. If the ARTIC Support for X.25 on Windows NT is already installed, deinstall the X.25 Support.
2. Start the executable **x25###w.exe** with a parameter that points to a temporary directory, where **###** is a placeholder for the version number. For example, open a Command Prompt window and type the following command:

```
x25###w c:\temp
```

where **###** is a placeholder for the version number.

The files are stored in the directory **c:\temp**.

3. Change to the temporary directory and start the program **x25inst.exe**.

Note: If you run **x25inst.exe** for the first time, you must create a directory for the ARTIC X.25 Support files, for example, **c:\x25**.

4. Remove the contents of the temporary directory.
5. Restart your system.

Before you can use the ARTIC X.25 Support program, you have to configure it. How to configure it is described in the following section.

Configuring the ARTIC Support for X.25 on Windows NT

To add or change configuration profiles of the ARTIC X.25 Support, start the executable **x25conf.exe**. This file is located in the target directory of the installation.

You then get the ARTIC X.25 Configuration window. This window contains a notebook with the following pages:

- Link Profiles
- Nickname Entries
- Incoming Call Routing
- Line Tracing

To configure the ARTIC X.25 Support, you have to change the Link Profiles and the Incoming Call Routing.

Customizing X.25 Link Profiles

The following figure shows an example of the Link Profiles page.

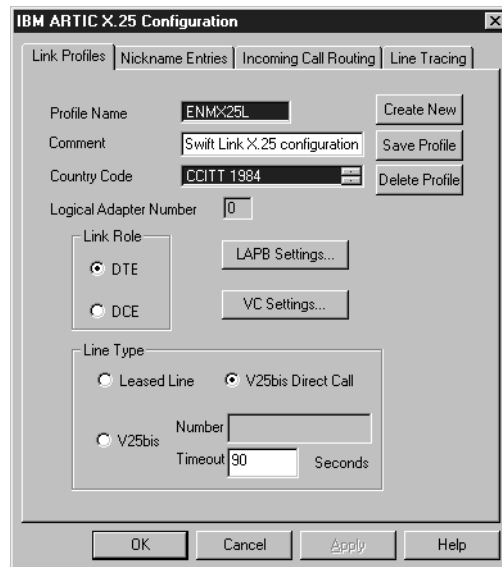


Figure 51. Link Profiles Page

To create a new profile for SWIFT Link click **Create New** and check the following entries on the Link Profiles page:

1. The **Profile Name** field must contain **ENMX25L**.
2. The **Country Code** field must contain **CCITT 1984**.

To change the country code to **CCITT 1984**, highlight the **CCITT 1984** item before you click **Save Profile**. To do this, click the item with the left mouse button. If the item is not highlighted, another country code might be used.

3. The **Link Role** field must contain **DTE**.
4. The contents of the **Line Type** field depend on your SWIFT Link setup:
 - If you use an encryptor box between your computer and the modem the following conditions apply:
 - If the phone number is stored in the encryptor box, you can use **Leased Line** or **V25bis Direct Call**.
 - If the phone number is not stored in the encryptor box, you have to use **V25bis Direct Call**. The number is stored in the modem.
 - If you have only a modem but no encryptor box, you can use **V25bis Direct Call** and store the phone number in the modem. You can also use **V25bis** and enter the phone number in the **Number** field.

Note that the phone number can contain special characters as defined in the V25bis standard. After you dial the first digit, you might have to wait for the dial tone and then insert a W. For information about the correct control characters, refer to the description of your modem.

5. The **Timeout** field specifies the amount of time in which the modem dials and confirms state *ready* of the connection. If the modem needs more time than indicated in the **Timeout** field, increase the value in this field.
6. Click **Save Profile** to save the entries.

Additionally, you can set the following parameters on the Link Profiles page:

- Link Access Procedure - Balanced (LAPB)
- Virtual Circuit (VC)

The following figure shows the default LAPB settings that are required for the SWIFT Link.

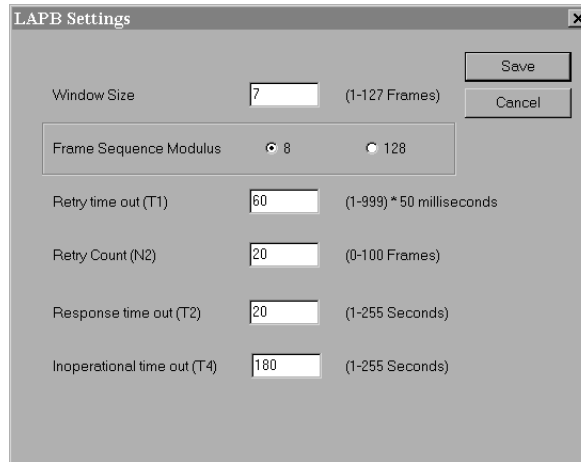


Figure 52. The LAPB Settings Window

The values shown in this window are default values. You do not have to change them.

Click **Save** to save the settings shown.

The following figure shows possible settings for the Switched Virtual Circuits (SVCs).

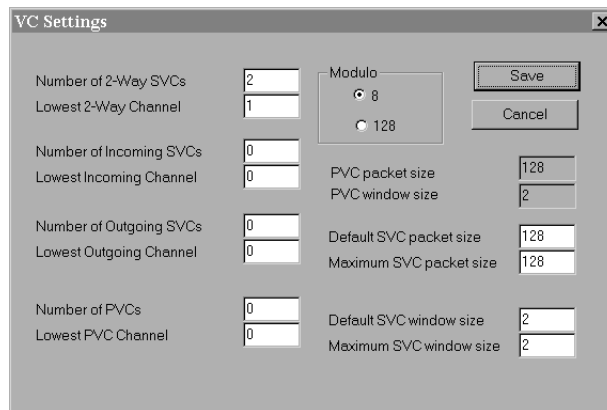


Figure 53. The VC Settings Window

The number of 2-way SVCs can be 1 to 10. For a leased line, 1 is sufficient. For a switched line, the minimum number of 2-way SVCs is 2. Otherwise, collision situations can occur because both sides try to establish a connection. The 2 SVCs are needed only for the period of time in which the collision is solved.

Click **Save** to save the changes.

Customizing the Incoming Call Routing Page

The following figure shows an example of the Incoming Call Routing page:

The screenshot shows the 'IBM ARTIC X.25 Configuration' window with the 'Incoming Call Routing' tab selected. The 'Route Entry Name' field contains 'ENMX25R' and the 'Description' field contains 'Swift Link Routing Table Entry'. The 'Link Profile Name' field contains 'ENMX25L'. Below these are five fields for 'Called Address', 'Called Address Ext', 'Calling Address', 'Calling Address Ext', and 'Call User Data', each containing an asterisk (*). At the top right are buttons for 'Create New', 'Delete Entry', and 'Save Changes'. At the bottom are buttons for 'OK', 'Cancel', 'Apply', and 'Help'.

Figure 54. The Incoming Call Routing Page

If an external computer dials in, different applications can be active and wait for the incoming call. The Incoming Call Routing table specifies the application to handle the incoming call.

To create a new X.25 routing table:

1. Click **Create New**.
2. In the **Route Entry Name** field, type **ENMX25R**.
3. In the **Link Profile Name** field, type **ENMX25L**.
4. Check that all other fields are filled with an asterisk (*). The * represents a wildcard character that allows any address or data.
5. Click **Save**.
6. Click **Save Changes** to save the entries.
7. Click **OK** to leave the configuration program.

Part 4. Appendixes

Appendix A. MERV Link Application User Exits

The MERV Link application user exit allows the access to MERV messages and system data at a defined point during the MERV Link transfer process on the Application Support (AS) layer.

All User Exit functions are held in a shared module. You can specify a shared module for each ASP during MERV Link customization. The module is loaded dynamically when a MERV Link transfer is started. If the specified module is not found, the default module **ekauexit.obj** is used. The name of the user exit directory is specified at the time the MERV instance is created in the MERV Instance Settings window of the MERV Control Center.

Multiple ASPs can use the same shared module.

The MERV Link transfer processes call the User Exit functions at a fixed point during message transmission.

The User Exit functions use the User-Exit Interface Library **ekauexit.dll** to access message-related data. This library provides:

- User-Exit Interface functions to copy and update data fields.
- Functions to allow the default processing of status report generation and correlation.

Data Types of the User-Exit Interface

A set of symbol definitions, data types, and functions is specified to ease the use of the User-Exit Interface.

The definitions and data types are summarized in the default user-exit header file **ekauexit.h** that is supplied with MERV USE & Branch for Windows NT.

- Network defines:

```
#define NET_SWIFT '2'  
#define NET_TELEX '3'  
#define NET_OWN  '4'
```

- Logging defines:

```
#define LOG_IAM           0x0001  
#define LOG_ISN          0x0002  
#define LOG_MSGNET      0x0004  
#define LOG_MSGOK       0x0008  
#define LOG_MSGACK      0x0010  
#define LOG_MSGROUTE    0x0020  
#define LOG_MSGCMNT     0x0040  
#define LOG_MSGTEXT     0x0080  
#define LOG_BUCKSLIP    0x0100  
#define LOG_SUBJECT     0x0200  
#define LOG_MSGTYPE     0x0400  
#define LOG_MRN         0x0800  
#define LOG_MSGUSER     0x1000  
#define LOG_UEXITFLAG   0x2000  
#define LOG_LASP        0x4000  
#define LOG_STATUSREP   0x8000
```

- Field Type (FIELDTYPE):

The field type is an enumerated data type of valid field names for data associated with a message currently processed by MERVA Link.

```
typedef enum {
    FLD_IAM,
    FLD_ISN,
    FLD_MSGNET,
    FLD_MSGOK,
    FLD_MSGACK,
    FLD_MSGROUTE,
    FLD_MSGCMNT,
    FLD_MSGTEXT,
    FLD_LASP,
    FLD_MRN,
    FLD_MSGUSER,
    FLD_UEXITFLAG1,
    FLD_UEXITFLAG2,
    FLD_BUCKSLIP,
    FLD_SUBJECT,
    FLD_MSGTYPE
} FIELDTYPE;
```

The field name must be specified using API functions.

- Enumerations for the ENMWriteToLogFile function
 - The enumeration CON_MSG_ID specifies whether a message has to be displayed on the message console. It also specifies the message type that has to be displayed.

```
enum CON_MSG_ID {
    CON_ID_NONE = '.',
    CON_ID_INFO = 'I',
    CON_ID_ERROR = 'E',
    CON_ID_FATAL = 'F'
};
```

- The enumeration INTERVENTION specifies whether operator intervention for the problem is required.

```
enum INTERVENTION {
    INT_NOT_REQ = '.',
    INT_REQ = 'R'
};
```

- Status Report Field Type (SRFIELDTYPE):

The status report type is an enumerated data type of valid field names for data associated with the status report.

```
typedef enum {
    FLD_REPORTDATA1 = 1,
    FLD_REPORTDATA2 = 2,
    FLD_REPORTDATA3 = 3,
    FLD_REPORTDATA4 = 4,
    FLD_REPORTDATA5 = 5,
    FLD_REPORTDATA6 = 6,
    FLD_REPORTDATA7 = 7,
    FLD_REPORTDATA8 = 8
} SRFIELDTYPE;
```

- Status Report Header (STR_RECEIPT_HEADER):

```
typedef struct {
    char    szTime[12+1];
    char    szRC[2+1];
    char    szDC[6+1];
    int     iNbDataElements;
} STR_RECEIPT_HEADER ;
```

Data Fields of the User-Exit Interface

The User-Exit Interface functions can access message-related information. This data is located in the MERVA Link transfer process and copied to or from the memory that is specified by the User-Exit Interface functions.

Table 17 provides a description of each field type, the length of the associated field, and the access right.

Table 17. Overview of All Field Types That Are Specified

Field Type	Length	Access	Description
Message-Related Data Fields			
FLD_MRN	16	read	Message reference number.
FLD_MSGUSER	8	read	ID of the user who last modified the message.
FLD_ISN	24	read/write	Input sequence number. It is used for messages sent to the SWIFT network.
FLD_IAM	24	read	Interapplication message identifier. It is used by MERVA Link to correlate the received status report with the message waiting in the acknowledgment wait queue.
FLD_MSGNET	1	read/write	Identifier for the network used. It is used to identify the message class. MERVA uses the following classes: <ul style="list-style-type: none"> • SWIFT network • OWN network • TELEX network Note: MSGNET can only have one of the following values: <ul style="list-style-type: none"> • NET_SWIFT • NET_OWN • NET_TELEX
FLD_MSGOK	8	read/write	Message OK field. It is a free-format text field and can be used for routing.
FLD_MSGACK	127	read/write	Message acknowledgment field. This information is not directly sent to the partner, because the acknowledgment is not part of the message. Note: MERVA Link provides a data area for transferring additional data to the partner. This area is used to transmit the acknowledgment to the partner.
FLD_MSGROUTE	19	read/write	Message route field. It is a free-format text field and can be used for routing.
FLD_MSGCMNT	1999	read/write	Message comment field.
FLD_MSGTEXT	28000	read/write	Message text.
FLD_UEXITFLAG1	1	read	First user exit flag specified during customization.
FLD_UEXITFLAG2	1	read	Second user exit flag specified during customization.
MERVA Link Data Fields			
FLD_LASP	8	read	Name of the related local ASP.
FLD_BUCKSLIP	256	read/write	The MERVA Link data area for a short notice attached to the message.

Table 17. Overview of All Field Types That Are Specified (continued)

Field Type	Length	Access	Description
FLD_SUBJECT	60	read/write	The MERVA Link data area to specify a subject.
FLD_MSGTYPE	8	read/write	The MERVA Link data area to specify a MERVA Link message type.

MERVA Link Status-Report Header Fields

A status report can contain user-defined data elements for correlation purposes. One status-report data element contains an identifier and the data. MERVA Link allows up to eight report data elements. You can copy data from or assign data to such data elements.

The MERVA Link application user exit specifies whether to send a status report (receipt report) or the message (refer to “ENMOutgoing—Send the Message or Status Report” on page 162). The status report is used to return notification of receipt or nonreceipt of a message to its originator.

When MERVA Link receives a status report both, the status report and the message waiting in the MERVA Link Ack Wait queue, are specified within the data element of the application user exit. The MERVA Link application user exit then correlates the message with the report. MERVA Link provides a default status-report generation and correlation for SWIFT messages.

When the transfer process creates, sends, or receives a status report the following additional data fields are available.

Table 18. MERVA Link Status-Report Header Fields

Field	Description	Access
szTime	Receipt date/time stamp	read/write
szRC	Receipt return code	read/write
szDC	Receipt diagnosis code	read/write
iNbDataElements	Number of receipt-report-data elements	read

MERVA Link User-Exit Interface Functions

This section lists the functions of the MERVA Link user-exit interface, and for each function:

- Describes its purpose and format
- Shows its syntax of each function
- Shows the number and order of its parameters

These functions can be used in the predefined user exit functions described in “MERVA Link Application User Exit Functions” on page 160).

ENMGetField—Copy Data from the Specified Field

Purpose

The ENMGetField function copies the content of the specified field into the specified data buffer.

Format

```
iRc = ENMGetField( FieldType, pchFieldDataBuf, iFieldDataBufLen,  
piBytesWritten );
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-20
Specified data buffer is too small.

-100
The content of the specified field is not yet valid in the current user exit function.

FieldType (FIELDTYPE) - input

Specifies the field that has to be copied.

pchFieldDataBuf (char *) - input

Specifies the data buffer where the content has to be stored.

iFieldDataBufLen (int) - input

Specifies the size of the specified data buffer.

piBytesWritten (int*) - output

Points to the number of bytes written.

ENMPutField—Put Data to the Specified Field

Purpose

The ENMPutField function puts the content of the specified data buffer to the specified field.

Format

```
iRc = ENMPutField( FieldType, pchFieldData, iFieldDataLen );
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-20

The size of the data field is smaller than the length of the data to be written.

-100

The content of the specified field is not yet valid in the current User Exit function.

FieldType (FIELDTYPE) - input

Specifies the field that has to be changed.

pchFieldData (char *) - input

Specifies the data buffer that is copied to the internal data buffer.

iFieldDataLen (int) - input

Specifies the length of the specified data field.

ENMTraceFields—Write Information to Trace File

Purpose

The ENMTraceFields function writes information about all internal data fields to the MERVA Link trace file. This function only writes data to the trace file when logging is activated (in MERVA Link operating).

Format

```
iRc = ENMTraceFields( pchString, LOG_... );
```

Parameters

iRc (int) - return

Values are:

- 1 Logging is not active.
- 0 Function completed successfully.
- 1 Unable to write to file.

pchString(char *) - input

Pointer to a comment string that is to be printed before the logging fields are printed.

LOG_...(Logging fields) - input

For example: LOG_MSGTXT | LOG_MSGCMNT | LOG_BUCKSLIP

ENMWriteToLogFile—Write Information to Diagnosis Log and Message Console

Purpose

The ENMWriteToLogFile function writes its own information to the MERVA diagnosis log and, if specified, to the message console.

Format

```
iRc = ENMWriteToLogFile( pszMsgTxt, eErrLvl, eIntervention );
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

<0 Could not write information to the diagnosis log.

pszMsgTxt(char *) - input

Pointer to a string that is to be printed.

eErrLvl(enum CON_MSG_ID) - input

Specifies whether the message has to be displayed on the message console (if != CON_ID_NONE) and which kind of message (INFO, ERROR, or FATAL ERROR) it is.

eIntervention(enum INTERVENTION) - input

Specifies that an operator intervention is required for the described problem. If eErrLvl is set to CON_ID_NONE, this parameter is ignored.

ENMGenerateDefaultStatusReport—Generate the Status Report for SWIFT Messages

Purpose

The ENMGenerateDefaultStatusReport function generates the status report for SWIFT messages according to the MERVA ESA User Exit.

Format

`iRc = ENMGenerateDefaultStatusReport();`

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-100

The content of the specified element is not yet valid in the current User Exit function.

-1000

Unexpected error.

-2000

Operating system error.

-3000

Syntax error in SWIFT message.

ENMCorrelateDefaultStatusReport—Update the Message Field

Purpose

The ENMCorrelateDefaultStatusReport function updates the message field according to the status report. The status report is generated like the MERVA ESA User-Exit Interface status report.

Format

```
iRc = ENMCorrelateDefaultStatusReport();
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-100

The content of the specified element is not yet valid in the current User Exit function.

-1000

Unexpected error.

-3000

Syntax error in SWIFT message.

ENMGetStatusRepHeader—Copy the Status Header Information

Purpose

The ENMGetStatusRepHeader function copies the status-report header information to the specified data structure.

Format

```
iRc = ENMGetStatusRepHeader( pReceiptHeader );
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-100

The content of the specified element is not yet valid in the current User Exit function.

-1000

Unexpected error.

pReceiptHeader(STR_RECEIPT_HEADER *) - output

Pointer to a status report header structure.

ENMPutStatusRepHeader—Change the Status Report Header Information

Purpose

The ENMPutStatusRepHeader function changes the status report header information.

Format

```
iRc = ENMPutStatusRepHeader( pReceiptHeader );
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-100

The content of the specified element is not yet valid in the current User Exit function.

-1000

Unexpected error.

pReceiptHeader(STR_RECEIPT_HEADER *) - input

Pointer to a status report header structure.

ENMGetStatusRepData—Copy the Information of a Status Report Data Field

Purpose

The ENMGetStatusRepData function copies the information of a status report data field to the specified buffer. A data field points to a data element. A data element consists of a data ID and data.

Format

```
iRc = ENMGetStatusRepData( SRFieldType, pusDeId, pcaDataBuf, iDataBufLen, piBytesWritten);
```

Parameters

iRc (int) - return

Values are:

0 Function completed successfully.

-20

Field data size is smaller than the specified data length.

-100

The content of the specified element is not yet valid in the current user exit function.

-1000

Unexpected error.

-4000

SRFieldType is out of range or no data element is specified for this index.

SRFieldType (SRFIELDTYPE) - input

Specifies the status report data field that has to be copied.

pusDeId (unsigned short *) - output

Contains the ID of this data element.

pcaDataBuf (unsigned char *) - output

Points to the memory to copy the data of the data element.

iDataBufLen (int) - input

Specifies the size of the specified data buffer.

piBytesWritten (int *) - output

Points to the number of bytes written.

ENMPutStatusRepData—Copy the Data and the Identifiers of a Report Data Field

Purpose

The ENMPutStatusRepData function copies the data and the identifiers of a report data field to the internal data structure.

Format

```
iRc = ENMPutStatusRepData( SRFieldType, usDeId, pcaDataBuf, iDataBufLen );
```

Parameters

iRc (int) - return

Values are:

0 (- no error)

Function completed successfully.

-100 (- error)

The content of the specified element is not yet valid in the current User Exit function.

-1000 (- error)

Unexpected error.

-2000 (- error)

Operating system error.

-4000 (- error)

SRFieldType is out of range.

SRFieldType (SRFIELDTYPE) - input

Specifies the status report data field which content has to be changed.

usDeId (unsigned short) - input

Contains the ID of this data element.

pcaDataBuf (unsigned char *) - input

Points to the data that is to be written to the specified field.

iDataBufLen (int) - input

Specifies the size of the specified data buffer.

ENMIsStatusRepDataValid—Check if a Status Report Data Element Exists

Purpose

The ENMIsStatusRepDataValid function can be used to check if a status-report data element exists for a given data field.

Format

```
iRc = ENMIsStatusRepDataValid( SRFieldType );
```

Parameters

iRc (int) - return

Values are:

0 (- no error)

There is no data element.

1 (- no error)

A data element exists.

-100 (- error)

The content of the specified element is not yet valid in the current User Exit function.

-1000 (- error)

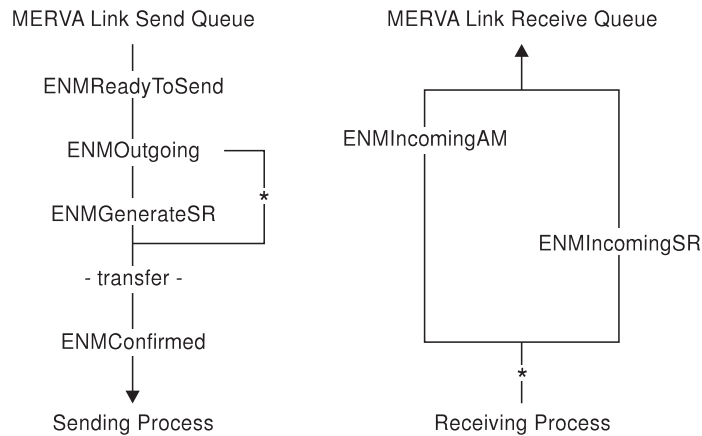
Unexpected error.

SRFieldType (SRFIELDTYPE) - input

Specifies the status report data field that must be checked.

MERVA Link Application User Exit Functions

The MERVA Link Application User Exit functions are called at a defined point during the MERVA Link transfer process. The following user exit functions are provided for the AS layer.



* Shows the position when messages are converted from or to ASCII / EBCDIC (message transfer to/from MERVA ESA).

Figure 55. Function Calls during the Transfer Process

The user exit functions are accessed as follows:

- All functions are contained in a user-exit module. For each ASP, the module can be selected when customizing MERVA Link.
- Multiple ASPs can use the same module.
- MERVA Link loads the functions from the specified module. If no module or a wrong module is specified, MERVA Link loads the functions from the default module and writes a diagnosis log entry.

ENMReadyToSend—Change Message Text or Related Data

Purpose

The ENMReadyToSend function allows you to change the message text or related data of the message (queue buffer information) that is currently processed. It is called immediately after a message is retrieved from the ready-to-send queue and before it is transmitted and routed. In this step a message can be checked and it can be decided whether to send the message or not, or to route it without sending.

Format

```
iRc = ENMReadyToSend();
```

Parameters

iRc (int) - return

Values are:

0 (- no error)

The transfer process continues to send the message.

>0 (- error)

The transfer process does not send the message. The message is routed according to the routing conditions.

<0 (- error)

The transfer process terminates. Any changes done in message-related fields are lost.

Default processing

— None —

Note

The fields FLD_BUCKSLIP, FLD_SUBJECT, and FLD_MSGTYPE cannot be accessed.

ENMOutgoing—Send the Message or Status Report

Purpose

The ENMOutgoing function decides whether the complete message or a status report is sent. This call decides whether the User Exit function ENMGenerateSR is called or not.

This function is called immediately before a message is transmitted from the AS layer to the protocol layer. The message has already been routed to the control queue.

Format

iRc = ENMOutgoing();

Parameters

iRc (int) - return

Values are:

0 (- no error)

The message is sent.

1 (- no error)

A status report must be sent. The User Exit function ENMGenerateSR is called.

<0 (- error)

The message is not sent to the partner. The MERVA Link transfer process stops. The message is not removed from the send queue.

Default processing

This function returns the appropriate code for a SWIFT or Telex acknowledgment. No code is returned, if a message is to be sent.

Example

```
int ENMOutgoing( void )
{
    /* Decide to send message or status report dependent on:          */
    /* 1. Message acknowledgment is not empty                        */
    /* 2. Message must be sent via MERVA Link (message has an IAM id) */
    /* 3. The customization flag is set (inside customizer)          */
    /* When the message will be sent, copy ack field into bucksliip  */
    char  szMsgAck [MSG_ACK_LENGTH];
    char  szIAMId  [IAM_LENGTH];
    char  cFlag1;
    int   *piBytesWritten;
    int   iRcMsgAck, iRcIAMId, iRcFlag1;

    iRcMsgAck = ENMGetField( FLD_MSGACK, szMsgAck, sizeof(szMsgAck), piBytesWritten );
    iRcIAMId  = ENMGetField( FLD_IAM, szIAMId, sizeof(szIAMId), piBytesWritten );
    iRcFlag1  = ENMGetField( FLD_UEXITFLAG1, &cFlag1, 1, piBytesWritten );

    if ((iRcMsgAck==0) && (iRcIAMId==0) && (iRcFlag1==0))
        return( 1 );

    if (iRcMsgAck==0)
        iRcMsgAck = ENMPutField( FLD_BUCKSLIP, szMsgAck, strlen(szMsgAck) );

    return( 0 );
}
```

ENMGenerateSR—Generate the Message Type Specific Status Report

Purpose

The ENMGenerateSR function generates the message-type specific status report. The partner must be able to correlate the status report with the message waiting in the MERVA Link acknowledgement wait queue.

This function is called after the User Exit function ENMOutgoing decided to send a status report. (Refer to “ENMOutgoing—Send the Message or Status Report” on page 162.)

Format

```
iRc = ENMGenerateSR();
```

Parameters

iRc (int) - return

Values are:

0 (- no error)

Function completed successfully.

<0 (- error)

No status report is generated. The MERVA Link transfer process stops.

Default processing

This function generates the SWIFT or Telex status report.

Example

```
int ENMGenerateSR( void )
{
    /* Generate SWIFT status report according to MERVA ESA user exit */
    /* Use default function for generating the SWIFT status report */

    int iRc;

    iRc = ENMGenerateDefaultStatusReport();

    return( iRc );
}
```

ENMConfirmed—Create Application or User-Specific Acknowledgment

Purpose

The ENMConfirmed function creates an application or user-specific acknowledgment. Add additional information for further processing, such as routing.

This function is called when the message has been confirmed by the partner system. It has not yet been routed out of the send queue.

Format

iRc = ENMConfirmed();

Parameters

iRc (int) - return

The return value does not have an impact on further processing.

Default processing

— None —

ENMIncomingAM—Message-Type Specific Processing

Purpose

The ENMIncomingAM function defines message-type specific processing. It changes the message text or related data of the message that is currently processed.

This function is called when the message is not yet in the queue, has not yet been confirmed, and has not yet been routed. It can still be rejected.

Format

`iRc = ENMIncomingAM();`

Parameters

iRc (int) - return

Values are:

0 (- no error)

Function completed successfully.

<0 (- error)

The message is rejected. The MERVA Link transfer process stops.

Default processing

This function retrieves acknowledgment information from the buckslip and generates the system-specific network ID.

Example

```
int ENMIncomingAM( void )
{
    /* Copy data from buckslip field to message */
    /* acknowledgment field */

    char szMLBuckslip [BUCKSLIP_LENGTH];
    int *piBytesWritten;
    int iRc;

    iRc = ENMGetField( FLD_BUCKSLIP, szMLBuckslip, sizeof(szMLBuckslip), piBytesWritten );
    if (iRc != 0) {
        return( -1 );
    };

    if (*piBytesWritten <= MSG_ACK_LENGTH-1) {

        iRc = ENMPutField( FLD_MSGACK, szMLBuckslip, *piBytesWritten );
        if (iRc != 0) {
            return( -1 );
        };
    } else {
        return( -1 );
    };
    return( 0 );
}
```

ENMIncomingSR—Update Message

Purpose

When a status report is received, MERVA Link expects the corresponding message to be in the MERVA Link Ack Wait queue. The ENMIncomingSR function correlates the message with the status report. The message is updated with information from the status report. If the corresponding message is not found, the user exit writes a log.

This function is called when message correlation has been done. If a status report has been received, the message-related fields contain the data of the corresponding message. These fields are empty, if the message is not found.

Format

iRc = ENMIncomingSR();

Parameters

iRc (int) - return

Values are:

0 (- no error)

Function completed successfully.

<0 (- error)

The status report is rejected. The MERVA Link transfer process stops.

Default processing

This function updates the message with information from the status report, if the message correlation was successful. For unsuccessful correlation, it writes information to the diagnosis log.

Example

```
int ENMIncomingSR( void )
{
    /* Correlate message waiting in the acknowledgment wait queue */
    /* with status report received. */
    /* Use default correlation routine. */

    int iRc;

    iRc = ENMCorrelateDefaultStatusReport();

    return( iRc );
}
```

Generating a MERVA Link Application User Exit

You need the VisualAge for C++ for Windows Compiler Version 3.5 to build a MERVA Link application user exit. Some files are located in the subdirectory **samples\userexit** of the installation directory of MERVA USE & Branch for Windows NT.

ekauexit.c The default user-exit source code

ekauexit.h The default user-exit header file

ekauexit.mak The default user-exit make file

The file `ekauexit.c` is provided as an example. To create your own user exit, copy this file, change the User Exit functions, and use the make file supplied. The make process is started with the command:

```
nmake -f ekauxit.mak
```

For more information about the make file and the command refer to the appropriate compiler documentation.

Defining a User Exit for MERVA Link

To define a user exit for MERVA Link:

- Specify the path for the user exits, when creating the MERVA instance. For more information refer to “Creating a MERVA Instance” on page 14.
- Specify the name of a specific User Exit module during MERVA customization on the ASP Information window shown in Figure 31 on page 63.
- Copy the User Exit module to the specified directory.

In addition to the user exit, you can specify two flags that can be used in the User Exit modules.

Appendix B. Standard Routing

The following tables define the standard routing of MERVA USE & Branch for Windows NT. You can use standard routing for production or change it according to your needs.

The information is divided into the following parts:

- A description of the purpose groups, message parts, fields, and constants.
- The complete routing conditions.
- A list, in alphabetical order, of the supplied queues with an explanation of the corresponding routing conditions.

Purpose Groups

Table 19. Purpose Groups

Purpose Group	Notes	Max. Queues
API	Messages from or to external applications	No limit ¹
Delete ²	Messages routed here are deleted	1
Corrupt Messages	Messages containing errors	No limit
MERVA Link Received ³	Entry point for messages received by MERVA Link	No limit
MERVA Link Ready to Send	Messages ready to be sent by MERVA Link	No limit
MERVA Link Ack Wait	Messages waiting to be correlated with a MERVA Link status report	No limit
MERVA Link Control	Messages received by MERVA and temporarily kept to allow message integrity checking	1
Message Completion	Messages to be completed	No limit
Message Retype Verification	Messages requiring verification	No limit
Message Authorization	Messages ready for authorization	No limit
Message Edit	Messages failing authorization or verification	No limit
Message Creation ³	Entry point for messages created by users	1
Print	Messages for automatic printing	9999
SWIFT Ready to Send	Messages ready to be sent by the SWIFT Link	No limit
SWIFT Manual Authentication	Messages that require manual authentication	No limit
SWIFT Received ³	Entry point for messages received from SWIFT. Up to 2 per LT allowed	No limit
USE Received Messages	Input to the USE Background Process: MT 96X, MT 076, MT 087, MT 092, MT 094	No limit
USE Received Commands	Input to the USE Background Process: Free-format Messages (MT 999) that are created by another MERVA system for MERVA-internal USE processing	No limit
USE NAKed	Input to the USE Background Process: Messages that are negatively acknowledged	No limit
USE MT 960/966	Incoming BKE Initiation Requests (without pre-agreement) and BKE Discontinuation Requests	No limit
USE Message Entry ³	Entry point for all messages created by SWIFT USE functions	1
Telex NAKed	Negatively acknowledged Telex messages	No limit
<p>Notes:</p> <ol style="list-style-type: none"> 1. In general, there is no limit. There is, however, a system limitation of 1024 routing names. The sum of queue names, field names, and constant names must not exceed 1024. 2. All messages that are routed to Delete are automatically deleted. 3. Queues that belong to these purpose groups represent the entry point for routing. They do not contain messages. 		

Standard Message Queues

Table 20. Standard Message Queues

Message Queue	Purpose Group	Notes
MBDELETE	Delete	Messages entering this queue are deleted immediately.
MBMERROR MBRERROR MBCRCERR	Corrupt Messages	Messages containing routing errors, messages that cannot be routed, and messages containing CRC errors.
MLAKWAIT	MERVA Link Ack Wait	Messages waiting to be correlated with a MERVA Link status report.
MLCNTRL	MERVA Link Control	MERVA Link control queue for checking message integrity.
MLMERROR	Corrupt Messages	Messages that cannot be handled by MERVA Link.
MLRECEIV	MERVA Link Received	Message received by MERVA Link.
MLSEND	MERVA LINK Ready to Send	Messages to be sent by MERVA Link.
SL_IN	API	Entry point for messages from applications to be sent to SWIFT.
SLRCVFIN SLRCVNST SLRCVSY	API	Messages received from SWIFT to be retrieved by an application.
SLRL1ACK	API	FIN messages successfully sent to SWIFT.
SLINCMS1 SLINCMS2	SWIFT Received	Messages received from SWIFT.
SLMANAUT	SWIFT Manual Authentication	Messages received here when Automatic Authentication has failed.
SLPRINT1 SLPRINT2	Print	SWIFT Messages to be printed.
SLRNRM02 SLRSYS01 SLRSYS02 SLRURG02	SWIFT Ready to Send	Used by SWIFT Link processes.
SMCREATE	Message Creation	Entry point for messages created by users.
SMEDIT	Message Edit	Messages to be repaired.
SMINCMPT	Message Completion	Messages saved from Create Messages.
SMVERIFY	Message Retype Verification	Messages to be verified and retyped.
SMAUTH1	Message Authorization	Messages to be authorized.
TP2_ACK	API	Positively acknowledged telex messages.
TP2_NAK	Telex NAKed	Negatively acknowledged telex messages.
TP2_RCV	API	Received telex messages.
TP2_SND	API	Telex messages ready to be sent.
TP2_WAIT	API	Telex messages waiting for acknowledgment.
UNAKED	USE NAKed Messages	Input to the USE Background Process.
UFROMSWF	USE Received Messages	Input to the USE Background Process.
USEINCMD	USE Received Commands	Command input to the USE Background Process.
USESEND	USE Message Entry	Entry point for all USE messages created by USE functions.

Table 20. Standard Message Queues (continued)

Message Queue	Purpose Group	Notes
UWOPREAG	USE MT 960/966	Incoming BKE Initiation Requests for which no pre-agreement exists and incoming BKE Discontinuation Requests.

Message Parts

Table 21. Message Parts

Message Part Name	Length	Explanation	Type
MESSAGE BUFFER	max. 28000	The entire message, including the header.	Text
MRN FIELD	16	Contains the Message Reference Number (MRN).	Text
MSGACK FIELD	127	Contains, for example, the SWIFT acknowledgment or SWIFT Link error message.	Text
MSGCOMM FIELD	1999	Contains, for example, user comments about the message.	Text
MSGNETID FIELD	1	Identifies the target network: 2=SWIFT 3=Telex 4=NET_OWN	Numeric
MSGOK FIELD	8	Used to classify the message status.	Text
MSGROUTE FIELD	19	Contains, for example, values for routing decisions that are made by an application program.	Text
MSGTRUSR FIELD	8	Contains the user ID of the user who last worked on the message, for example, who authorized the message.	Text
MSGTXT1 FIELD	8	Used by MERVA Link. Name of the LU from which the message was received.	Text
MSGTXT2 FIELD	49	This field is reserved for future use.	Text
MSGTXT3 FIELD	8	Used by MERVA Link. Name of the ASP from which the message was received.	Text
MSGTXT4 FIELD	999	Can be used to store telex envelope data (see Table 22 on page 174).	Text
MSGTXT5 FIELD	999	Used by SWIFT Link. Contains information about authentication.	Text
MSGUSER FIELD	8	Contains the user ID of the user who last changed the message.	Text
SWIFT TEXT	-	Message text, which begins after "{4:". Length varies.	Text

Layout of Message Part MSGTXT4 FIELD (Telex Envelope Data)

The message part MSGTXT4 FIELD is divided as follows:

- The **Telex Header** contains data required to send a message using the telex network.
- The **Telex Information** contains data describing the telex transmission process.

For a detailed description of the fields refer to *MERVA USE & Branch for Windows NT Application Programming Guide*.

Telex Header

Table 22. Layout of MSGTXT4 FIELD (Telex Header)

Field Name	Offset	Length
testkey_cal	0	1
testkey_rc	1	1
testkey_val	2	16
testkey_comment1	18	35
testkey_comment2	53	35
sender_addr0	88	35
sender_addr1	123	35
sender_addr2	158	35
sender_addr3	193	35
date	228	8
to_id	236	11
receiver_addr0	247	35
receiver_addr1	282	35
receiver_addr2	317	35
receiver_addr3	352	35
line	387	2
dial_up1	389	20
answ_back1	409	20
dial_up2	429	20
answ_back2	449	20
type	469	1
timed_time	470	4
time_date	474	8
ref_text	482	16
note	498	64

Telex Information

Table 23. Layout of MSGTXT4 FIELD (Telex Information)

Field Name	Offset	Length
ack_info	700	1
type	701	1
report_nr	702	5
dial_up	707	20
rcv_awb	727	20
starttime	747	14
duration	761	6
reason_code	767	8
telex_box	775	1
telex_line	776	2
term_code	778	1
exceptions	779	1
poss_dupl	780	1
cargo	781	50

Fields

Table 24. Fields

Field Name	Explanation	Message Part	Scan Patterns		Offset	Length	Type
			Start Tag	End Tag			
ACKTELEX	Contains the telex transmission acknowledgment information.	MSGTXT4 FIELD			700	1	Text
ACKINFO	Located in the acknowledgement sent by SWIFT to the CBT: 0=Accepted 1=Rejected	MSGACK FIELD	451:		0	1	Text
ALWAYS	Defines a routing condition that is always true.	MSGNETID FIELD			0	1	Nu- meric
APDU	Identifies the APDU within the basic header.	MESSAGE BUFFER	{1:		1	2	Nu- meric
APPL	Application identifier within the basic header: • "F"=Financial (FIN) • "A"=General Purpose (APC) • "L"=General Purpose (LTC)	MESSAGE BUFFER	{1:		0	1	Text
EMB_TYP1	Block identifier of the retrieved (embedded message). Located in the application header.	SWIFT TEXT	{2:		1	1	Nu- meric
DOMAIN	Contains the MERVA instance name.	MRN FIELD			0	8	Text
MRN	Message reference number	MRN FIELD			8	8	Nu- meric
MLORGLU	LU name of the MERVA Link from which the message was received.	MSGTXT1 FIELD			0	8	Text
MLORGASP	ASP name from which the message was received.	MSGTXT3 FIELD			0	8	Text
MSGACK	Contains the acknowledgment sent by SWIFT	MSGACK FIELD			0	127	Text
MSGCAT	First number of the message type, identifying the message category. Located in the application header.	MESSAGE BUFFER	{2:		1	1	Nu- meric
MSGCOMM	Contains user comments.	MSGCOMM			0	100	Text
MSGGROUP	Identifies the group of messages within a message category.	MESSAGE BUFFER	{2:		1	2	Nu- meric
MSGINFO	Contains either the acknowledgment returned by SWIFT or a message generated by the SWIFT Link.	MSGACK FIELD			0	7	Text
MSGMAC	Contains information about authentication (for example, which key is used for authentication).	MSGTXT5 FIELD			0	127	Text

Table 24. Fields (continued)

Field Name	Explanation	Message Part	Scan Patterns		Offset	Length	Type
			Start Tag	End Tag			
MSGNETID	Contains the network identifier.	MSGNETID FIELD			0	1	Numeric
MSGOK	Contains the message state from the previous function, such as "OK", "CANCEL", or "INCMPLT".	MSGOK FIELD			0	8	Text
MSGPAC	Contains information about authentication (for example, which key is used for authentication).	MSGTXT5 FIELD			127	127	Text
MSGPRITY	Contains the message priority: <ul style="list-style-type: none"> • "N"=Normal • "U"=Urgent • "S"=System 	MESSAGE BUFFER	{2:		16	1	Text
MSGROUTE	Contains values for routing conditions.	MSGROUTE			0	12	Text
MSGTYPE	Contains the SWIFT message type.	MESSAGE BUFFER	{2:		1	3	Numeric
TRN	Contains the SWIFT Transaction Reference Number	MESSAGE BUFFER	:20:		0	16	Text
MSGTRUSR	Contains the user ID of the user who last worked on the message, for example, who authorized the message.	MSGTRUSR			0	8	Text
MSGTXT2	Reserved.	MSGTXT2			0	49	Text
MSGTXT4	Contains telex envelope data.	MSGTXT4			0	999	Text
MSGUSER	Contains the user ID of the user who last changed the message.	MSGUSER			0	8	Text

Constants

Table 25. Constants

Constant Name	Evaluation	Explanation
ABORT_AP	"33"	APDU identifier 33
ABORT_LT	"35"	APDU identifier 35
ACK_ACC	"0"	Acknowledgment information in field 451 is accepted
ACK_REJ	"1"	Acknowledgment information in field 451 is rejected
APC	"A"	General Purpose Application (APC)
APDU_01	"01"	APDU identifier
AUTINKEY	"ENM9978"	See message ENM9978 (incorrect key)
AUTNOKEY	"ENM9979"	See message ENM9979 (key not found)
AUTNOTRL	"ENM9986"	See message ENM9986
AUTTOOLG	"ENM9994"	See message ENM9994 (message too long)
AUT_FAIL	"ENM9985"	See message ENM9128 (authentication failed)
CANCEL	"CANCEL"	Placed into MSGOK field when action CLOSE is performed on Create Messages
DELETE	"DELETE"	Placed into MSGOK field when action DELETE is performed
EDIT	"EDIT"	Placed into MSGOK field when action ROUTE TO EDIT is performed
FAILED	"FAILED"	Placed into MSGOK field when a user marks a message as failed during manual authentication of SWIFT messages fails
FIN	"F"	Financial application (FIN)
INCOMPLT	"INCOMPLT"	Placed into MSGOK field when action SAVE is performed
INC_966	"ENN9193"	Incoming MT 966 (BKE Discontinuation Request)
LOGIN	"02"	APDU identifier of LOGIN message
LOGOUT	"06"	APDU identifier of LOGOUT message
LTC	"L"	General purpose application (LTC)
MTYP_010	"010"	SWIFT message type 010
MTYP_011	"011"	SWIFT message type 011
MTYP_015	"015"	SWIFT message type 015
MTYP_021	"021"	SWIFT message type 021
MTYP_066	"066"	SWIFT message type 066
MTYP_074	"074"	SWIFT message type 074
MTYP_075	"075"	SWIFT message type 075
MTYP_076	"076"	SWIFT message type 076
MTYP_082	"082"	SWIFT message type 082
MTYP_083	"083"	SWIFT message type 083
MTYP_085	"085"	SWIFT message type 085
MTYP_087	"087"	SWIFT message type 087
MTYP_090	"090"	SWIFT message type 090
MTYP_092	"092"	SWIFT message type 092
MTYP_094	"094"	SWIFT message type 094

Table 25. Constants (continued)

Constant Name	Evaluation	Explanation
MTYP_999	"999"	SWIFT message type 999
MTYP_19X	"19"	All SWIFT message types beginning with 19
MTYP_29X	"29"	All SWIFT message types beginning with 29
MTYP_96X	"96"	All SWIFT message types beginning with 96
MTYP_SYS	"0"	System message category
NORMAL	"N"	Normal message priority
NO_PREAG	"ENN9069"	Identifies incoming BKE Initiation Requests (MT 960) for which no pre-agreement exists in the bilateral key database.
OK	"OK"	Placed into the MSGOK field when the message is correct
QUIT	"05"	APDU identifier of the QUIT message
ROUTE	"0"	Used with the ALWAYS field to define a routing condition that is always true
SELECT	"03"	APDU identifier of the SELECT message
SWIFTNET	"2"	SWIFT network identifier
TELEXNET	"3"	Telex network identifier
TX_ACK	"0"	Positive telex transmission acknowledgment
TX_NAK	"8"	Negative telex transmission acknowledgment
URGENT	"U"	Urgent message priority

Routing Conditions

The following tables list the standard routing conditions for MERVA USE & Branch for Windows NT using:

- SWIFT Link
- MERVA Link
- MERVA-MQI Attachment

Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link

This is the standard routing which has been installed when MERVA USE & Branch for Windows NT was installed.

Table 26. Standard Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link

Messages in queue...	Are routed to...	When...
MBDELETE	MBDELETE	
MBMERROR	MBMERROR	
MBCRCERR	MBCRCERR	
MBRERROR		
SL_IN	TP2_SND	MSGNETID = TELEXNET
	SLRSYS01	MSGCAT = MTYP_SYS and APPL = APC
	SLRSYS02	MSGCAT = MTYP_SYS and APPL = FIN
	SMAUTH1	APDU = APDU_01 and APPL = FIN
	MBMERROR	Otherwise
SLRL1ACK	MBDELETE	
SLRCVFIN	MBDELETE	
SLRCVNST	MBDELETE	
SLRCVSY	MBDELETE	
SLPRINT1	MBDELETE	
SLPRINT2	MBDELETE	
SLRSYS01	SLPRINT1	APDU = LOGIN or APDU = LOGOUT or APDU = ABORT_LT
	SLPRINT2	ACKINFO = ACK_ACC or ACKINFO = ACK_REJ
	MBMERROR	Otherwise
SLRSYS02	MBMERROR	MSGNETID <> SWIFTNET
	SLPRINT1	APDU = SELECT or APDU = QUIT or APDU = ABORT_AP
	SLPRINT2	ACKINFO = ACK_ACC or ACKINFO = ACK_REJ
	MBMERROR	Otherwise

Table 26. Standard Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link (continued)

Messages in queue...	Are routed to...	When...
SLRURG02	SMEDIT	MSGOK = EDIT
	TP2_SND	MSGOK = OK and MSGNETID = TELEXNET
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SLRL1ACK	ACKINFO = ACK_ACC
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise
SLRNRM02	SMEDIT	MSGOK = EDIT
	TP2_SND	MSGOK = OK and MSGNETID = TELEXNET
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SLRL1ACK	ACKINFO = ACK_ACC
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise
SLINCMS1	UFROMSWF	MSGTYPE = MTYP_092 or MSGTYPE = MTYP_094
	SLPRINT1	APDU<>APDU_01
	SLPRINT2	MSGCAT = MTYP_SYS
	MBMERROR	Otherwise

Table 26. Standard Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link (continued)

Messages in queue...	Are routed to...	When...
SLINCMS2	SLPRINT1	APDU <> APDU_01
	UFROMSWF	MSGGROUP = MTYP_96X or MSGTYPE = MTYP_087 or MSGTYPE = MTYP_076
	SLRCVSYs	MSGTYPE = MTYP_010 or MSGTYPE = MTYP_011 or MSGTYPE = MTYP_015
	SLRCVSYs	MSGTYPE = MTYP_066 or MSGTYPE = MTYP_082 or MSGTYPE = MTYP_083
	SLRCVSYs	MSGTYPE = MTYP_021 and EMB_TYP1 <> MTYP_SYS
	SLPRINT2	MSGCAT = MTYP_SYS
	SLMANAUT	MSGINFO = AUTNOKEY or MSGINFO = AUT_FAIL
	SLMANAUT	MSGINFO = AUTP_SUSP or MSGINFO = AUT_DISC
	SLRCVNST	MSGGROUP = MTYP_19X or MSGGROUP = MTYP_29X
	SLRCVFIN	Otherwise
SLMANAUT	SLPRINT2	MSGOK = FAILED or MSGINFO = AUTNOKEY or MSGINFO = AUT_FAIL
	SLPRINT2	MSGINFO = AUT_SUSP or MSGINFO = AUT_DISC
	SLRCVNST	MSGGROUP = MTYP_19X or MSGGROUP = MTYP_29X
	SLRCVFIN	Otherwise
SMAUTH1	SLRURG02	MSGOK = OK and MSGNETID = SWIFTNET and MSGPRITY = URGENT
	SLRNRM02	MSGOK = OK and MSGNETID = SWIFTNET and MSGPRITY = NORMAL
	TP2_SND	MSGOK = OK and MSGNETID = TELEXNET
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMEDIT	SMVERIFY	MSGOK = OK
	MBDELETE	MSGOK = DELETE
	SMEDIT	MSGOK = INCOMPLT
	MBMERROR	Otherwise

Table 26. Standard Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link (continued)

Messages in queue...	Are routed to...	When...
SMINCMPT	SLRSYS01	MSGOK = OK and APPL = APC
	SLRSYS01	MSGOK = OK and APPL = LTC
	SLRSYS02	MSGOK = OK and APPL = FIN and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = DELETE or MSGOK = CANCEL
	MBMERROR	Otherwise
SMVERIFY	SMAUTH1	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
MLMERROR	MBMERROR	
MLSEND	MLAKWAIT	MSGACK is EMPTY and MSGNETID = TELEXNET
	MLAKWAIT	MSGACK is EMPTY and MSGNETID = SWIFNET
	MBDELETE	MSGNETID = TELEXNET
	MBDELETE	MSGNETID = SWIFNET
	MBMERROR	Otherwise
MLRECEIV	MLCNTRL	ALWAYS >= ROUTE and continue routing
	SLRSYS01	MSGCAT = MTYP_SYS and APPL = APC
	SLRSYS02	MSGCAT = MTYP_SYS and APPL = FIN
	SLRURG02	APDU = APDU_01 and APPL = FIN and MSGPRITY = URGENT
	SLRNRM02	APDU = APDU_01 and APPL = FIN and MSGPRITY = NORMAL
	MBMERROR	ALWAYS >= ROUTE
	MBMERROR	Otherwise
MLCNTRL	MBDELETE	
MLAKWAIT	MBDELETE	
TP2_ACK	MBMERROR	
TP2_WAIT	TP2_ACK	ACKTELEX = TX_ACK
	TP2_NAK	ACKTELEX = TX_NAK
	MBMERROR	Otherwise

Table 26. Standard Routing Conditions for MERVA USE & Branch for Windows NT using SWIFT Link (continued)

Messages in queue...	Are routed to...	When...
TP2_NAK	TP2_SND	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBDELETE	MSGOK = DELETE
	MBMERROR	Otherwise
TP2_RCV	MBMERROR	
TP2_SND	TP2_WAIT	MSGOK = OK
	MBMERROR	Otherwise
UFROMSWF	UWOPREAG	MSGINFO = NO_PREAG or MSGINFO = INC_966
	SLPRINT2	Otherwise
UNAKED	SLPRINT2	
UWOPREAG	UFROMSWF	
USEINCMD	MBMERROR	
USESEND	MBMERROR	MSGTYPE = MTYP_999
	SLRURG02	MSGGROUP = MTYP_96X and MSGPRITY = URGENT
	SLRNRM02	MSGGROUP = MTYP_96X and MSGPRITY = NORMAL
	SLRSYS02	MSGTYPE = MTYP_075
	SLRSYS01	MSGTYPE = MTYP_074 or MSGTYPE = MTYP_085 or MSGTYPE = MTYP_090
	MBMERROR	Otherwise

Routing Conditions for MERVA USE & Branch for Windows NT using MERVA Link

The standard routing for MERVA USE & Branch for Windows NT differs from the MERVA standard routing in the following aspects:

- USE messages are received from and sent to MERVA Link instead of SWIFT Link.
- MERVA-internal USE commands are received from and sent to MERVA Link.
- The SWIFT Link routing is replaced by MERVA Link. The remote front-end application scenario is used when SWIFT Link acknowledgments are processed by MERVA Link. For a detailed description on this scenario, refer to “MERVA Link Remote Front-End Scenario” on page 67.
- Telex messages received from MERVA Link are sent to Telex send queues.
- Telex messages received from the Telex network are sent to MERVA Link.

To import the standard routing for MERVA Link that is delivered with MERVA USE & Branch for Windows NT:

1. Start MERVA USE & Branch for Windows NT in customization mode.
2. Change to the directory <ENM_PATH>\Samples
3. Enter


```
enmcximp -f enmcxuab -c 1
```

This loads the file **enmcxuab.rou**, which is located in the directory from step 2.

For more information about the import utility, refer to “Chapter 10. Exporting and Importing Customization Data” on page 89.

Table 27 lists the routing conditions supplied with MERVA USE & Branch for Windows NT using MERVA Link.

Table 27. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA Link

Messages in queue...	Are routed to...	When...
MBDELETE	MBDELETE	
MBMERROR	MBMERROR	
MBCRCERR	MBCRCERR	
MBRERROR	MBRERROR	
SMCREATE	MLSNDURG	MSGOK = OK and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = CANCEL
	MBMERROR	Otherwise
SMINCMPT	MLSNDURG	MSGOK = OK and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = DELETE or MSGOK = CANCEL
	MBMERROR	Otherwise
SMVERIFY	SMAUTH1	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMAUTH1	MLSNDNRM	MSGOK = OK and MSGNEDIT = SWIFNET and MSGPRITY = URGENT
	MLSNDLOW	MSGOK = OK and MSGNEDIT = SWIFNET and MSGPRITY = NORMAL
	TP2_SND	MSGOK = OK and MSGNEDIT = TELEXNET
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMEDIT	SMVERIFY	MSGOK = OK
	SMEDIT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = DELETE
	MBMERROR	Otherwise
SLPRINT1	MBDELETE	
SLPRINT2	MBDELETE	

Table 27. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA Link (continued)

Messages in queue...	Are routed to...	When...
SLRCVFIN	MBMERROR	
SLRCVNST	MBMERROR	
SLRCVSY	MBMERROR	
SLRL1ACK	MBDELETE	
MLMERROR	MBMERROR	
MLAKWAIT	SLPRINT2	MSGCAT = MTYP_SYS
	SLRL1ACK	ACKINFO = ACK_ACC
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SMEDIT	ACKINFO = ACK_REJ or MSGINFO = AUTINKEY or MSGINFO = AUTNOKEY
	SMEDIT	MSGINFO = AUTNOTRL or MSGINFO = AUTTOOLG
	MBMERROR	Otherwise
MLRECEIV	MLCNTRL	ALWAYS >= ROUTE and continue routing
	TP2_SND	MSGNETID = TELEXNET
	USEINCMD	MSGNETID = SWIFNET and MSGTYPE = MTYP_999
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_076
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_087
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_092
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_094
	UFROMSWF	MSGNETID = SWIFNET and MSGGROUP = MTYP_96X
	SLRCVSY	MSGTYPE = MTYP_010 or MSGTYPE = MTYP_011 or MSGTYPE = MTYP_015
	SLRCVSY	MSGTYPE = MTYP_066 or MSGTYPE = MTYP_082 or MSGTYPE = MTYP_083
	SLRCVSY	MSGTYPE = MTYP_021 and EMB_TYP1 <> MTYP_SYS
	SLPRINT2	MSGCAT = MTYP_SYS
	SLRCVNST	MSGTYPE = MTYP_19X or MSGTYPE = MTYP_29X
	SLRCVFIN	ALWAYS >= ROUTE
	MBMERROR	Otherwise

Table 27. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA Link (continued)

Messages in queue...	Are routed to...	When...
MLSNDURG	MBDELETE	MSGTYPE = MTYP_999
	MLAKWAIT	MSGNETID = SWIFTNET
	MBMERROR	Otherwise
MLSNDNRM	MLAKWAIT	MSGNETID = SWIFTNET
	MBDELETE	MSGNETID = TELEXNET
	MBMERROR	Otherwise
MLSNDLOW	MLAKWAIT	MSGNETID = SWIFTNET
	MBMERROR	Otherwise
MLCNTRL	MBDELETE	
TP2_NAK	TP2_SND	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBDELETE	MSGOK = DELETE
	MBMERROR	Otherwise
TP2_RCV	MLSNDNRM	
TP2_SND	TP2_WAIT	MSGOK = OK
	MBMERROR	Otherwise
TP2_WAIT	MLSNDNRM	ACKTELEX = TX_ACK
	MLSNDNRM	ACKTELEX = TX_NAK
	MBMERROR	Otherwise
UFROMSWF	UWOPREAG	MSGINFO = NO_PREAG or MSGINFO = INC_966
	SLPRINT2	Otherwise
UNAKED	SLPRINT2	
USEINCMD	MBDELETE	
USESEND	MLSNDURG	MSGTYPE = MTYP_999
	MLSNDNRM	Otherwise
UWOPREAG	UFROMSWF	

Routing Conditions for MERVA USE & Branch for Windows NT using MERVA-MQI Attachment

The standard routing for MERVA USE & Branch for Windows NT using MERVA-MQI Attachment differs from the MERVA standard routing in the following aspects:

- USE messages are received from and sent to MERVA-MQI Attachment instead of SWIFT Link.
- MERVA-internal USE commands are received from and sent to MERVA-MQI Attachment.
- The SWIFT Link routing is replaced by MERVA-MQI Attachment.

To import the standard routing for MERVA-MQI Attachment that is delivered with MERVA USE & Branch for Windows NT:

1. Start MERVA USE & Branch for Windows NT in customization mode.

2. Change to the directory <ENM_PATH>\Samples\USE_MQI
3. Enter

```
enmcximp -f enmcxmqa -c 1
```

This loads the file **enmcxmqa.rou**, which is located in located in the directory from step 2.

For more information about the import utility, refer to “Chapter 10. Exporting and Importing Customization Data” on page 89.

Table 28 lists the routing conditions supplied with MERVA USE & Branch for Windows NT using MERVA-MQI Attachment.

Table 28. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA-MQI Attachment

Messages in queue...	Are routed to...	When...
MBDELETE	MBDELETE	
MBMERROR	MBMERROR	
MBCRCERR	MBCRCERR	
MBRERROR	MBRERROR	
SMCREATE	MQSND1	MSGOK = OK and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = CANCEL
	MBMERROR	Otherwise
SMINCMPT	MQSND1	MSGOK = OK and MSGCAT = MTYP_SYS
	SMVERIFY	MSGOK = OK
	SMINCMPT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = DELETE or MSGOK = CANCEL
	MBMERROR	Otherwise
SMVERIFY	SMAUTH1	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMAUTH1	MQSND1	MSGOK = OK and MSGNETID = SWIFNET
	TP2_SND	MSGOK = OK and MSGNETID = TELEXNET
	SMEDIT	MSGOK = FAILED
	MBMERROR	Otherwise
SMEDIT	SMVERIFY	MSGOK = OK
	SMEDIT	MSGOK = INCOMPLT
	MBDELETE	MSGOK = DELETE
	MBMERROR	Otherwise
SLPRINT1	MBDELETE	
SLPRINT2	MBDELETE	

Table 28. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA-MQI Attachment (continued)

Messages in queue...	Are routed to...	When...
SLRCVFIN	MBMERROR	
SLRCVNST	MBMERROR	
SLRCVSY	MBMERROR	
SLRL1ACK	MBDELETE	
MQRCV	USEINCMD	MSGNETID = SWIFNET and MSGTYPE = MTYP_999
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_076
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_087
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_092
	UFROMSWF	MSGNETID = SWIFNET and MSGTYPE = MTYP_094
	UFROMSWF	MSGNETID = SWIFNET and MSGGROUP = MTYP_96X
	SLRCVSY	MSGTYPE = MTYP_010 or MSGTYPE = MTYP_011 or MSGTYPE = MTYP_015
	SLRCVSY	MSGTYPE = MTYP_066 or MSGTYPE = MTYP_082 or MSGTYPE = MTYP_083
	SLRCVSY	MSGTYPE = MTYP_021 and EMB_TYP1 <> MTYP_SYS
	SLPRINT2	MSGCAT = MTYP_SYS
	SLRCVNST	MSGTYPE = MTYP_19X or MSGTYPE = MTYP_29X
	SLRCVFIN	ALWAYS >= ROUTE
	MBMERROR	Otherwise
MQSND1	MQWAIT	MSGOK = "MVQWAIT"
	MBDELETE	Otherwise
MQSND2	MQWAIT	MSGOK = "MVQWAIT"
	MBDELETE	Otherwise

Table 28. Standard Routing Conditions for MERVA USE & Branch for Windows NT using MERVA-MQI Attachment (continued)

Messages in queue...	Are routed to...	When...
MQWAIT	MBDELETE	MSGTYPE = MTYP_999
	SLRL1ACK	ACKINFO = ACK_ACC
	UNAKED	ACKINFO = ACK_REJ and MSGGROUP = MTYP_96X
	UNAKED	ACKINFO = ACK_REJ and MSGTYPE = MTYP_075
	SMEDIT	ACKINFO = ACK_REJ or ACKINFO = AUTNOTRL or MSGINFO = AUTINKEY
	SMEDIT	MSGINFO = AUTNOKEY or MSGINFO = AUTTOOLG
	SLPRINT2	MSGCAT = MTYP_SYS
	MBMERROR	Otherwise
TP2_NAK	TP2_SND	MSGOK = OK
	SMEDIT	MSGOK = FAILED
	MBDELETE	MSGOK = DELETE
TP2_RCV	SLPRINT2	
TP2_SND	TP2_WAIT	MSGOK = OK
	MBMERROR	Otherwise
TP2_WAIT	SLPRINT2	ACKTELEX = TX_ACK
	TP2_NAK	ACKTELEX = TX_NAK
	MBMERROR	Otherwise
UFROMSWF	UWOPREAG	MSGINFO = NO_PREAG or MSGINFO = INC_966
	SLPRINT2	Otherwise
UNAKED	SLPRINT2	
USEINCMD	MBDELETE	
USESEND	MQSND2	MSGTYPE = MTYP_999
	MQSND1	Otherwise
UWOPREAG	UFROMSWF	

Routing Descriptions for Each Source Queue

The following section describes message queues that are supplied with the standard routing.

MBCRCERR: Messages That Failed CRC Checking

MERVA verifies the contents of messages by using a check sum on the message contents. This is called a Cyclic Redundancy Check (CRC).

This queue contains messages that failed a CRC test. MERVA therefore rejects the messages. It assumes that these messages are corrupted or changed outside MERVA.

The messages are not routed to this queue but directly sent.

MBDELETE: Messages To Be Deleted

This queue does not contain messages because the system deletes any messages that are routed to this queue. Note that you cannot route a message to MBDELETE and to another queue at the same time.

Note: Messages that are sent to this queue are deleted.

MBMERROR: Messages That Failed Routing

This queue contains messages that are not routed as expected. To achieve this, you must define this queue as the default queue in routing conditions. Messages are sent to this queue when there is a logical *hole* in the routing, or when the contents of the message to be routed are not as expected. If all defined routing conditions fail, the message is routed to this queue. You can access these messages with the Message Retrieval functions.

MBRERROR: Messages That Caused an Internal Routing Failure

This queue contains messages for which MERVA detects an internal problem during processing. The message does not necessarily have to be wrong.

Examine the routing trace in the diagnosis log or contact your IBM representative.

MLAKWAIT: Messages to Be Correlated with Incoming Status Reports

This queue is only used for the remote front-end application scenario described in “MERVA Link Remote Front-End Scenario” on page 67. Messages to be correlated with incoming Status Reports must be routed to this queue from the MERVA Link send queues. This queue must also be defined as the Ack Wait queue in the MERVA Link partner definitions for the partner from which the appropriate Status Reports are to be received. You can use this queue as the Ack Wait queue for all partner definitions or define a new one for each partner.

This queue is not used in the MERVA sample routing. You must change the routing if you want to keep correlated messages.

Contributing Queue:

- MLSEND

MLCNTRL: Message Control

When MERV Link receives a message, the message is treated as described in the queue description. The routing into MLCNTRL is required at restart to determine whether the processing of a message was terminated before the previous abnormal end.

No routing is performed on this queue. Messages are deleted automatically by the receiving process of MERV Link. Ensure that each message received by MERV Link is routed to this queue. Refer to the sample routing of the MLRECEIV queue. You cannot create another MERV Link control queue.

Contributing Queue:

- MLRECEIV

MLMERROR: Messages That Failed Normal Processing

This queue contains messages where MERV Link has detected an internal problem during processing. Examine the logs for further explanation.

You can access these messages with the Message Retrieval functions.

MLRECEIV: Messages Received from Another MERV Link

This is a virtual queue. When MERV Link receives messages from a partner MERV Link, the messages are routed as if they are currently held in this queue. You can create a receive queue for each partner or use the same one for all partners. The relationship between the partner and the receive queue is defined in the MERV Link partner definitions.

Every message that is routed from the MERV Link receive queue is duplicated. The first copy is always routed to the control queue MLCNTRL for message integrity checking, using the statement ALWAYS>=ROUTE. The specified condition is always true. Do not change this entry. Add this entry to the routing of each further MERV Link receive queue that you define.

The second copy is routed to a destination that is determined by the network identifier and the type of message.

All other messages are routed to MBMERROR because the MERV Link sample routing only prepares your complete routing. You must define the queue to which you want to route these messages received by MERV Link.

The last condition, ALWAYS>=ROUTE, is always true. It is needed to route the message to MBMERROR when none of the previous conditions, except the first one, is true. This condition replaces the default routing. The default routing fails if the first condition is always true.

Contributing Queues:

- None

MLSEND: Messages to Be Sent to MERV Link

You can use the routing statements for this queue as a skeleton when you configure the routing to use MERV Link.

The first routing statement, MSGACK IS EMPTY, is for messages in which the MSGACK field is empty. This means that the message is not an acknowledged

message. It is a received message or a message created on the local system and sent to another system with MERVA Link. It is forwarded to the MLAKWAIT queue to wait for a status report from the partner system. If you do not implement a remote front-end application scenario for SWIFT Link, you can delete this condition. For the remote front-end application scenario, this condition is only needed for the send queues of the creating system.

If you do not delete the first conditions in the sample routing, the MSGACK field of the message is filled. In this case, it can be a message that was not sent as an application message but for which MERVA Link sent a status report. This depends on the setting of the status report switch in the MERVA Link partner definitions. It also depends on whether the message was previously received by MERVA Link. The message is routed to the MBDELETE queue because it is assumed that it is no longer needed in this system. You can change the setting according to your needs.

Contributing Queues:

- None

MQRCV: Messages Received from MQSeries

This queue contains

- USE messages received from SWIFT via SWIFT Link on another MERVA system and via MQSeries®
- Free-format messages (MT999) from other MERVA systems that are to be routed to the USE incoming commands queue of the USE Background process.

Examples of such messages are:

- Session key requests
- BK Update messages
- Pre-agreement update messages

Contributing Queues:

- None

MQSND1: Messages to Be Sent to MQSeries Awaiting SWIFT ACK or NAK

This queue contains SWIFT USE messages (for example MT96X and MT075) as well as SWIFT System and FIN messages created on the workstation. Messages from this queue are routed to the MQWAIT queue, where they wait for MQSeries reports (COA or COD), SWIFT messages (ACK or NAK), or both.

Contributing Queues:

- SMAUTH1
- SMCREATE
- SMINCMPT
- USESEND

MQSND2: Messages to Be Sent to MQSeries

This queue contains free-format messages (MT999) that are created by the USE Background process. These messages include:

- Session key responses
- BK update messages
- Responses to pre-agreement update messages from other MERVA systems

Contributing Queues:

- USESEND

MQWAIT: Messages awaiting SWIFT ACK or NAK information

Each message created by MERVA USE & Branch for Windows NT that is supposed to be sent to the remote application, and that is waiting for a response (MQSeries COA or COD reports, or SWIFT ACK or NAK message, or both), is routed to the wait queue (MQWAIT). A message is routed from the wait queue only if all required responses have been received.

Contributing Queues:

- MQSND1
- MQSND2

SL_IN: SWIFT Messages Loaded at the API

This queue is set up for application programs that load SWIFT messages to MERVA. The application program should use it as a virtual queue to route messages directly to the defined target queue. Messages are routed depending on their priority.

Contributing Queues:

- None

SLINCMS1: Incoming GPA Messages

This is a virtual queue. Incoming USE system messages (MT 092 and MT 094) are routed to the UFROMSWF queue for processing by the USE Background Process. Other incoming system messages are routed directly from this queue to the SLPRINT1 or SLPRINT2 queue to be printed.

SLINCMS2: Incoming FIN Messages

This is a virtual queue. All system acknowledgment messages are sent to the SLPRINT1 queue to be printed.

Incoming USE messages (MT 96x, MT 087, MT 076) are routed to the UFROMSWF queue for processing by the USE Background Process.

Messages of type 010, 011, 015, 066, 082, or 083 and retrieved messages are sent to the Distribution for Delivery or Non-Delivery Information queue (SLRCVSY) for unloading.

All remaining system messages are routed to the SLPRINT2 queue.

Messages that have failed authentication are sent to the SLMANAUT queue for manual authentication.

The remaining messages are non-system FIN messages that are authenticated correctly or that do not require authentication. All messages of group 19x or 29x are routed to the SLRCVNST queue. The others are routed to the SLRCVFIN queue.

SLMANAUT: Manual Authentication

This queue contains incorrect messages received from SWIFT. The following MERVA error message codes are shown in the MSGACK field of messages in this queue if they fail automatic authentication and require manual authentication:

ENM9978 Authentication routine detected invalid key
ENM9979 Key for Authentication not found
ENM9985 Authentication failed

You can look at these messages with the Manual Authentication function and correct the authenticator key as indicated by the error message. You can then reroute the message as an incoming message or print it.

Contributing Queue:

- SLINCMS2

SLPRINT1: Messages to Be Printed

This queue contains operator messages, such as positive and negative login, select acknowledgments, and logout, quit, and abort acknowledgments.

Messages are printed by the Automatic Print function.

SLPRINT2: Messages to Be Printed

This queue contains messages that are no operator messages and that do not belong to one of the following message types:

MT010 Non-delivery Warning: Message not delivered before expiration
MT011 Delivery Notification: Message delivered
MT015 Delayed ACK: Syntactically valid request not fulfilled
MT066 Undelivered Report: Undelivered messages
MT082 Timed Undelivered Report: Messages undelivered at time
MT083 Cut-off Undelivered Report: Messages undelivered at cut-off time.

All received and processed, or sent and negatively acknowledged USE messages are also routed to this queue.

Messages are printed by the Automatic Print function.

SLRCVFIN: Message Received OK (Other Than 19x and 29x)

This queue contains messages successfully received from SWIFT with message types not in the ranges 190 to 199 and 290 to 299, inclusive.

These messages are available at the API for unloading.

Contributing Queue:

- SLINCMS2

SLRCVNST: Message Received OK (Message Types 19x and 29x)

This queue contains messages successfully received from SWIFT with message types in the ranges 190 to 199 and 290 to 299, inclusive.

These messages are available at the API for unloading.

Contributing Queue:

- SLINCMS2

SLRCVSY: Distribution of Delivery or Non-Delivery Information

The queue contains messages received from SWIFT with the following types:

MT010	Non-delivery Warning: message not delivered before expiration
MT011	Delivery Notification: message delivered
MT015	Delayed ACK: syntactically valid request not fulfilled
MT066	Undelivered Report: undelivered messages
MT082	Timed Undelivered Report: messages undelivered at time
MT083	Cut-off Undelivered Report: messages undelivered at cut-off time

Messages are available at the API for unload.

Contributing Queue:

- SLINCMS2

SLRL1ACK: Acknowledged Messages

This queue contains messages previously sent to SWIFT and acknowledged positively. These messages can be retrieved by an application program for status control.

Contributing Queues:

- SLRNRM02
- SLRURG02

SLRNRM02, SLRURG02: Ready-to-Send FIN Normal/Urgent Messages

SLRNRM02 contains normal, SLRURG02 contains urgent priority FIN messages to be transmitted to the SWIFT network. Messages are held in these queues until the acknowledgment is received from SWIFT

If the acknowledgment is positive, the messages are sent to the SLRL1ACK queue.

If the acknowledgment is not positive, the messages are sent to the SMEDIT queue for correction. Some NAKed USE messages are sent to the UNAKED queue for processing by the USE Background Process.

Contributing Queue:

- SL_IN

SLRSYS01: Ready-to-Send GPA System Messages

This queue contains GPA messages to be transmitted to the SWIFT network. Messages are held in this queue until the acknowledgment information is received from SWIFT They are then sent to SLPRINT1 or SLPRINT2 to be printed.

Contributing Queues:

- SMCREATE
- SL_IN

SLRSYS02: Ready-to-Send FIN System Messages

This queue contains FIN system messages to be transmitted to the SWIFT network. Messages are held in this queue until the acknowledgment information is received from SWIFT

Other messages are then sent to SLPRINT1 or SLPRINT2 to be printed.

Contributing Queues:

- SMCREATE
- SL_IN

SMAUTH1: Messages Being Authorized

This queue contains messages to be authorized. When you use the Authorize Messages function, you access messages that reside in this queue.

The routing destination is determined by the action performed to exit the function and the priority field in a correct message:

- ACCEPT

This sets the field MSGOK in the message to **OK**. When the message is routed, it is forwarded, depending on the priority of the message, to SLRURG02 (urgent SWIFT) or SLRNRM02 (normal SWIFT).

- REJECT

This sets the field MSGOK in the message to **FAILED**. When the message is routed, it is forwarded to SMEDIT to be corrected later.

If all routing fails, the message is sent to the MBMERROR error queue for later analysis.

Contributing Queue:

- SMVERIFY

SMCREATE: Newly Created Messages

This is a virtual queue. Messages created by the Create Messages function are routed from this queue.

The routing destination is determined by the action performed to exit the function:

- ACCEPT

This sets the field MSGOK in the message to **OK**. When the message is routed, it is forwarded to the SMVERIFY queue to be verified.

- SAVE

This sets the field MSGOK in the message to **INCOMPLT**. When the message is routed, it is forwarded to SMINCMPT to be completed later.

- CLOSE

This sets the field MSGOK in the message to **CANCEL**. When the message is routed, it is forwarded to MBDELETE and deleted.

If all routing fails, the message is sent to the MBMERROR error queue for later analysis.

Contributing Queues:

- None

SMEDIT: Messages to Be Corrected

This queue contains messages to be corrected. Messages are sent to this queue when they fail verification or authorization. You can try to correct the message with the Edit Messages function. If the message is corrected successfully, it is returned to the verification queue.

The routing destination is determined by the action performed to exit the function:

- **ACCEPT**
This sets the field MSGOK in the message to **OK**. When the message is routed, it is sent to the verification queue.
- **SAVE**
This sets the field MSGOK in the message to **INCOMPLT**. When the message is routed, it is forwarded to SMINCMPT to be completed later.
- **DELETE**
This sets the field MSGOK in the message to **DELETE**. When the message is routed, it is forwarded to MBDELETE and deleted.

If all routing fails, the message is sent to the MBMERROR error queue for later analysis.

Contributing Queues:

- SMVERIFY
- SMAUTH1
- SLRURG02
- SLRNRM02
- TP2_NAK

SMINCMPT: Newly Created Messages and Messages Created But Incomplete

This is a virtual queue for messages that are newly created and routed from this queue.

This queue contains new messages that are incomplete. You can access an incomplete message by selecting it from the **Message List View**.

The routing destination is determined by the action performed to exit the Create Messages function:

- **SAVE**
This sets the field MSGOK in the message to **INCOMPLT**. When the message is routed, it is returned to SMINCMPT to be completed later.
- **ACCEPT**
This sets the field MSGOK in the message to **OK**. When the message is routed, it is forwarded to the SMVERIFY queue to be verified.
- **DELETE**
This sets the field MSGOK in the message to **DELETE**. When the message is routed, it is forwarded to MBDELETE and deleted.
- **CLOSE**
This sets the field MSGOK in the message to **CANCEL**. When the message is routed, it is forwarded to MBDELETE and deleted.

If all routing fails, the message is sent to the MBMERROR error queue for later analysis.

Contributing Queue:

- SMCREATE

SMVERIFY: Messages to Be Verified by Retyping

This queue contains messages waiting for retype verification. When you use the Retype Verify Messages function, you access messages that reside in this queue.

The routing destination is determined by the action performed to exit the Retype Verify Messages function:

- ACCEPT

This sets the field MSGOK in the message to **OK**. When the message is routed, it is forwarded to SMAUTH1 to be authorized.

- REJECT

This sets the field MSGOK in the message to **FAILED**. When the message is routed, it is forwarded to SMEDIT to be corrected later.

If all routing fails, the message is sent to the MBMERROR error queue for later analysis.

Contributing Queues:

- SMEDIT
- SMINCMPT
- SMCREATE

TP2_ACK: Positively Acknowledged Telex Messages

This queue contains messages that have been sent to the telex network and have been positively acknowledged.

Contributing Queue:

- TP2_WAIT

TP2_NAK: Negatively Acknowledged Telex Messages

This queue contains messages that have been sent to the telex network and have been negatively acknowledged.

You can work with these messages using the Queue list. Select:

- **Delete** to delete the message.

- **Move** to move the message to another queue, for example, to:

TP2_SND

To retry the telex transmission without changing any telex envelope data.

SMEDIT

To change telex envelope data before retrying the telex transmission.

Contributing Queue:

- TP2_WAIT

TP2_RCV: Received Telex Messages

This queue contains received telex messages.

Contributing Queues:

- None

TP2_SND: Ready-to-Send Telex Messages

This queue contains messages that are to be transmitted to the telex network. They are sent to TP2_WAIT to wait for acknowledgment.

Contributing Queues:

- SLRNRM02
- SLRURG02
- SMAUTH1
- TP2_NAK

TP2_WAIT: Telex Messages Waiting for Acknowledgment

This queue contains messages that are currently transmitted to the telex network. Messages are held in this queue until the acknowledgment is received from the telex network.

If the acknowledgment is positive, the messages are sent to the TP2_ACK queue.

If the acknowledgment is negative, the messages are sent to the TP2_NAK queue for further processing.

Contributing Queue:

- TP2_SND

UFROMSWF: USE Messages from SWIFT

This queue contains USE messages (MT 96x, MT 076, MT 087, MT 092, MT 094) that are received from the SWIFT network. The messages are read by the USE Background Process, for example, to update the status of the corresponding bilateral key exchange.

Completed new bilateral keys are stored in the database and sent for distribution in a free-format message if this is specified in the corresponding pre-agreement.

BK Initiation Request Messages (MT 960) for which no pre-agreement exists in the bilateral key database are marked with the constant NO_PREAG in the MSGINFO field and routed to the UWOPREAG queue for processing by the user key management officer (UKMO). The UKMO must decide whether to continue or discontinue the BKE process. This decision is indicated in the MSGOK field of the message (OK or CANCEL) when the message is routed back to the UFROMSWF queue. The USE Background Process then processes the message as a normal incoming MT 960, or sends a BKE Discontinuation Request (MT 966) message to the correspondent.

BK Discontinuation Requests (MT 966) are marked with the constant INC_966 in the MSGINFO field and routed to the UWOPREAG queue for processing by the user key management officer (UKMO). The UKMO must decide when to process the request. The message is routed back to the UFROMSWF queue.

After processing, the messages are routed to the SLPRINT2 queue.

You can create a number of queues for this purpose group, for example, to separate processing of USE messages for different home destinations.

Contributing Queues:

- SLINCMS1

- SLINCMS2
- UWOPREAG

UNAKED: Negatively Acknowledged USE Messages

This queue contains USE messages (MT 96x, MT 075, MT 085) that are sent to the SWIFT network and acknowledged negatively. The messages are read by the USE Background Process to update the status of the corresponding bilateral key exchange. After processing, the messages are routed to the SLPRINT2 queue.

Contributing Queues:

- SLRURG02
- SLRNRM02

USEINCMD: Incoming USE Commands

This queue is for free-format messages (MT 999) that are already created by another MERVA system for MERVA-internal USE processing. These messages include:

- Session key requests
- BK update messages
- Pre-agreement requests and updates

The messages are processed by the USE Background Process.

USESEND: USE Messages to Send

This is a virtual queue. It is used as the starting point for the routing of all USE messages from the USOF and UKMO functions and the USE Background Process.

All messages except the MT 999 are routed to a SWIFT Link ready-to-send queue. Usually, the MT 999 is used to communicate with MERVA ESA, for example, to distribute a new bilateral key. If your system creates USE-related MT 999 messages, change the target queue MBMERROR in the first routing statement according to your needs.

If you want to send MT 999 USE-related messages through MERVA Link, ensure that messages are not routed to MLAKWAIT from the MERVA Link send queue because these messages are never acknowledged.

UWOPREAG: Incoming MT 960 without Pre-Agreement

This queue contains BK Initiation Request Messages (MT 960) messages for which no pre-agreement exists in the bilateral key database. To process the messages use the Incoming MT 960 program in the MERVA Menu. The user must decide whether to continue or discontinue the BKE process. This decision is indicated in the MSGOK field of the message, and the message is routed back to the UFROMSWF queue for processing.

You can create a number of queues for this purpose group, for example, to separate processing of MT 960 messages for different home destinations.

Contributing Queues:

- UFROMSWF

Appendix C. 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 partner 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 DSLUNIT and DSLUEND 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 MERVAs 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 MERVAs Extended Connectivity for communication between MERVAs and MERVAs 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 MERVAs 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

- *DB2 Administration Guide*, S10J-8157
- *DB2 Building Applications for Windows and OS/2 Environment*, S10J-8160
- *DB2 API Reference*, S10J-8167
- *DB2 Troubleshooting Guide*, S10J-8169
- *eNetwork Personal Communications Version 4.2 for Windows 95 and Windows NT Quick Beginnings*, GC31-8476
- *eNetwork Personal Communications Version 4.2 for Windows 95 and Windows NT Reference*, GC31-8477
- *CID Enablement Guidelines*, S10H-9666
- *CICS-RACF Security Guide*, SC33-1185
- *ITSC Redbook APPC Security: MVS/ESA, CICS/ESA, and OS/2*, GG24-3960
- *IMS/ESA Version 4 Data Communication Administration Guide*, SC26-3060
- *MQSeries Application Programming Reference*, SC33-1673

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

A

Ack_Timeout_Period (SWIFT Link customization) 104
Ack wait queue (MERVA Link) 66
ACK_Wait_Queue (MERVA Link customization) 101
Activation_Date (SWIFT Link customization) 104
Activation_Time (SWIFT Link customization) 104
Actual_Value (routing customization) 98
Add_103 (SWIFT Link customization) 105
adding, administrator 20
adding, initial MERVA user 19
adding, other user 21
adding, user 19
address
 of emitting LT 76
 of Key Management Authority 76
 of S.W.I.F.T. support center 76
administrator, adding 20
Alarm_ (alarm customization) 95
Alarmname (alarm customization) 95
alarms
 customization (dialog) 83
 example ASCII file 95
 import input (check) 91
 keywords used in ASCII file 95
 linking to a message queue 83
 maintaining 83
Alarms (alarm customization) 95
All_Access_Rights (user access rights, customization) 107
Application (user access rights, customization) 107
Application_Support (MERVA Link customization) 100
application user exit (MERVA Link) 66
Application_User_Exit (MERVA Link customization) 101
application user exit flag 66
application user exits 145
ASCII (transfer format) 65
ASP (MERVA Link customization) 100
Ass_Timeout_Period (SWIFT Link customization) 104
Assigned_FIN_queues (SWIFT Link customization) 103
Assigned_GPA_queues (SWIFT Link customization) 103
Assigned_LT (user access rights, customization) 107
Associated Queue (user access rights, customization) 107
authentication 64
Authentication (MERVA Link customization) 101
Authentication_Mode (SWIFT Link customization) 104

authority
 End User Administration (USER) Authority 4
 System Administration (SYSADM) Authority 4
 User Administration (USERADM) Authority 4
automatic print
 customization (dialog) 81
 example ASCII file 96
 import input (check) 91
 keywords used in ASCII file 96
Automatic_Print (print customization) 96
automatic start 64
automatic start of MERVA Link queues 59
autoprt.nnn 82
Autostart (MERVA Link customization) 101
Autostart (print customization) 96

B

BKE_Start (SWIFT USE, customization) 110
Boolean Operators (routing customization) 43

C

Card_Management (SWIFT USE, customization) 110
card reader (USE), remote 78
Category (user access rights, customization) 108
Central_Institution (SWIFT Link customization) 104
Cert_exp_after (SWIFT USE, customization) 110
changes, saving 33
CID (SWIFT Link customization) 105
components (export/import) 91
Concatenation (routing customization) 99
Condition1 ... Condition4 (routing customization) 98
Connection_ID (SWIFT Link customization) 104
Constant (routing customization) 98
constants
 definition 35, 39
 keyword list 98
Continue (routing customization) 43
Copy_Service_ (SWIFT Link customization) 104
CRC 191
creating a MERVA instance 14
Currency (SWIFT Link customization) 105

Currency_Code (SWIFT Link customization) 105
currency code customization
 currency code, customize 84
 customizing, currency code 84
customization
 automatic start of MERVA Link queues 59
 export/import commands 89
 window dialog 31
customization report 33
cyclic redundancy check 191

D

database
 installation 15
Datatype (routing customization) 97
date format 85
Date_Range (SWIFT Link customization) 105
default printer 82
Default_Queue (routing customization) 98
Definitions (SWIFT USE, customization) 109
deinstalling MERVA USE & Branch 24
Delay (SWIFT Link customization) 104
Deliv_Notification (USE, customization) 109
Description (MERVA Link customization) 101
Destination (USE, customization) 109
destination checking 86
Details (user access rights, customization) 107
dial type 49
Dial_Type (SWIFT Link customization) 104
Duration (SWIFT Link customization) 104

E

EBCDIC (transfer format) 65
emitting LT 75
Emitting_LT (USE, customization) 109
encryption 64
Encryption (MERVA Link customization) 101
encryptor box 51
End Tag (routing customization) 38
End_Tag (routing customization) 97
End User Administration (USER) Authority 5
ENMConfirmed function 164
ENMCorrelateDefaultStatusReport function 154
enmcxexp command 89
enmcximp command 90

- ENMGenerateDefaultStatusReport function 153
- ENMGenerateSR function 163
- ENMGetField function 149
- ENMGetStatusRepData function 157
- ENMGetStatusRepHeader function 155
- ENMIncomingAM function 165
- ENMIncomingSR function 166
- ENMIsStatusRepDataValid function 159
- ENMOutgoing function 162
- ENMPutField function 150
- ENMPutStatusRepData function 158
- ENMPutStatusRepHeader function 156
- ENMReadyToSend function 161
- ENMTraceFields function 151
- ENMWriteToLogFile function 152
- export customization data 89
- extensions (export/import) 91

F

- field, message 37
- Field (routing customization) 97, 98
- FIELD (routing customization) 42
- Filter (user access rights, customization) 108
- FIN (financial application) 48
- FIN_Copy_Data (SWIFT Link customization) 104
- FIN Copy Service
 - control institution 51
 - defining 51
 - field, MSGINFO2 55
 - MSGINFO2 55
 - routing 55
- format of date 85
- front-end application 67
- Full_Copy_Flag (SWIFT Link customization) 105
- functions, user-exit interface 148
- Further_Routing (routing customization) 99

G

- general purpose application (GPA) 48
- Granted_Right (user access rights, customization) 107

H

- hardware requirement 5

I

- import customization data 89
- included_MSG_Field_ (SWIFT Link customization) 105
- initial Merva user, adding 19
- installation
 - MERVA database 15
 - overview 9
 - procedure 9
 - program file 10
 - verifying 22

- installing service 14
- instance 4

K

- Key_certification (SWIFT USE, customization) 110
- Key Management Authority address 76

L

- Length (routing customization) 38, 97
- license key, installing 16
- Line ASCII (transfer format) 65
- Line EBCDIC (transfer format) 65
- line type 50
- Line_Type (SWIFT Link customization) 103
- Link_Name (SWIFT Link customization) 104
- Live_Service_Flag (SWIFT Link customization) 104
- Local_ASP_Name (MERVA Link customization) 100
- Local_DTE (SWIFT Link customization) 104
- Local_MTP_Name (MERVA Link customization) 101
- Local_Node (MERVA Link customization) 100
- Local_Phone_Number (SWIFT Link customization) 104
- log level
 - for programmer's trace log 86
 - for queue trace log 86
 - for routing trace 86
- log period 86
- logging level 86
- logical terminals, queue assignment 48
- Logical_Terminals (SWIFT Link customization) 103
- low send queue (MERVA Link) 64
- Low_Send_Queue (MERVA Link customization) 101
- LT_ (SWIFT Link customization) 103
- LT_Name (SWIFT Link customization) 103

M

- machine requirement 5
- Manual Authentication function 195
- master logical terminal (LT) 47
- MBCRCERR 191
- MBDELETE 191
- MBMERROR 191
- MBRERROR 191
- MERVA database, creating 15
- MERVA database, how to remove 24
- MERVA instance 4
 - creating 14
- MERVA instance, how to remove 25
- MERVA license key, installing 16
- MERVA Link
 - application user exits 145
 - ASP information in ASCII file 100

- MERVA Link (*continued*)
 - customization 59
 - example ASCII file 102
 - introduction to 57
 - keywords used in ASCII file 100
 - partner nodes in ASCII file 100
 - user-exit interface 148
- MERVA_Link (MERVA Link customization) 100
- MERVA Link definitions
 - Ack wait queue 66
 - Alternate 62
 - application user exit 66
 - application user exit flag 66
 - ASP name 63
 - authentication 64
 - automatic start 64
 - autostart 64
 - conversation security information 61, 62
 - description 63
 - encryption 64
 - gateway 62
 - host name 61
 - import input (check) 92
 - local MTP name 63
 - message audit logging enabled 66
 - MTP ASP name 63
 - partner ASP name 63
 - partner MTP name 63
 - partner node 61
 - partner node name 64
 - password 62
 - password encryption method 62
 - port number 61
 - preferred connection type 65
 - receive queue 65
 - security information 61
 - security key modifier 66
 - security user exit 66
 - send queues
 - (Urgent/Normal/Low) 64
 - SNA APPC information 61
 - symbolic destination 61
 - TCP/IP information 61
 - transaction program 61
 - transfer format 65
 - user ID 61
- MERVA program file, how to remove 25
- Message_Audit_Logging (MERVA Link customization) 101
- message buffer 67
- message fields
 - defining 37
 - definition 35
 - determining start position of 37
 - keyword list 97
 - predefined 37
- Message_Headers (USE, customization) 109
- message headers for USE 75
- Message Part (routing customization) 37
- Message_Part (routing customization) 97
- message print separator 82
- message queues 36
 - adding 36

- message queues (*continued*)
 - definition 35
 - deleting 36
 - keyword list 97
 - purpose group 36
 - setting up 36
- Message Reference Number (MRN) 4
- Message_Type_ (SWIFT Link customization) 105
- Message_Type (user access rights, customization) 108
- Message_Types (user access rights, customization) 107
- Messages_from_Queue (routing customization) 98
- MLAKWAIT 191
- MLCNTRL 192
- MLMERROR 192
- MLRECEIV 192
- MLSEND 192
- modems 51
- MQRCV 193
- MQSND1 193
- MQSND2 193
- MQWAIT 194
- MSGACK FIELD 55
- MSGTXT4 FIELD 174, 175

N

- NAKed (SWIFT USE, customization) 109
- Name (print customization) 96
- Name (routing customization) 97
- Name (SWIFT Link customization) 104, 105
- Name (USE, customization) 109
- Name (user access rights, customization) 108
- Network_Data (SWIFT Link customization) 103
- Nodename (MERVA Link customization) 100, 101
- normal send queue (MERVA Link) 64
- Normal_Send_Queue (MERVA Link customization) 101
- Notices 203

O

- Obsolesc_Period (SWIFT USE, customization) 109
- Offset (routing customization) 38, 97
- OPERAND (routing customization) 43
- Operator (routing customization) 98
- OPERATOR (routing customization) 42
- other user, adding 21

P

- Partner_ASP_Name (MERVA Link customization) 101
- Partner_MTP_Name (MERVA Link customization) 101
- Partner_Node (MERVA Link customization) 100

- Partner_Nodes (MERVA Link customization) 100
- Preferred_Connection (MERVA Link customization) 101
- preferred connection type 65
- Print_Queue_ (print customization) 96
- print separator 82
- Print_to_file (print customization) 96
- Printer (print customization) 96
- printing, automatic 81
- Priority (USE, customization) 109
- Processes (SWIFT USE, customization) 109
- program requirements 6
- programmer's trace log 86
- Protocol Data Unit (PDU) 49
- PSPDN 50
- PSTN 50, 51
- PTT 50
- Purpose_Group (routing customization) 97
- purpose groups
 - defining parameters for 36
 - definition 35

Q

- Queue (routing customization) 97
- Queue (user access rights, customization) 107
- queue assignment 48
- Queue Buffer (transfer format) 65
- Queue_Name (alarm customization) 95
- Queue_Name (routing customization) 98
- queue trace log 86
- queues, standard routing
 - MBCRCERR 191
 - MBDELETE 191
 - MBMERROR 191
 - MBRERROR 191
 - MLAKWAIT 191
 - MLCNTRL 192
 - MLMERROR 192
 - MLRECEIV 192
 - MLSEND 192
 - MQRCV 193
 - MQSND1 193
 - MQSND2 193
 - MQWAIT 194
 - SL_IN 194
 - SLINCMS1 194
 - SLINCMS2 194
 - SLMANAUT 194
 - SLPRINT1 195
 - SLPRINT2 195
 - SLRCVFIN 195
 - SLRCVNST 195
 - SLRCVSYS 196
 - SLRL1ACK 196
 - SLRNRM02 196
 - SLRSYS01 196
 - SLRSYS02 197
 - SLRURG02 196
 - SMAUTH1 197
 - SMCREATE 197
 - SMEDIT 198

- queues, standard routing (*continued*)
 - SMINCMPT 198
 - SMVERIFY 199
 - TP2_ACK 199
 - TP2_NAK 199
 - TP2_RCV 199
 - TP2_SND 200
 - TP2_WAIT 200
 - UFROMSWF 200
 - UNAKED 201
 - USEINCMD 201
 - USESEND 201
 - UWOPREAG 201

R

- Receive (SWIFT Link customization) 103
- receive queue (MERVA Link) 65
- Receive_Queue (MERVA Link customization) 101
- Received_Commands (SWIFT USE, customization) 109
- Received_Messages (SWIFT USE, customization) 109
- Remote_DTE (SWIFT Link customization) 104
- removing
 - MERVA database 24
 - MERVA instance 25
 - MERVA program file 25
- report generation 33
- requirements
 - hardware 5
 - software 6
- resynchronization 50
- Retries (SWIFT Link customization) 104
- Right_nnnn (user access rights, customization) 107
- routing
 - conditions, tests with 40
 - conditions in ASCII file 98
 - constants (dialog) 39
 - constants in ASCII file 98
 - customization (dialog) 36
 - example ASCII file 99
 - keywords used in ASCII file 97
 - message fields (dialog) 37
 - message fields in ASCII file 97
 - message queues (dialog) 36
 - message queues in ASCII file 97
 - standard routing using MERVA Link 184
 - standard routing using MERVA-MQI Attachment 187
 - standard routing using SWIFT Link 180
- Routing_Conditions (routing customization) 98
- routing parameters
 - import input (check) 91
 - message queues 36
 - predefined 36
- routing trace 86

S

- saving changes 33
- Scan (routing customization) 38, 97
- Security_Key_Modifier (MERVA Link customization) 101
- Security_User_Exit (MERVA Link customization) 101
- Send_Immediate (SWIFT Link customization) 103
- Send_Normal (SWIFT Link customization) 103
- send queues (MERVA Link) 64
- Send_System (SWIFT Link customization) 103
- Send_Urgent (SWIFT Link customization) 103
- service, installing 14
- service identifier 51
- SL_IN 194
- SLINCMS1 194
- SLINCMS2 194
- SLMANAUT 194
- SLPRINT1 195
- SLPRINT2 195
- SLRCVFIN 195
- SLRCVNST 195
- SLRCVSYS 196
- SLRL1ACK 196
- SLRNRM02 196
- SLRSYS01 196
- SLRSYS02 197
- SLRURG02 196
- SMAUTH1 197
- SMCREATE 197
- SMEDIT 198
- SMINCMPT 198
- SMVERIFY 199
- SNA_Symbolic_Destination (MERVA Link customization) 100
- SNA_TP_Name (MERVA Link customization) 100
- software requirements 6
- Source Queue (routing customization) 40
- Source_Queue (routing customization) 98
- standard routing
 - using MERVA Link 180, 184
 - using MERVA-MQI Attachment 187
- Start Tag (routing customization) 38
- Start_Tag (routing customization) 97
- State_of_Receiving_Process (MERVA Link customization) 101
- status report 66
- Support_Center (SWIFT USE, customization) 110
- Suspend_Desired (SWIFT Link customization) 104
- SWIFT line details 49
- SWIFT line timeouts 50
- SWIFT Link
 - copy service in ASCII file 104
 - example ASCII file 105
 - keywords used in ASCII file 103
 - message fields in ASCII file 105
 - network data in ASCII file 103
 - queue assignments in ASCII file 103

- SWIFT_Link (SWIFT Link customization) 103
- SWIFT Link definitions
 - add field 103 automatically 54
 - authentication mode 54
 - CID 53
 - connection types 50
 - copy service identifier 53
 - currency code 54
 - customization (dialog) 47
 - definition's activation 53
 - dial type 49
 - FIN (financial application) 48
 - FIN copy service 51
 - full copy flag 54
 - general purpose application (GPA) 48
 - import input (check) 92
 - line details 49
 - line type 50
 - live service flag 54
 - LT identifier 47
 - master LT 47
 - message types 54
 - network data 49
 - PDU Size 49
 - queue assignment 48
 - receive 48
 - resynchronization 50
 - send immediate 48
 - send normal 48
 - send system 48
 - send urgent 48
 - synonym LTs 48
 - timeouts 50
 - value date range 54
- SWIFT_USE (USE, customization) 109
- Symbolic_Address (SWIFT Link customization) 104
- Synonym_ (SWIFT Link customization) 103
- synonym logical terminal (LT) 48
- System Administration (SYSADM) Authority 4
- system configuration
 - date format 85
 - default printer 82
 - destination checking 86
 - logging level 86
 - message print separator 82

T

- Target_Queue_ (routing customization) 98
- Target Queue (routing customization) 41
- TCP/IP_Host_Name (MERVA Link customization) 100
- TCP/IP_Port_Number (MERVA Link customization) 100
- telex header, layout of 174
- telex information, layout of 175
- Timeout_for_BKE (SWIFT USE, customization) 109
- Timeout_for_Cert (SWIFT USE, customization) 109
- TP2_ACK 199

- TP2_NAK 199
- TP2_RCV 199
- TP2_SND 200
- TP2_WAIT 200
- TPDU_Size (SWIFT Link customization) 104
- trace log, queue 86
- transfer format 65
- Transfer_Format (MERVA Link customization) 101
- Transport Protocol Data Unit (TPDU) 49
- Type (routing customization) 39

U

- UFROMSWF 200
- UKMO 76
- UNAKED 201
- uninstalling MERVA USE & Branch 24
- urgent send queue (MERVA Link) 64
- Urgent_Send_Queue (MERVA Link customization) 101
- USE
 - background process 76
 - background processes in ASCII file 109
 - destinations 75
 - example ASCII file 110
 - import input (check) 93
 - keywords used in ASCII file 109
 - message headers 75
 - message types 77
 - start parameters 76
 - USE definitions in ASCII file 109
- USE card reader 78
- USEINCMD 201
- user, adding 19
- User (user access rights, customization) 107
- user access rights 91
 - example ASCII file 108
 - export output 91
 - import input (check) 92
 - keywords used in ASCII file 107
 - message types in ASCII file 107
 - queues in ASCII file 107
 - update rights in ASCII file 107
- User Administration (USERADM) Authority 4
- User_Exit_Flag_1 (MERVA Link customization) 101
- User_Exit_Flag_2 (MERVA Link customization) 102
- user-exit interface functions 148
- user exits, MERVA Link 145
- User_ID (MERVA Link customization) 100
- user privileges 36
- User_Rights (user access rights, customization) 107
- UserID (user access rights, customization) 107
- USESEND 201
- USOF 76
- UWOPREAG 201

V

- Value (routing customization) 98
- Value_Date (SWIFT Link customization) 105
- verifying, installation 22

X

- X.25 Adapter
 - adapter, X.25 137
 - card, co-processor 137
 - co-processor 137

Readers' Comments — We'd Like to Hear from You

MERVA ESA Components
MERVA USE & Branch for Windows NT
Installation and Customization Guide
Version 4 Release 1

Publication No. SH12-6335-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Complete	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to find	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to understand	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Well organized	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Applicable to your tasks	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

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.



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



Program Number: 5648-B30



Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and other countries.

SH12-6335-04

