GXS

Expedite Notification Manager for MVS Programming Guide

Version 1 Release 1

GC34-2327-01

Second Edition (November 2005)

This book is current as of its edition date. New editions will reflect changes in procedures or technical details..

Contents

To the reader

References in this document to your marketing representative refer to your local provider of GXS services.

About this book

This book describes Expedite Notification Manager and its functions.

Who should read this book

This book is intended for users who have a basic understanding of the Expedite and Information Exchange products, and the APPC/MVS environment.

Typographic conventions used in this book

To use Expedite Notification Manager, you need to understand the basic command syntax and how to use the profile commands.

The following is an example of Expedite Notification Manager command syntax:

```
command parameter(value) parameter(value) ...parameter(value);
```

- Descriptions of the syntax elements are:.

Syntax Element	Description
#	Defines or delimits a comment line. You can type any information you like after a #, and Expedite Notification Manager ignores the characters that follow it on the same line. If you include a # in a parameter value, Expedite Notification Manager knows the # is part of a command and does not ignore the parameter value of the characters that follow it.
Command	Identifies the Expedite Notification Manager command.
Parameter	Identifies a parameter of the associated command. ■ All parameter values are shown in italics. ■ Required parameter values are shown in boldface. ■ Default parameter values are underlined.
Value	Defines the value associated with the parameter.
...	In the previous example, the ellipsis (...) indicates that you can specify as many parameters as necessary. (It is not part of the syntax.)
;	Ends the command.

You can type Expedite Notification Manager commands and parameters in uppercase lowercase letters. The commands and parameters can span several lines in the Expedite Notification Manager profile. However, the following limitations apply:

- Type the entire command name, for example, **trace**, on a single input line.
- Type the entire parameter name, for example, **link,** on a single input line.
- Ensure that each parameter is immediately followed by a left parenthesis. Do not use spaces between parameter names and values. For example, type **link(y)** rather than **link (y)**.
- End each command with a semicolon.

What this book contains

- Chapter 1, "Introducing Expedite Notification Manager," gives an overview of Expedite Notification Manager..
- Chapter 2, "Setting up Expedite Notification Manager," provides a general explanation of how Expedite Notification Manager works, and describes the files you need to create.
- Chapter 3, "Expedite Notification Manager profile commands," explains how to create profiles, and describes the profile commands and profile response records.
- Chapter 4, "Receiving notification data," describes the format of information written to the Expedite Notification Manager notification log, and provides examples.
- Chapter 5, "Application examples," describes how you can write applications using the information supplied in the event notification
- Appendix A, "Expedite Notification Manager logs and traces," provides examples of Expedite Notification Manager logs and traces.
- Appendix B, "Messages and codes," provides the Expedite Notification Manager return codes with explanations and user actions

This book also contains a glossary and an index.

Related books

The following books contain information related to the topics covered in this book:

- *Expedite Base/MVS Programming Guide, GC34-2204*
- *z/OS V1R4.0 MVS Planning: APPC Management, SA22-7599*
- *Using Information Exchange Administration Services User's Guide, GC34-2221*

Chapter 1

Introducing Expedite Notification Manager

Expedite Notification Manager works with Information Exchange to provide event-driven notifications. An event-driven notification is when Information Exchange notifies Expedite Notification Manager that an event has occurred on Information Exchange. Based on your event profile in Information Exchange Administration Services, Information Exchange notifies Expedite Notification Manager using a notification that contains data about the event that occurred.

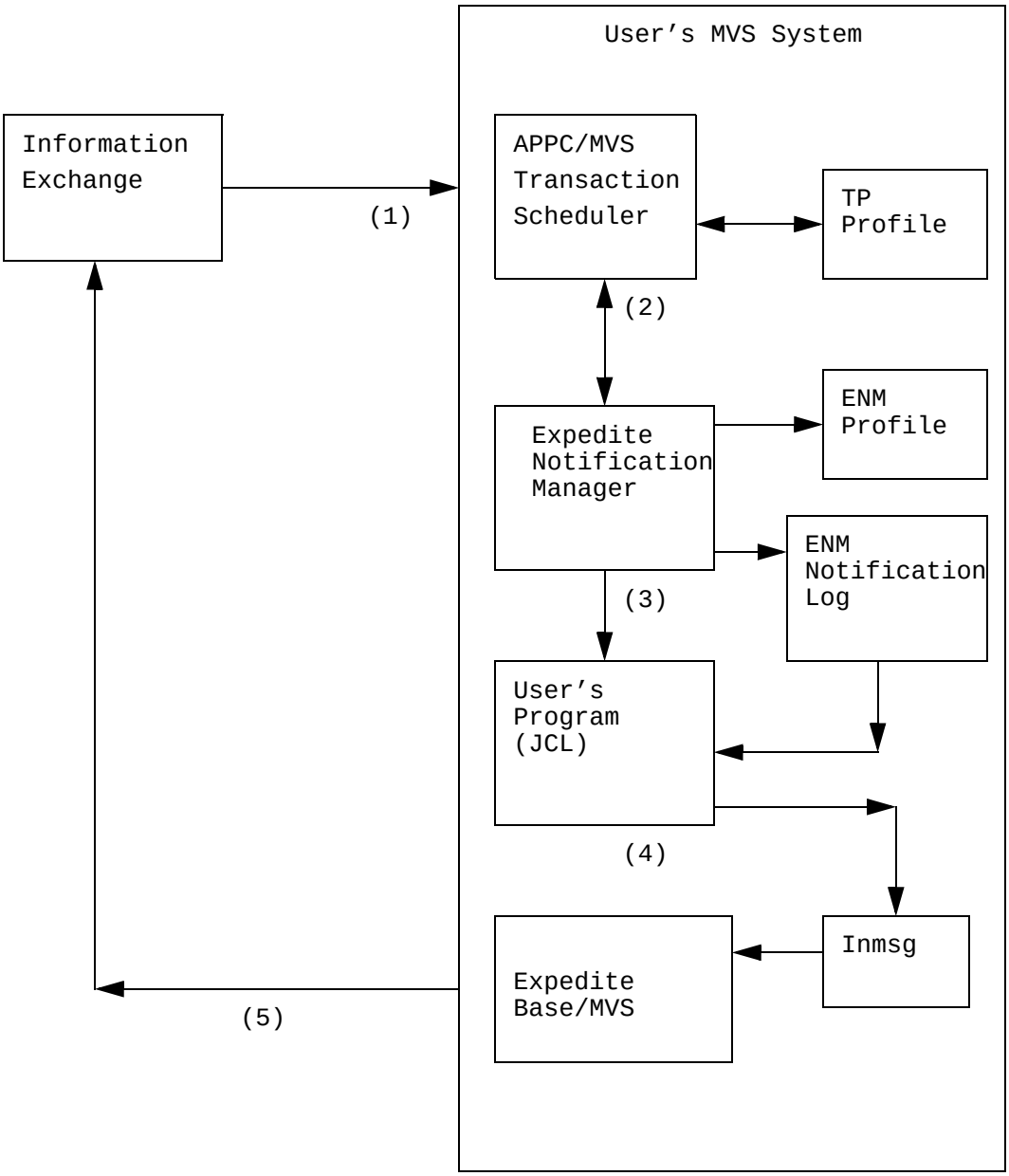
There are three types of events:

Message-arrival	A message-arrival event occurs when a message arrives at your Information Exchange mailbox. You can have notifications sent for every message, or you can create message-arrival definitions using Information Exchange Administration Services to limit the number of notifications sent to Expedite Notification Manager.
Scheduled	A scheduled event occurs based on a schedule defined in Information Exchange Administration Services. You can specify the date, time, and frequency of the schedule for sending notifications.
User-initated	A user-initiated event occurs when your Information Exchange administrator requests a notification to be sent.

When a notification is received, Expedite Notification Manager logs the notification in a file. Expedite Notification Manager can also start an application in response to the notification. This application can be Expedite Base/MVS, or a user-written application.

System diagram

The following is the system diagram for Expedite Notification Manager. This pictorial overview represents a complete transaction beginning with a message-arrival, schedule-based, or user-initiated event at Information Exchange and ending with the invocation of an application in your system by Expedite Notification Manager. In this case, the user application invokes Expedite Base/MVS to retrieve a mail item from Information Exchange. This process flow is repeated for each event triggered in Information Exchange.



1. Information Exchange Notification Manager allocates a conversation with Expedite Notification Manager when an event occurs.
2. APPC/MVS locates the process required to handle the request from the APPC/MVS TP profile. It submits the JCL stored in the APPC/MVS TP profile; in this case, the Expedite Notification Manager job.

3. Expedite Notification Manager receives the notification and, based on Expedite Notification Manager profile commands, writes the information to the Expedite Notification Manager notification log. If the USERAPPL field in the notification contained a value and the profile command file contained EXECUTE USERAPPL(1), Expedite Notification Manager opens the file and submits the program.
4. Optionally, your program can then process the Expedite Notification Manager notification log to produce input to Expedite, check the status of the last Expedite Base/MVS run, and submit Expedite Base/MVS.
5. Expedite Base/MVS receives the data from your Information Exchange mailbox.

Chapter 2

Setting up Expedite Notification Manager

This chapter gives you the information you need to set up Expedite Notification Manager before you begin to use it. It provides a general explanation of how Expedite Notification Manager works, describes the files you need to create and their attributes, or properties, and contains sample JCL statements.

How Expedite Notification Manager uses the notification log

Expedite Notification Manager is a file-oriented application. Expedite Notification Manager runs in response to an event occurring in Information Exchange. Information Exchange notifies the user that a pre-defined event has occurred. Expedite Notification Manager is started based on a notification received from Information Exchange.

Expedite Notification Manager receives the event notification and writes it to the notification log referenced by the ENMNTFY ddname.

Because Information Exchange events can occur in rapid succession, care should be taken when setting up Expedite Notification Manager on a user's system to prevent successive runs from overwriting an unprocessed notification.

File limitations

The following limitations apply to all files used with Expedite Notification Manager. These limitations apply in addition to any other limitations given for specific files.

- Use only sequential files and partitioned data set (PDS) members.
- Do not use virtual storage access method (VSAM) files.
- Do not specify DISP=MOD with PDS members.
- Use DISP=MOD with care when not otherwise restricted. If you use DISP=MOD, to avoid reprocessing, you must remove any processed data before running the same job again.
- Do not use spanned records in PDS members.

- Do not use RECFM=U.
- When creating these files, do not use a "numbers on" text-editing option. Create these files without line numbers.

Notification log description

The notification log that all Expedite Notification Manager users are required to create is described here. You might need to create other files as well, depending on how your application will interface with Expedite Notification Manager.

Notification log (ENMNTFY)

The ddname of the notification log is ENMNTFY. This file is used by Expedite Notification Manager to write the received event notification. It also contains a return code from Expedite Notification Manager processing of the event notification.

Several methods can be used in association with this file; only some possibilities are listed below.

- The file associated with ddname ENMNTFY could be allocated as a generation data group (GDG). Each time the Expedite Notification Manager job runs, a new generation is created. You must make sure that notifications are processed, if required, before the GDG reaches its limit and begins to discard older generations. For more information on how to define a GDG, refer to *DFSMS/MVS Access Method Services for the Integrated Catalog Facility*.
- The file associated with ddname ENMNTFY could be allocated as a new data set in the Expedite Notification Manager step that receives the notification and then passed to a second step for processing by a user-written application.

ATTENTION: Expedite Notification Manager appends the data to this file. You should remove processed notifications prior to a subsequent Expedite Notification Manager run.

ENMNTFY attributes

Create ENMNTFY with the following attributes:

This attribute:	Has these Characteristics:
Space	ENMNTFY has no minimum space requirement. It must be large enough to hold unprocessed notifications.
Record format	ENMNTFY can have fixed, fixed blocked, variable, or variable blocked records.
Record length	ENMNTFY has a record length of 80 bytes.

ENMNTFY example

A ENMNTFY example log is shown in Chapter 4, "Receiving notification data."

Optional files

You can use some of the optional Expedite Notification Manager files. These files are described here.

Profile command file (ENMIPRO)

The ddname of the profile command file is ENMIPRO. Expedite Notification Manager uses the information contained in ENMIPRO to determine whether or not to start a user-supplied application or produce trace, log, or link output.

Defaults exist for all command parameters contained in the profile command file. However, if you want to override the default parameter values, the overrides must be provided in this file.

ENMIPRO attributes

Create ENMIPRO with the following attributes:

This attribute:	Has these characteristics:
Space	ENMIPRO has no minimum space requirement. It must be large enough to hold all the profile commands.
Record format	ENMIPRO can have fixed, fixed blocked, variable, or variable blocked records.
Record length	ENMIPRO has a record length of 80 bytes.

ENMIPRO example

The following is an example of an ENMIPRO file:

```
execute userappl(1);
trace log(y);
```

Profile response file (ENMOPRO)

The ddname of the profile response file is ENMOPRO. When Expedite Notification Manager reads the profile command file, it echoes the profile commands to ENMOPRO, along with the associated return codes. The return record in ENMOPRO contains the return codes for each command. The PROFILERC record contains the return code for the processing of the entire ENMIPRO file.

NOTE: When the information is to be sent to the job output, the ddname of the profile response file should point to SYSOUT=* rather than to a data file.

ENMOPRO attributes

Create ENMOPRO with the following attributes:

This attribut:	Has these characteristics:
Space	ENMOPRO must be large enough to hold the echoed ENMIPRO commands and their associated responses.
Record format	ENMOPRO can have fixed, fixed blocked, variable, or variable blocked records.
Record length	ENMOPRO has a record length of 80 bytes.

ENMOPRO example

The following is the ENMOPRO file produced by the profile command file:

```
execute userappl(1);
RETURN(00000);
trace log(y);
RETURN(00000);
PROFILERC(00000);
```

Error message file (ENMERROR)

Expedite Notification Manager can provide error descriptions in the RETURN and PROFILERC response records. These error descriptions are contained in the ENMERROR file which was provided on the installation tape for Expedite Notification Manager.

If you include this file with the ddname ENMERROR in Expedite Notification Manager, Expedite Notification Manager writes error descriptions associated with the error message codes when errors occur. If you do not include this file when you run Expedite Notification Manager, error message codes display without error message text. It is recommended that you use the ENMERROR file. Your system programmer can provide the file name to use.

Extended error text file (ENMTEXT)

Expedite Notification Manager can provide extended error descriptions in the RETURN and PROFILE response records. These extended error descriptions are contained in the ENMTEXT file which was provided on the installation tape for Expedite Notification Manager.

If you include this file with the ddname ENMTEXT in Expedite Notification Manager, Expedite Notification Manager includes extended error descriptions associated with error message codes when errors occur. If you do not include this file when you run Expedite Notification Manager, error messages display without extended error text. It is recommended that you use the ENMTEXT file. Your system programmer can provide you with the file name to use.

Logic trace file (ENMTRACE)

The ddname for the logic trace file is ENMTRACE. It contains Expedite Notification Manager module execution information. If you requested a trace using the TRACE profile command, pre-allocate the file and include the ddname ENMTRACE in your Expedite Notification Manager execution JCL. You need to request a trace only when you are working with help desk on a problem.

ENMTRACE attributes

Create ENMTRACE with the following attributes:

This attribute:	Has these characteristics:
Space	ENMTRACE produces approximately 100 80-byte records
Record format	ENMTRACE can have fixed, fixed blocked, variable, or variable b locked records.
Record length	ENMTRACE has a record length of 80 bytes.
Data set organization	ENMTRACE must be a sequential data set. It cannot be a member of a PDS.

Link trace file (ENMLINK)

The ddname for the link trace file is ENMLINK. It contains communication information produced by the link trace. If you request a link trace using the LINK parameter of the TRACE profile command, pre-allocate the file and include the ddname ENMLINK. You need to request a link trace only if you are working with help desk on a problem.

ENMLINK attributes

Create ENMLINK with the following attributes:

This attribute:	Has these characteristics:
Space	ENMLINK produces 100 121-byte records.
Record format	ENMLINK has a record format that can have fixed, fixed blocked, variable, and variable blocked records.
Record length	ENMLINK has a record length of 121 bytes.
Data set organization	ENMLINK must be a sequential data set. It cannot be a member of a PDS.

Run-time information (ENMRUN)

The ddname for the run-time information file is ENMRUN. If you request run-time information using the trace profile command, pre-allocate the file and include the ddname ENMRUN in your Expedite Notification Manager execution JCL.

Several methods can be used in association with this file; only some possibilities are listed below.

- The file associated with ddname ENMNRUN could be allocated as a generation data group (GDG). Each time the Expedite Notification Manager job runs, a new generation is created. The user must make sure run-time information is processed, if required, before the GDG reaches its limit and begins to discard older generations. For more information on how to define a GDG, refer to *DFSMS/MVS Access Method Services for the Integrated Catalog Facility.*
- The file associated with ddname ENMNRUN could be allocated as a new data set in the Expedite Notification Manager step that receives the notification and then passed to a second step for processing by a user-written application.

ATTENTION: Expedite Notification Manager appends the data to this file. Users should remove processed run-time information prior to subsequent Expedite Notification Manager runs.

ENMRUN attributes

Create ENMRUN with the following attributes:

This attribute:	Has these characteristics:
Space	ENMRUN produces 15 80-byte records per execution of Expedite Notification Manager.
Record format	ENMRUN can have fixed, fixed blocked, variable, or variable blocked records.
Record length	ENMRUN has a record length of 80 bytes.
Data set organization	ENMRUN must be a sequential data set. It cannot be a member of a PDS.

ENMRUN example

An example of an ENMRUN file is shown in Appendix A.

Internal reader (ENMIRDR)

This is the ddname for the internal reader. If you have Expedite Notification Manager submit the JCL for a user-written application as specified in the **User Application** field of the Information Exchange Administration Services event profile, this ddname is required. A value can be specified in the **User Application** field when an event is defined using Information Exchange Administration Services. This value represents a previously defined file that contains the JCL necessary to run a user-written application. Expedite Notification Manager submits this JCL via the internal reader ddname (ENMIRDR). Example JCL is provided in the section titled "Internal reader method" found in this book.

User-Application file

To start a user-written application using the internal reader, follow these steps:

1. Install Expedite Notification Manager.
2. Allocate a file with the characteristics listed below.
3. Put all necessary JCL records in the file.
4. Set the USERAPPL parameter on the EXECUTE profile command to a value of 1.
5. Assign the file name in the **User Application** field of the event, under event profile, in Information Exchange Administration Services.
6. Include the ENMIRDR ddname in the Expedite Notification Manager JCL stored in the APPC/MVS transaction program (TP) profile.

User-application file attributes

Create your user-application file with the following attributes:

This attribute:	Has these characteristics
Space	There is no minimum space requirement. It must be large enough to hold all your application JCL statement
Record format	You can have fixed, fixed blocked, variable, or variable blocked records.
Record length	The record length must be 80 bytes.
Data set organization	This file may be a sequential data set or a PDS member.

JCL for running Expedite Notification Manager examples

The following examples show different ways you can implement Expedite Notification Manager on your system. This JCL must be included in the APPC/MVS transaction program profile.

Internal reader method

The following JCL example shows how to specify the DD for the internal reader. This JCL must be added to the APPC/MVS transaction program profile.

```
//jobname JOB (insert your installation specific job statement)
//enmstep  EXEC PGM=ESMGR
//STEPLIB  DD  DSN=expedite.nm.esmload,DISP=SHR
//ENMNTFY  DD DSN=user1.notification.log,DISP=SHR
//ENMIPRO  DD DSN=user1.profile.input,DISP=SHR
//ENMOPRO  DD DSN=user1.profile.output,DISP=SHR
//ENMLINK  DD DSN=user1.link.trace,DISP=SHR
//ENMIRDR DD SYSOUT=(A,INTRDR)
//ENMTRACE DD DSN=user1.logic.trace,DISP=SHR
//ENMRUN   DD DSN=user1.run.log,DISP=SHR
//ENMERROR DD DSN=expedite.notification.error.messages,DISP=SHR
//ENMTEXT  DD DSN=expedite.notification.error.text,DISP=SHR
```

NOTE: You cannot use SYSIN or SYSOUT datasets in your user application JCL

Required setup:

- User application specified in Information Exchange Administration Services event profile. For this example, assume that the profile specifies: user1.application.jcl.
- ENMIPRO file contains an EXECUTE command with parameter userappl(1).
- user1.application.jcl is a file on your system and contains appropriate JCL statements.
- User has defined a message-arrival event in Information Exchange Administration Services.

When a message arrives in your mailbox that matches the event criteria and Expedite Notification Manager is executed on your MVS system:

1. The notification is logged in ENMNTFY.

2. The LU 6.2 conversation completes between Information Exchange and Expedite Notification Manager.
3. An open for output is performed on ddname ENMIRDR.
4. An open for input is performed on user1.application.jcl.
5. A record is read from user1.application.jcl.
6. The read record is written to ddname ENMIRDR.
7. The previous 2 steps are repeated until there are no more records in user1.application.jcl.
8. Expedite Notification Manager ends.

NOTE: JES3 users need to specify the internal reader as follows:

//ENMIRDR DD SYSOUT=(A,INTRDR),RECFM=FB,LRECL=80,BLKSIZE=0

User application as a second step

The following JCL sample shows how a user application (in this case Expedite Base/MVS) can be run as a second step provided that the Expedite Notification Manager step completed with a return code of 4 or less. This JCL must be added to your APPC/MVS transaction program profile.

NOTE: It is your responsibility to determine if Expedite Base/MVS completed successfully.

```
//jobname JOB (insert your installation specific job statement)
//enmstep  EXEC PGM=ESMGR
//STEPLIB  DD  DSN=expedite.nm.esmload,DISP=SHR
//ENMNTFY  DD DSN=user1.notification.log,DISP=SHR
//ENMIPRO  DD DSN=user1.profile.input,DISP=SHR
//ENMOPRO  DD DSN=user1.profile.output,DISP=SHR
//ENMLINK  DD DSN=user1.link.trace,DISP=SHR
//ENMTRACE DD DSN=user1.logic.trace,DISP=SHR
//ENMRUN   DD DSN=user1.run.log,DISP=SHR
//ENMERROR DD DSN=expedite.notification.error.messages,DISP=SHR
//ENMTEXT  DD DSN=expedite.notification.error.text,DISP=SHR
//expstep EXEC PGM=IEBASE,REGION=0K,COND=(4,LT)
//STEPLIB   DD DISP=SHR,DSN=expedite.msg.expload
//SYSPRINT DD DSN=user1.sysprint.data,DISP=SHR
//INMSG    DD DSN=user1.message.command,DISP=SHR
//INPRO    DD DSN=user1.profile.command,DISP=SHR
//OUTMSG   DD DSN=user1.message.response,DISP=OLD
//OUTPRO   DD DSN=user1.profile.response,DISP=SHR
//ERRORMSG DD DISP=SHR,DSN=expedite.error.messages
//ERRORTXT DD DISP=SHR,DSN=expedite.error.text
```

Required setup:

- ENMIPRO file contains an EXECUTE command with parameter userappl(0) or no ENMIPRO file specified (userappl defaults to 0).

- You defined a message-arrival event in Information Exchange Administration Services.

When a message arrives in your mailbox that matches the event criteria and Expedite Notification Manager is executed on your MVS system:

1. The notification is logged in ENMNTFY.
2. The LU 6.2 conversation completes between Information Exchange and Expedite Notification Manager.
3. The enmstep ends.
4. The expstep executes.

Installing Expedite Notification Manager in APPC/MVS

Transaction programs are maintained in the APPC/MVS transaction program profile using the APPC/MVS administration utility (ATBSDFMU), which is a batch job, or by using the APPC/MVS Administration Dialog. For more information about either of these facilities, refer to *z/OS MVS Planning: APPC Management*. Recommended values for the Expedite Notification Manager transaction program are provided in the program directory for Expedite Notification Manager.

Chapter 3

Expedite Notification Manager profile commands

You can create an Expedite Notification Manager profile to allow various Expedite Notification Manager options such as tracing, or to provide information to Expedite Notification Manager such as execution options. This chapter discusses the Expedite Notification Manager profile and describes the profile commands and response records. The Expedite Notification Manager profile is optional; if you do not create this profile, the default values will be used.

Expedite Notification Manager profile (ENMIPRO)

To create an Expedite Notification Manager profile, using a text editor, enter the profile commands in the profile command file and reference this file using the ddname ENMIPRO. Expedite Notification Manager echoes these commands along with the response records and their associated return codes to the profile response file (ENMOPRO ddname). You can review ENMOPRO to verify successful completion of the profile commands.

Expedite Notification Manager profile commands

The following commands can be used in your Expedite Notification Manager profile.

EXECUTE

Use this command to specify execution options for Expedite Notification Manager.

TRACE

Use this command to specify the information that Expedite Notification Manager records into the trace files.

EXECUTE command

The EXECUTE command specifies how to process the notification.

The syntax of the EXECUTE command is:

execute

userappl(0|1);

userappl

Indicates whether Expedite Notification Manager should try to start the program specified in the **User Application** field passed from the Information Exchange Administration Services profile.

0	Do not start the user application. This is the default.
1	Try to start the application specified in the **User Application** field.

EXECUTE command example

```
execute userappl(0);
```

TRACE command

The TRACE command specifies what information is recorded in the trace files during a session. Possible traces are LINK, LOGIC, and LOG. All traces are used for problem determination. Trace information is written to the appropriate file.

The syntax of the TRACE command is:

TRACE

link(n|y)
logic(n|y)
log(n|y);

link

Indicates whether or not Expedite Notification Manager should trace communication link protocol. Expedite Notification Manager writes the communication protocol data to the file referenced by the ENMLINK ddname.

n	Do not write the communication link information to the file referenced by the ENMLINK ddname. This is the default.
y	Write the communication link information to the file referenced by the ENMLINK ddname.

logic

Indicates whether the file referenced by the ENMTRACE ddname should contain the Expedite Notification Manager module execution information.

n	Do not write the Expedite Notification Manager module execution information to the file referenced by the ENMTRACE ddname. This is the default.
y	Write the Expedite Notification Manager module execution information to the file referenced by the ENMTRACE ddname.

log

Indicates if Expedite Notification Manager run-time information should be recorded.

n	Do not write the Expedite Notification Manager run-time information to the file referenced by the ENMRUN ddname. This is the default.
y	Write the Expedite Notification Manager run-time information to the file referenced by the ENMRUN ddname.

TRACE command example

```
trace link(y);
```

Working with profile response records

The profile response file ENMOPRO contains an echo of the profile commands and their response records.

The profile response records are:

- PROFILERC

  This record indicates the completion of ENMIPRO.

- RETURN

  This record indicates the completion of a command in ENMIPRO. The following sections provide detailed information on each of these records.

PROFILERC record

The PROFILERC record is the last record in ENMOPRO. The PROFILERC record indicates the processing of the profile commands is complete. A zero value indicates that all the profile commands completed successfully.

The following example shows the format of the PROFILERC record:

PROFILERC(*return code*) ERRDESC(*error description*)

ERRTEXT(*error text*);

NOTE: There is only one PROFILERC record in each ENMOPRO.

PROFILERC
Indicates whether Expedite Notification Manager processed the profile commands successfully. If the return code is zero, the commands completed successfully. If the return code is not zero, the program displays an error number along with ERRDESC and ERRTEXT records. This parameter contains **5** numeric characters.

ERRDESC
Provides a short description of an error. If the return code is zero, this parameter is not in ENMOPRO. This parameter contains **1** to **76** alphanumeric characters.

ERRTEXT
Provides a detailed description of an error and may suggest steps to correct the problem. There may be multiple error text records in the file. If the return code is zero, this parameter is not in ENMOPRO. This parameter contains **1** to **76** alphanumeric characters.

RETURN record

The RETURN record indicates the completion of a command in ENMIPRO. A zero value indicates that the command completed successfully.

The following example shows the format of the RETURN record:

RETURN(*return*) ERRDESC(*error description*)

ERRTEXT(*error text*);

RETURN
Indicates completion of an Expedite Notification Manager profile command. If the return code is zero, the command completed successfully. If the return code is not zero, the program displays an error number along with ERRDESC and ERRTEXT records. This parameter contains **5** numeric characters.

ERRDESC
Provides a short description of an error. If the return code is zero, this parameter is not in ENMOPRO. This parameter contains **1** to **76** alphanumeric characters.

ERRTEXT
Provides a detailed description of an error and may suggest steps to correct the problem. There may be multiple error text records in the file. If the return code is zero, this parameter is not in ENMOPRO. This parameter contains **1** to **76** alphanumeric characters.

Chapter 4

Receiving notification data

Expedite Notification Manager places all notification data in the notification log. The Expedite Notification Manager notification log contains the data from notifications, such as event date, event time, event type, final return code data, and possible error text. Expedite Notification Manager only logs notifications received in Expedite Notification Manager format (as specified in the Information Exchange event profile). New notifications are appended to the end of this file; therefore, it can become very large and you need to provide maintenance at regular intervals.

All the parameters shown here may not be included in the notification log, because Expedite Notification Manager does not write parameters with blank values. The parameters received by Expedite Notification Manager depend on the notification type and the information completed in the Information Exchange event profile.

Notification log information

The following is the format of information written to the Expedite Notification Manager notification log. Only parameters with nonblank values are written.

EVENT record

The syntax of the EVENT record is:

EVENT
EVENTDATE(*date*) EVENTTIME(*time*)
EVENTTYPE(*S|M|A*)
EVENTID(*eventid*)
ACCOUNT(*account*) USERID(*userid*)
SNDSYSID(*sender's system ID*) SNDACCOUNT(s*ender's account*)
SNDUSERID(*sender's userid*)
ALIAS(*alias type*) ALIASNAME*(alias name*)
SENDERREF(*reference ID*) CLASS(*class*)
PRIORITY(*A|H*) MSGNAME(*message name*)
MSGLENGTH(*message length*) MODE(*T/S/R/blank*)
DATATYPE(*A|E|N*) MSGKEY(*message key*)

CYCLE(*D*|*E*|*M*|*N*|*W*|*X*|*Y*|*Z*) DAY(*day*)
TIME(*time*) TIMEZONE(*offset*)
TPNAME(*transaction*)
XTPNAME(*hex transaction*)
PARTNERLU(*partner LU*)
USERAPPL(*user application*)
APPLDATA(*application data*);

EVENTDATE
Date the event was received by Expedite Notification Manager. The format of this parameter is MM/DD/YYYY. This parameter contains 10 alphanumeric characters.

EVENTTIME
Time the event was received by Expedite Notification Manager. The time is displayed in local-system time and military time. The format of this parameter is HH:MM:SS. This parameter contains 8 alphanumeric characters.

EVENTTYPE
Type of notification received.

S	Scheduled event
M	Message-arrival event
A	Administrator/user-initiated event

EVENTID
Event ID specified by the user for the event. This parameter contains 1 to 8 alphanumeric characters.

ACCOUNT
Account for whom the event occurred. This parameter contains 1 to 8 alphanumeric characters.

USERID
User ID for whom the event occurred. This parameter contains 1 to 8 alphanumeric characters.

SNDSYSID
System ID of the message sender. This parameter contains 1 to 3 alphanumeric characters.

SNDACCOUNT
Account ID of the message sender. This parameter contains 1 to 8 alphanumeric characters.

SNDUSERID
User ID of the message sender. This parameter contains 1 to 8 alphanumeric characters.

ALIAS
Table type and table name of an alias table used by the message sender to refer to the receiver.

G*XXX*	Global alias table, where *XXX* identifies a 1- to 3-character table name.
O*XXX*	Organizational alias table, where *XXX* identifies a 1- to 3-character table name.
P*XXX*	Private alias table, where *XXX* identifies a 1- to 3-character table name.

This parameter contains 1 to 4 alphanumeric characters.

ALIASNAME
Alias name defined in the alias table used by the message sender to refer to the receiver. This parameter contains 1 to 16 alphanumeric characters.

SENDERREF
Sender ID specified in the EDI data, if applicable. This parameter contains 1 to 35 characters.

CLASS
User class of the data specified by the message sender to identify the data. This parameter contains 1 to 8 alphanumeric characters.

PRIORITY
Message priority specified in the event definition.

A	All messages
H	High-priority

MSGNAME
Name for the data specified by the message sender. This parameter contains 1 to 8 alphanumeric characters.

MSGLENGTH
Message size in characters, without headers and CDH. This parameter contains 1 to 10 numeric characters.

MODE
Sender-specified network data class field for this data.

blank	A normal message with no CDH
T	A test-mode message with no CDH
S	A normal message with CDH available
R	A test-mode message with CDH available

DATATYPE
Data type selected.

A	Any data
E	EDI data
N	Non-EDI data

MSGKEY
Message key for specific receive. This parameter contains 1 to 20 alphanumeric characters.

CYCLE
Scheduled-event cycle definition.

D	Daily
E	Last day of the month

M	Monthly
N	Every day except the day specified in the **day of the month** field
W	Weekly
X	Every day except Saturday
Y	Every day except Sunday
Z	Every day except Saturday and Sunday

DAY

Days of the week or month in the definition that triggered the event. This field defaults to blanks for message-arrival events. For M and N schedules, this field contains the exact date (1-31). For schedules of D, X, Y, Z, E, this field contains 000. For W schedules, this field contains a cumulative value for all days.

Day of the week:	Assigned value:
Sunday	128
Monday	64
Tuesday	32
Wednesday	16
Thursday	8
Friday	4
Saturday	2
reserved	1

This is a cumulative number representing the sum of all scheduled days. For example, d=%DAY% substitution value displays as d=112. This represents the total of 64 + 32 + 16, or Monday, Tuesday, and Wednesday. This parameter contains 1 to 3 numeric characters.

TIME

Scheduled time in the definition that triggered the event. The time is based on a 24-hour clock. For example, 09:00=9 a.m.; 16:00=4 p.m.; 21:00=9 p.m. The scheduled time displays in the time zone of the scheduled-event definition owner. This parameter is in the format HH:MM and contains 5 numeric characters.

TIMEZONE

Event offset from Greenwich mean time in the format EHHMM or WHHMM. This parameter contains 5 alphanumeric characters.

TPNAME

TPNAME as defined in the Information Exchange Administration Services event profile. The same name must be defined in the APPC/MVS TP profile. This parameter contains 1 to 64 alphanumeric characters.

XTPNAME

Hexadecimal representation of the TPNAME, if it was specified as a hex value in Information Exchange Administration Services. This parameter contains 1 to 128 alphanumeric characters.

PARTNERLU
Partner's LU name defined in the Information Exchange Administration Services event profile. This parameter contains 1 to 8 alphanumeric characters.

USERAPPL
User application field as defined in the Information Exchange Administration Services event profile. This represents a file containing JCL that is to be submitted on the user's Expedite Notification Manager system. This parameter contains 1 to 75 alphanumeric characters; however, only the first 54 characters are used because of file naming limitations.

APPLDATA
Application data field specified in the Information Exchange Administration Services event profile. This parameter contains from 1 to 300 alphanumeric characters.

RETURN record

The syntax of the RETURN record is:

RETURN(*return*) REASON(*reason*)
ERRTEXT(*error text*) ...;

RETURN
Completion code for the Expedite Notification Manager command. If the return code is zero, the command completed normally. If the return code is not zero, Expedite Notification Manager displays an error number and possibly a REASON and ERRTEXT value. This parameter contains 1 to 5 numeric characters.

REASON
More information about what caused the error. Expedite Notification Manager usually includes REASON when an error occurs opening a file. This parameter contains 1 to 76 alphanumeric characters.

ERRTEXT
Detailed description of an error and may suggest steps to correct the problem. Expedite Notification Manager may include this parameter more than once in a record if the error text consists of multiple lines. This parameter contains 1 to 76 alphanumeric characters.

Notification log example

The following are examples of Expedite Notification Manager notifications:

- User-initiated

```
EVENT
    EVENTDATE(06/28/1996) EVENTTIME(14:07:42)
    EVENTTYPE(A) ACCOUNT(ACCT) USERID(USER1) TPNAME(ENMGR)
    PARTNERLU(ACCTN03)
    USERAPPL(USER1.APPL.JCL(MBR1)) APPLDATA(This is a test);
    RETURN(0);
```

- Message-arrival

```
EVENT
    EVENTDATE(06/28/1996) EVENTTIME(14:08:00)
    EVENTTYPE(M) ACCOUNT(ACCT) USERID(USER1) SNDACCOUNT(ACCT)
    SNDUSERID(USR2)
    CLASS(FFMSG001) MSGLENGTH(0000000079) MODE(S) DATATYPE(N)
    MSGKEY(FFAD163464279E96F4F4) TPNAME(ENMGR) PARTNERLU(ACCTNM03)
    USERAPPL(USER1.APPL.JCL(MBR1))
    APPLDATA(msgkey=FFAD163464279E96F4F4);
    RETURN(0);
```

- Scheduled

```
EVENT
    EVENTDATE(06/28/1996) EVENTTIME(14:14:43)
    EVENTTYPE(S) EVENTID(SCHED) ACCOUNT(ACCT) USERID(USER1) CYCLE(D)
    DAY(000) TIME(14:14) TIMEZONE(W0400) TPNAME(ENMGR)
    PARTNERLU(ACCTM03)
    USERAPPL(USER1.APPL.JCL(MBR1)) APPLDATA(cycle=D);
    RETURN(0);
```

Chapter 5

Application Example

Using the information supplied in the event notification, you can write applications to determine which Expedite Base/MVS job is executed. The example below illustrates an application that processes the notification, uses the user ID for whom the event occurred as the member name, and submits the JCL contained in that member. The following assumptions are made:

- A partitioned data set (PDS) containing the JCL necessary to run Expedite Base/MVS has been created. (The REXX source code shown below has the name coded as user1.expedite.jcl.)
  - The member name for the PDS corresponds to the user ID for whom the job will be run. The PDS member points to a file containing the user's Information Exchange user ID and password.
  - The Expedite Base/MVS job issues a receive command to receive all items from the user's mailbox.
- The notification log is created each time the Expedite Notification Manager is initiated by an Information Exchange event and deleted when the application completes normally.

The JCL to run this application example is provided with the Expedite Notification Manager product in the SMPJCL file, and is shown below.

```
//jobname JOB (insert your installation specific job statement)
//STEP1   EXEC PGM=IEBGENER
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD DUMMY
//SYSUT1 DD DSN=user1.rexx.src(esmsmp),DISP=SHR
//SYSUT2    DD DISP=(,PASS),DSN=&&CLIST(expsub)
//           UNIT=SYSDA,SPACE=(CYL,(1,1,1))
//STEP2   EXEC PGM=IKJEFT01,DYNAMNBR=50,REGION=6M
//SYSPROC   DD DISP=(OLD,DELETE),DSN=&&CLIST
//NOTELOG   DD DSN=user1.notification.log,DISP=(OLD,DELETE)
//SYSTSPRT  DD SYSOUT=*
//SYSTSIN   DD DSN=user1.command.file(esmsmp),DISP=SHR
```

This JCL has 2 steps:

- STEP1 puts the REXX source into member EXPSUB of a temporary data set called &&CLIST (SYSUT2 ddname).
- STEP2 executes EXPSUB.

The file user1.rexx.src(expsmp) is provided with Expedite Notification Manager in the SMPJCL file. The contents of this file is shown below.

```
/* REXX Clist                                                    */
/* EXPSMPL: Expedite Notification Manager application           */
/*          example                                             */
/* FUNCTION: reads the notification log(ddname notelog)         */
/*         finds the notification userid to use as a member     */
/*         name in a PDS containing Expedite Base/MVS jobs.     */
/* INPUT                                                        */
/*       notelog - DDname pointing to the notification log      */
/*             created when Expedite Notification Manager       */
/*             ran                                              */
/*       Expedite Base/MVS file name - the name of the PDS      */
/*             containing Expedite Base/MVS JCL broken up       */
/*             by userid.                                       */
/* OUTPUT                                                       */
/*       Expedite Base/MVS job is submitted for the userid      */
/*             for whom the event occurred in Information       */
/*             Exchange.                                        */
/* WARNING:  This example does not determine the status of      */
/*       your last Expedite Base/MVS job.  You must make        */
/*       sure that the last execution of Expedite Base/MVS      */
/*       for the userid completed successfully or data could    */
/*       be lost.  Recovery is discussed in the Expedite        */
/*       Base/MVS manual GC34-2204.                             */
done = ‘no’        /* initialize loop control variables        */
eof  = 'no'
/* loop until end of file or " USERID(" found                   */
do while done = 'no'
  "EXECIO 1 DISKR notelog"  /* Process notelog file             */
  if RC = 0 then            /* a record was read                */
  do
    PULL record             /* place data in variable record    */
    if POS(' USERID(',record) ¬= 0 Then
    done = 'yes'            /* "USERID(" found on this record*/
    else NOP
  end
  else                      /* no record read                   */
  do
    eof = 'yes'             /* must mean end of file found      */
    leave                   /* exit loop                        */
  end
end                         /* end of loop                      */
/* If no userid found then no further processing                */
if eof = 'yes' then
EXIT 0
/*                                                              */
/* Otherwise parse out the actual userid from the log entry.    */
/* Determine where the userid field begins (just after the      */
/* “ USERID(“                                                   */
startpos = POS(' USERID(',record) + length(' USERID(')
/*  Find the first ")" following the string "USERID("           */
/*  to determine the member name length.                        */
/*  Then calculate the length of the userid                     */
memlen = POS(')',record,startpos) - startpos
/*                                                              */
/*Assign the out_ds name to submit concatenating member name    */
in_ds = 'user1.expedite.jcl'
out_ds = in_ds"("SUBSTR(record,startpos,memlen)")"
"submit '"out_ds"'"
```

The file user1.command.file(esmcmd) used to execute the REXX source is provided with Expedite Notification Manager in the SMPJCL file. The contents of this file is shown below.

expsub

This example can be executed by Expedite Notification Manager by doing the following:

- Coding the ENMNTFY ddname in the APPC/MVS TP profile as follows:

```
//ENMNTFY  DD DSN=user1.notification.log,DISP=(NEW,CATLG),UNIT=SYSDA,
//         SPACE=(TRK,(1,1)),DCB=(RECFM=F,LRECL=80,DSORG=PS)
```

- Specifying the Information Exchange event profile USERAPPL parameter to be the file where the above JCL resides on your system.
- Creating an Expedite Notification Manager profile (ENMIPRO ddname) with an EXECUTE USERAPPL(1); command.

This example can also be executed as a second job step to the Expedite Notification Manager job. This is described in "User application as a second step" in Chapter 2. Substitute the REXX JCL for the Expedite Base/MVS JCL.

Appendix A

Expedite Notification Manager logs and traces

When you need detailed information about Expedite Notification Manager processing, you can examine the Expedite Notification Manager logs and traces. The notification log is always generated when notifications are received. To generate any run-time or trace information, you need to specify a trace command in your profile command file. This appendix describes the logs and traces, and provides examples.

Expedite Notification Manager logs

There are two types of Expedite Notification Manager logs: the notification (ENMNTFY ddname) and the run log (ENMRUN ddname). The notification log documents the received notification, and the run log documents Expedite Notification Manager processing.

Notification log

Expedite Notification Manager places all notification data in the notification log. The Expedite Notification Manager notification log contains the data from notifications, such as event date, event time, event type, final return-code data, and possible error text. For an example of the Expedite Notification Manager Notification log, see Chapter 4, "Receiving notifications."

Run log

During execution, Expedite Notification Manager creates various messages that display the status of Expedite Notification Manager. These messages can be written to the file referenced by the ENMRUN ddname by specifying LOG(Y) on the TRACE command, or they can be suppressed completely by specifying LOG(N) on the TRACE command. The following is an example of the run-time information written to the file.

```
Starting ENM on Mon Jun 17 10:53:36 1996
Connected
Receiving notification
Decoding GDS
GDS decoded
Sending acknowledgment
```

```
Disconnecting
Calling USER1.APPL.JCL(IRDR)
Ending ENM
```

Expedite Notification Manager traces

There are two types of Expedite Notification Manager trace files: the link trace (ENMLINK ddname) and the logic trace (ENMTRACE ddname). These trace files are used for problem determination. All trace file options default to N, so the appropriate trace parameter must be set to Y on the TRACE command, and the ddname included in the ENMJCL prior to Expedite Notification Manager being invoked by APPC/MVS, for the trace files to be created.

Link trace - TRACE LINK(Y)

The link trace lists the flow of data across the communication link. This trace is used primarily for problem determination by network support personnel.

Link trace example

```
Expedite Notification for MVS Version 1.1.1 - 05/01/05 started - Thu Jun  9 12:40:53 2005
12:40:53 | CMACCP(Accept Conversation) CONVID(73C8) Return(OK)
12:40:53 | CMECT(Extract Conversation Type) CONVID(73C8) Type(Mapped) turn(OK)
12:40:54 | CMEMN(Extract Mode Name) CONVID(73C8) Mode(LU62SYS1)Return(OK)
12:40:54 | CMEPLN(Extract Partner LU) LU(IBINET.IBM0DIM1 ) Return(OK)
12:40:54 | CMESL(Extract Sync Level) Level(None) Return(OK)
12:40:54 | CMECS(Extract Conversation State) State(Receive) Return(OK)
12:40:54 | CMSRT(Set Receive Type) RECEIVE(Wait) Return(OK)
12:40:54 | CMRCV(Receive) CONVID(73C8) Complete(Yes) Send(Yes) Return(OK)
12:40:55 | CMSEND(Send) CONVID(73C8) Sent(000000)  Return(OK)
12:40:55 | CMRCV(Disconnect) Return(Normal)
```

Logic trace - TRACE LOGIC(Y)

The logic trace lists the logic flow between Expedite Notification Manager components. This trace is primarily used for problem determination by network support personnel. The trace is stored in the file referenced by ENMTRACE ddname.

logic trace example

```
=Message=>Expedite Notification for MVS started - Version 1.1.1 - 05/0<-
=>1/05<-
=Module=>ProTrace rc(0)<-
=Module=>outretn rc(0)<-
=Module=>checkrc rc(0)<-
=Module=>getrc rc(0)<-
=Module=>outprorc rc(0)<-
=Module=>ProProfile rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
```

=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>ENMinit rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>ENMrcvn rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>valgds rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMNTFY<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>DecodeNLog rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>ENMsndr rc(0)<-
=Message=>Opening file as >text stream<-

```
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
=Module=>Disconnect rc(0)<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMERROR<-
=Message=>open mode->r<-
=Message=>Open successful<-
=Message=>Opening file as >text stream<-
=Message=>DDN:ENMRUN<-
=Message=>open mode->a<-
=Message=>Open successful<-
```

Appendix A

Messages and codes

This appendix lists and describes messages that Expedite Notification Manager generates; it also includes console messages. Informational messages are preceded by the letter I, error messages are preceded by the letter E, and warnings are preceded by the letter W. Console messages are written to the MVS operator's console.

19022 I Receiving notification

Explanation: The program is attempting to receive a notification.

User Response: Informational only. No response is needed.

19023 I Decoding GDS

Explanation: A notification has been received and is being decoded.

User Response: Informational only. No response is needed.

19024 I GDS decoded

Explanation: The notification has been successfully decoded.

User Response: Informational only. No response is needed.

19025 I Sending acknowledgment

Explanation: Information Exchange is being given notification status.

User Response: Informational only. No response is needed.

19026 I Calling %S

Explanation: The program is attempting to run the application passed in the received notification.

User Response: Informational only. No response is needed.

19027 I Disconnecting

Explanation: The program is being disconnected from Information Exchange.

User Response: Informational only. No response is needed.

19029 I Connected

Explanation: The application is establishing a communication session with Information Exchange.

User Response: Informational only. No response is needed.

19051 I Starting ENM on %S

Explanation: The Expedite Notification Manager program is starting.

User Response: Informational only. No response is needed.

19052 E Error allocating memory

Explanation: The program could not allocate enough memory to perform a necessary function.

User Response: End all unnecessary programs in the system.

19101 E Invalid profile command

Explanation: The profile command file contained an invalid profile command.

User Response: Check your profile command file ENMIPRO for possible causes.

19102 E Invalid profile parameter

Explanation: A profile command contained an invalid parameter.

User Response: Check your profile command file ENMIPRO for possible causes.

19103 E Missing semicolon or expected parameter not found

Explanation: Expedite Notification Manager did not get the expected parameter or semicolon.

User Response: Check your profile command file ENMIPRO for possible causes.

19104 E Expected profile command not found

Explanation: Expedite Notification Manager did not find the expected profile command.

User Response: Check your profile command file ENMIPRO for possible causes.

19105 E Duplicate profile parameter found

Explanation: A duplicate parameter was encountered.

User Response: Check your profile command file ENMIPRO for possible causes.

19106 E Null character found in profile file

Explanation: A NULL character was found in your command profile file.

User Response: Check your profile command file ENMIPRO for the NULL character and remove any occurrences.

19107 E Profile command or parameter too long

Explanation: A command or parameter was encountered that is too long.

User Response: Check your profile command file ENMIPRO for possible causes.

19131 E Invalid USERAPPL on EXECUTE command

Explanation: You specified an invalid USERAPPL parameter in the EXECUTE command. USERAPPL must be 0 or 1.

User Response: Correct the USERAPPL parameter on the EXECUTE command in the profile command file ENMIPRO. Retry the program.

19141 E Invalid LINK specified on TRACE command

Explanation: You specified an invalid LINK parameter on the TRACE command. LINK must be Y or N.

User Response: Correct the LINK parameter on the TRACE command in the profile command file ENMIPRO. Retry the program.

19142 E invalid LOGIC specified on TRACE command

Explanation: You specified an invalid LOGIC parameter in the TRACE command. LOGIC must be Y or N.

User Response: Correct the LOGIC parameter on the TRACE command in the profile command file, ENMIPRO. Retry the program.

19143 E invalid LOG specified on TRACE command

Explanation: You specified an invalid LOG parameter in the TRACE command. LOG must be Y or N.

User Response: Correct the LOG parameter on the TRACE command in the profile command file, ENMIPRO. Retry the program.

19156 E Error calling user's application

Explanation: The user application could not be run.

User Response: Correct prior messages for possible causes.

19177 E invalid conversation type

Explanation: An incorrect conversation type was found between the Expedite Notification Manager and Information Exchange.

User Response: Contact the Information Exchange Help Desk. You will need to provide an ENMLINK trace file.

19178 W Unable to determine conversation type

Explanation: Expedite Notification Manager was unable to determine conversation type.

User Response: This information will not be included in the ENMLINK trace file.

19179 W Unable to determine mode name

Explanation: Expedite Notification Manager was unable to determine the mode name.

User Response: This information will not be included in the ENMLINK trace file.

19180 W Unable to determine partner LU

Explanation: Expedite Notification Manager was unable to determine the partner LU.

User Response: This information will not be included in the ENMLINK trace file.

19181 W Unable to determine sync level

Explanation: Expedite Notification Manager was unable to determine the sync level.

User Response: This information will not be included in the ENMLINK trace file.

19182 E Invalid sync level

Explanation: The sync level between Expedite Notification Manager and Information Exchange is incorrect.

User Response: Contact the Information Exchange Help Desk. You will need to provide an ENMLINK trace file.

19183 W Unable to determine conversation state

Explanation: Expedite Notification Manager was unable to determine the conversation state.

User Response: This information will not be included in the ENMLINK trace file.

19185 E Unable to set receive type

Explanation: Expedite Notification Manager was unable to set the receive type.

User Response: This information will not be included in the ENMLINK trace file.

19201 E Error receiving notification

Explanation: An error occurred while attempting to receive a notification.

User Response: Check prior messages for possible causes.

19251 E Error decoding GDS in notification

Explanation: An invalid notification GDS was received.

User Response: Check file ENMTRACE for possible causes.

19252 E Error opening notification log

Explanation: The notification log could not be opened.

User Response: The file may be in use by another program. End all other applications and try again.

19253 E Error in notification log data

Explanation: An error occurred while updating the notification log

User Response: The file may be in use by another program or the file may be out of space. End all other applications, process any unprocessed notifications, reallocate the file and try again.

19254 E Error writing notification log

Explanation: An error occurred while updating the notification log

User Response: The file may be in use by another program or the file may be out of space. End all other applications, process any unprocessed notifications, reallocate the file and try again.

19256 E Error in GDSID from IE

Explanation: The data received was not recognized as a valid notification.

User Response: No action may be necessary. This may happen if your Information Exchange event profile specified a value of N for Expedite Notification Manager format. Please change that value to a Y and try again.

19257 E ERROR IN GDS data from IE

Explanation: The notification received has an internal error.

User Response: No action may be necessary. If the problem continues, notify the Information Exchange Help Desk.

19258 E Error in GDS length from IE

Explanation: The notification received has an internal error.

User Response: No action may be necessary. If the problem continues, notify the Information Exchange Help Desk.

99999 I Ending ENM

Explanation: Expedite Notification Manager is ending.

User Response: Informational only. Check prior messages to determine if Expedite Notification Manager completed successfully.

Console messages

ESM19700 CMACCP failed. Reason %S

Explanation: The CMACCP command issued by Expedite Notification Manager failed with the reason listed.

User Response: This message is written to the MVS operators console. The notification is not received. Contact the Customer Assistance Department.

ESM19701 Incorrect conversation type

Explanation: An incorrect conversation type was found between Expedite Notification Manager and Information Exchange.

User Response: This message is written to the MVS operators console. The notification is not received. Contact the Customer Assistance Department

ESM19702 Incorrect sync level

Explanation: An incorrect sync level was found between Expedite Notification Manager and Information Exchange.

User Response: This message is written to the MVS operators console. The notification is not received. Contact the Customer Assistance Department.

ESM19703 CMRCV failed. Reason %S

Explanation: The CMRCV command issued by Expedite Notification Manager failed with the reason listed.

User Response: This message is written to the MVS operators console. The notification is not received. Contact the Customer Assistance Department.

ESM19704 Unable to open %S

Explanation: Expedite Notification Manager was unable to open the file name specified.

User Response: This message is written to the MVS operators console. The user application will not be started. Check that the file exists or is not in use by another application.

ESM19705 Unable to open internal reader

Explanation: Expedite Notification Manager was unable to open the file specified on the ENMIRDR ddname.

User Response: This message is written to the MVS operators console. The user application will not be started. Check that the ENMIRDR ddname has been correctly specified in the transaction program definition to APPC/MVS.

Glossary

This glossary defines words as they are used in this book. If you are looking for a term and cannot find it here, see the Dictionary of Computing for additional definitions.

This glossary includes terms and definitions from:

- The American National Dictionary for Information Systems, ANSI x3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American Standards Institute, 11 West 42 Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The Information Technology Vocabulary, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition.

A

account. A name that identifies an account to a program, device, or system.

alphanumeric. Generally, any keyboard character, but for practical purposes should be restricted to alphabetic, numeric, space, and common punctuation characters.

C

CDH. Common Data Header

character. Generally, any letter, number, punctuation mark, or other symbol that can be entered at a computer keyboard.

command. A request to run a particular program or function.

Common Data Header (CDH). A set of control information about a file, which is sent to Information Exchange by the sending interface. When the file is received by the trading partner, the receiving interface can use the information in the CDH.

Customer Assistance Department. The group or organization within the network that responds to customer questions and assists them in using the product or service. See also Help Desk Operator.

D

data definition (DD) name. The name of a data definition

(DD) statement that corresponds to a data control block that contains the same name.

data definition statement. A job control statement

describing a data set associated with a specific job step

default. An alternative value an application program uses automatically when none has been specified.

E

EBCDIC. Extended binary-coded decimal inter-change code.

error me ssage file (ENMERROR). The file that provides Expedite Notification Manager with error descriptions in the RETURN and PROFILERC output records.

ENMTEXT. Extended error text file.

event. Any action that generates a notification in Information Exchange.

ENM. Expedite Notification Manager.

Expedite Notification Manager. An Advantis developed application that resides on the user's system and either answers the phone for dial-out, or is in the application started for LU6.2 leased-line call-out.

G

GDS. Generalized Data Stream.

Generalized Data Stream (GDS). A format of data as it comes from Information Exchange.

H

Help Desk Operator. A person who receives questions or problem reports from network users. See also Customer Assistance.

I

Network. The worldwide communications network that provides network solutions and a global infor-mation infrastructure through Advantis and other companies.

Information Exchange. A communication service that permits users to send and receive information electronically.

Information Exchange Administration Services. An online, panel-driven product that the Information Exchange Service Administrator uses to perform administrative tasks for Information Exchange.

Information Exchange mailbox. See mailbox.

Information Exchange service administrator. The person who coordinates the use of Information Exchange within a company.

M

mailbox. A temporary storage area for electronic mail from which data is retrieved by the intended recipient.

message-arrival event. The arrival of a message that meets criteria defined in the message-arrival definition

P

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (2) An item in a menu for which you specify a value or for which the system provides a value when the menu is interpreted. (3) Data passed between programs or procedures.

profile command file (ENMIPRO). A file in which you place profile commands pertaining to profile commands.

profile response file (ENMOPRO). A file containing Expedite Notification Manager’s response to profile commands. The ddname for this file is ENMOPRO.

response file. An output file into which Expedite Notification Manager echoes commands from ENMIPRO along with its associated return codes.

S

scheduled event. An event that occurs on a specified schedule.

service administrator. A primary contact person in an organization who controls use of the service by the users within the organization and who assists the service support groups.

syntax. The rules for constructing a command.

T

trace file. A file that provides you with trace information of transactions. In Expedite Notification Manager there are two types of trace files: link trace (ENMLINK) and logic trace (ENMTRACE).

U

user-initiated event. An event initiated by the user to occur immediately or at a specified date and time. The event occurs only once.

Index