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

ISO 9001 Certification
This product was developed using an ISO 9001 certified quality system.
Certification has been appointed by CSQ Italy (the Italian quality system assessors body). CSQ adheres to the mutually recognized European assessors (ITQ), which assesses and certifies quality systems in the field of information technology.
ITQ includes AFAQ (France), AIB (Belgium), BSI (United Kingdom), KEMA (Denmark), TÜV (Germany), and others.

Fourth Edition (June 1998)
This is a major revision of, and obsoletes, SH19-6964-02.
This edition applies to Version 2 Release 4 Modification Level 0 of Tivoli NetView Performance Monitor for MVS (program number 5655-043) and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. See “What’s New in This Book” for the changes made to this book. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.
Make sure you are using the correct edition for the level of the product.
Order the publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.
IBM welcomes your comments. A form for readers’ comments is provided at the back of this publication, or you may address your comments to the following address:

NPM Information Development
Rome Tivoli Laboratory
IBM SEMEA S.p.A.
Via Sciangai, 53
00144 Rome, Italy

FAX: (39.6) 5966.2077

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1984, 1998. All rights reserved.
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.
Contents

Notices .............................................. xi
Programming Interface Information .................. xi
Trademarks and Service Marks ....................... xii
IBM Trademarks .................................... xii
Trademarks of Other Companies .................. xiii

About This Book .................................... xv
Who Should Use This Book ........................ xv
How This Book Is Organized ........................ xv
Conventions Used in This Book .................... xvi
New in NPM Version 2 Release 4 ................... xvii
  New TCP/IP Session Collection Capability .......... xvii
  Multinode Persistent Sessions (MNPS) Data Collection .... xvii
  Communications Storage Manager (CSM) Data Collection .... xvii
  Rapid Transport Protocol (RTP) Statistics Support ...... xviii
What's New in This Book .......................... xviii
Using Syntax Diagrams .............................. xviii
  Reading Syntax Diagrams ....................... xviii
  Abbreviating Keywords ........................ xviii
  Parameters ..................................... xix
  Repeating Parameters ........................... xx
  Reading Fragments .............................. xx
  Long Syntax Diagrams ........................... xx
Where to Find More Information .................. xxi

Part 1. Installing NPM ................................ 1

Chapter 1. Preparing for Installation ................. 3
  Working with NPM Samples ..................... 3
  Rules for Coding NPM Initialization Statements .... 4
  Special Characters on Non-US Keyboards .......... 5
  Managing NPM Security .......................... 6
    Logon Processing ............................ 6
    Functional Security ......................... 7
    Graphic Subsystem Security .................. 8
    Console Security ............................ 8
  Defining Security for NPM-to-NPM Communication .... 8
  Using RACF with NPM ........................... 9

Chapter 2. Installing NPM with the NPM Initialization Program .... 11
  General Information ............................ 11
  Using the NPM Initialization Program ............ 14
    Using PF Keys ................................ 14
    Using Commands .............................. 14
    Getting Help ................................ 14
  Getting Started ............................... 14
  Using the Minimum or Normal Initialization Options .... 17
    Minimum Initialization ...................... 17
    Normal Initialization ...................... 18
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing NPM EXEC Members (FNMSCMDS)</td>
</tr>
<tr>
<td>Running NPM EXECs</td>
</tr>
<tr>
<td>Examples of NPM EXECs</td>
</tr>
<tr>
<td>Using the System Console</td>
</tr>
<tr>
<td>WTOR Interface</td>
</tr>
<tr>
<td>MVS MODIFY Command</td>
</tr>
<tr>
<td>Disabling System Console Support in MVS</td>
</tr>
<tr>
<td>Running an NPM EXEC from the System Console</td>
</tr>
<tr>
<td>Working with Groups of Logical Units</td>
</tr>
<tr>
<td>Contents of LU Groups</td>
</tr>
<tr>
<td>Processing an LU Group</td>
</tr>
<tr>
<td>Example of an LU Group List</td>
</tr>
<tr>
<td>Enhanced LU Groups</td>
</tr>
<tr>
<td>Defining NPM Desk/2 Resource Groups</td>
</tr>
</tbody>
</table>

**Chapter 11. Starting and Stopping NPM and Verifying Installation**

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before Starting NPM</td>
</tr>
<tr>
<td>FNMILOG</td>
</tr>
<tr>
<td>SYSPRINT</td>
</tr>
<tr>
<td>Starting NPM</td>
</tr>
<tr>
<td>Verifying NPM</td>
</tr>
<tr>
<td>Reviewing Your NPM Configuration</td>
</tr>
<tr>
<td>Verifying NPM Functions</td>
</tr>
<tr>
<td>Stopping NPM</td>
</tr>
<tr>
<td>Stopping under MVS</td>
</tr>
<tr>
<td>Cleaning Up NPM's CSA Control Blocks</td>
</tr>
</tbody>
</table>

**Chapter 12. Tuning NPM**

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inactivating Session Collection</td>
</tr>
<tr>
<td>Cutting CSA Usage during Session Collection</td>
</tr>
<tr>
<td>Collecting Session Data by Application</td>
</tr>
<tr>
<td>Tuning VSAM File Processing</td>
</tr>
<tr>
<td>Optimizing Data Set Organization</td>
</tr>
<tr>
<td>Using the LSR Option</td>
</tr>
<tr>
<td>Using the Performance Measurement Table</td>
</tr>
<tr>
<td>Tuning Initialization Statements</td>
</tr>
<tr>
<td>BUFFERS Statement</td>
</tr>
<tr>
<td>CDRM Statement</td>
</tr>
<tr>
<td>CONTROL Statement</td>
</tr>
<tr>
<td>FILE Statement</td>
</tr>
<tr>
<td>HOST Statement</td>
</tr>
<tr>
<td>NPM Statement</td>
</tr>
<tr>
<td>SAMP Statement</td>
</tr>
<tr>
<td>DEFAULTS Statement</td>
</tr>
<tr>
<td>NGRAPHS Statement</td>
</tr>
<tr>
<td>SYS Statement</td>
</tr>
<tr>
<td>TASK Statement</td>
</tr>
<tr>
<td>Tuning Performance Data Collection Commands</td>
</tr>
<tr>
<td>Tuning Network Collection</td>
</tr>
<tr>
<td>Tuning Session Collection</td>
</tr>
<tr>
<td>Tuning VTAM Collection</td>
</tr>
<tr>
<td>Tuning LAN Collection</td>
</tr>
<tr>
<td>Tuning NetWare Resources Data Collection</td>
</tr>
<tr>
<td>Tuning Accounting Collection</td>
</tr>
</tbody>
</table>
Chapter 13. Customizing NPM Messages

- Tuning Installation-Wide Exits ........................................ 221
- Tuning NPM for Large Numbers of NCPs ............................. 221
- Updating the NPM Message Table ..................................... 223
- Defining an NPM Message (FNMMNG) ................................. 223
- Message Inserts ......................................................... 229
- Defining an NPM Message Replacement String (FNMMRS) ..... 231
- Installing the Modified Message Module ............................ 232
- Writing Messages to the System Console .......................... 232
- Writing Messages to the SYSPRINT Data Set .................... 233
- Sending Messages to an NPM Operator Logged on to NPM ..... 234
- Sending Messages to an NPM Operator with Global Authority 234
- Sending Alerts to the NetView Program .......................... 235
- Message Table Changes for NPM V2R4 ............................ 235

Chapter 14. Customizing NPM Processing with Installation-Wide Exits 237

- The NPM Installation-Wide Exit ...................................... 237
- Register Contents ..................................................... 237
- NPM Installation-Wide Exit Functions .............................. 238
- Parameter Descriptions ............................................... 249
- Installing the NPM Installation-Wide Exit ......................... 258
- Installing the NPM Installation-Wide Exit for MVS .......... 259
- The PIU Analysis Exit ................................................. 259
- Register Contents ..................................................... 259
- Parameter Descriptions ............................................... 260
- Installing a PIU Analysis Exit ....................................... 262
- Testing a PIU Analysis Exit .......................................... 262

Glossary ........................................................................ 265

Index ............................................................................. 297
<table>
<thead>
<tr>
<th>Figures</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NPM Initialization Program Opening Panel (FNMYPLG) ................. 15</td>
</tr>
<tr>
<td>2. Migration Table (FNMYPMG) ............................................ 16</td>
</tr>
<tr>
<td>3. Main Menu (FNMYPMN) ..................................................... 17</td>
</tr>
<tr>
<td>4. Sample FNMINIT Member .................................................. 25</td>
</tr>
<tr>
<td>5. Full Customization Window .............................................. 113</td>
</tr>
<tr>
<td>6. Example of an Expanded Section of Full Customization Window .......... 113</td>
</tr>
<tr>
<td>7. Selecting VTAM Session Manager Exit Routines .......................... 122</td>
</tr>
<tr>
<td>8. Selecting SMF Exit Routines .......................................... 123</td>
</tr>
<tr>
<td>9. Control Flow for SMF Exits IEFU83 and IEFU84 .......................... 124</td>
</tr>
<tr>
<td>10. FNMUMODS Job to Receive and Apply USERMODs .......................... 135</td>
</tr>
<tr>
<td>11. Example FNMSRT .......................................................... 149</td>
</tr>
<tr>
<td>12. Example of Defining Default Data Destinations for Collections ........ 160</td>
</tr>
<tr>
<td>13. Example of Defining Default Data Destination Definitions for Other Record Types .................................................. 160</td>
</tr>
<tr>
<td>14. JCL for Compressing a Data Set ....................................... 161</td>
</tr>
<tr>
<td>15. Example of an LU Group ............................................... 170</td>
</tr>
<tr>
<td>16. Example of an NPM Desk/2 Group ...................................... 173</td>
</tr>
<tr>
<td>17. Example of a Definition of an NPM Desk/2 Resource Group ............. 174</td>
</tr>
<tr>
<td>18. JCL for Starting NPM under MVS ...................................... 177</td>
</tr>
<tr>
<td>19. NPM Virtual Storage Allocation ...................................... 182</td>
</tr>
<tr>
<td>20. NPM Logon (FN00LLOG) ............................................... 185</td>
</tr>
<tr>
<td>21. Primary Options (FN00PRI) ............................................ 186</td>
</tr>
<tr>
<td>22. National Language Selection (FN00LNG) ................................ 186</td>
</tr>
<tr>
<td>23. Security Profile (FN00UPB) ............................................ 187</td>
</tr>
<tr>
<td>24. Data File Display and Management (FN02OFL) ........................... 187</td>
</tr>
<tr>
<td>25. Summary Display (FN02OPS) ............................................ 188</td>
</tr>
<tr>
<td>26. Virtual Storage Summary Display (FN02OVP) ............................ 188</td>
</tr>
<tr>
<td>27. Start Session (FN02SCL) .............................................. 189</td>
</tr>
<tr>
<td>28. File Services Summary (FN02OPR) .................................... 189</td>
</tr>
<tr>
<td>29. Session Status/Stop (Active) Node Selection (FN03SSA) ............... 190</td>
</tr>
<tr>
<td>30. NPM Session Analysis Summary (FN02SAN) ................................ 190</td>
</tr>
<tr>
<td>31. NPM Session Analysis Summary - Application (FN03SMN) ............... 191</td>
</tr>
<tr>
<td>32. NPA LUL Display and Management (FN02OCC) ............................ 192</td>
</tr>
<tr>
<td>33. NCP Management Resource List - (FN03RSL) ............................. 192</td>
</tr>
<tr>
<td>34. NCP Management Network Start - (FN03STD) ............................ 193</td>
</tr>
<tr>
<td>35. NCP Management Network Start - (FN03STM) ............................ 193</td>
</tr>
<tr>
<td>36. File Services Summary (FN02OPR) .................................... 194</td>
</tr>
<tr>
<td>37. Network Review (FN03RVM) ............................................ 194</td>
</tr>
<tr>
<td>38. Network Review Data (FN03RVP) ...................................... 195</td>
</tr>
<tr>
<td>39. VTAM Verification (FN01VTM) .......................................... 195</td>
</tr>
<tr>
<td>40. View VTAM Environment Data (FN03VE1) ................................ 196</td>
</tr>
<tr>
<td>41. LAN Bridge Control (FN02LBB) ...................................... 197</td>
</tr>
<tr>
<td>42. LAN Bridge Selection (FN02LBS) ..................................... 197</td>
</tr>
<tr>
<td>43. LAN Bridge Collection Control (FN03LBC) ............................. 198</td>
</tr>
<tr>
<td>44. File Services Summary (FN02OPR) .................................... 198</td>
</tr>
<tr>
<td>45. LAN Bridge Selection (FN02LBS) ..................................... 199</td>
</tr>
<tr>
<td>46. LAN Review (FN02LBM) ................................................ 200</td>
</tr>
<tr>
<td>47. LAN Bridge Data Detail (FN03LBR) ................................... 200</td>
</tr>
<tr>
<td>48. LAN Segment Control (FN02LSB) ..................................... 201</td>
</tr>
</tbody>
</table>
Tables

1. NPM Information on the Internet ........................................... xxi
2. NPM Data Sets in MVS .......................................................... 3
3. Hexadecimal Representations of Special Characters in NPM .............. 5
4. FNMINIT - NPM V1R6 to NPM Version 2 Release 4 Statement and Parameter Comparison .................................................. 78
5. Mapping NPM V1R6 DEFAULTS Statement Parameters to NPM Version 2 .......................................................... 78
6. Mapping NPM V1R6 NPM Statement Parameters to NPM Version 2 ........................................................................... 79
7. Mapping NPM V1R6 TASK Statement Parameters to NPM Version 2 for VTAM Statistics Collection .................................................. 79
8. FNMSRT and FNMSCMDS - NPM V2 R1, R2, R3 to NPM V2R4 Command and Parameter Comparisons .................................................. 80
9. Mapping of NPM V1R6 Command Parameters to NPM Version 2 .......................................................... 82
10. FNMINIT - NPM V2 (R1, 2, & 3) to NPM V2R4 Statement and Parameter Comparison .................................................. 87
11. FNMSRT and FNMSCMDS - NPM V2 R1, R2, R3 to NPM V2R4 Command and Parameter Comparisons .................................................. 88
12. NPM Names ........................................................................ 96
13. Configuration Statements and Values ....................................... 98
14. SMP/E USERMODs for Installing Exits and Modules .................. 134
15. Collection Interval Example ...................................................... 147
16. FNMSG Message Severity Indicators ...................................... 224
17. New and Changed Messages for NPM V2R4 .......................................................... 236
18. Begin Function Parameter List .................................................... 239
19. End Function Parameter List ...................................................... 239
20. Logon Function Parameter List .................................................. 240
21. Logoff Function Parameter List .................................................. 240
22. NAPALU Connect Function Parameter List ................................... 241
23. NAPALU Disconnect Function Parameter List ................................... 241
24. NPM Connect Function Parameter List ............................................. 242
25. NPM Disconnect Function Parameter List ............................................. 242
26. DNC Session Function Parameter List ......................................... 243
27. File Switch Function Parameter List ............................................. 243
28. Message Function Parameter List ................................................ 244
29. NPMLOG Record Function Parameter List ....................................... 244
30. VTAMLOG Record Function Parameter List ....................................... 245
31. Session Record Function Parameter List ....................................... 245
32. Review Record Function Parameter List ....................................... 246
33. SMF Record Function Parameter List ............................................. 247
34. Performance Alert Function Parameter List ..................................... 247
35. NetWare Record Function Parameter List ....................................... 248
36. Session Analysis Function Parameter List ....................................... 248
37. NPM Environment Vectors ......................................................... 249
38. Exit Routine Function Code ......................................................... 253
39. Primary Function Codes with Secondary Function Codes .................... 253
40. Exit Options ..................................................................... 254
41. Session User Data Area ............................................................ 255
42. User Security Area ................................................................. 256
43. Message Parameter List ............................................................ 257
44. WTO Descriptor and Routing Codes ............................................. 257
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>45.</td>
<td>SMF Data Areas</td>
</tr>
<tr>
<td>46.</td>
<td>PIU Data Area</td>
</tr>
<tr>
<td>47.</td>
<td>PIU Analysis Flags</td>
</tr>
<tr>
<td>48.</td>
<td>PIU Analysis Function Parameter List</td>
</tr>
</tbody>
</table>
Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, New York 10594
USA

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 Corporation
P.O. Box 12195
3039 Cornwallis
Research Triangle Park, NC 27709-2195
USA

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

Programming Interface Information

This book is intended to help the customer install and customize NetView* Performance Monitor. It contains descriptions of NetView Performance Monitor parameters and the steps to follow to install and customize NetView Performance Monitor.

This book also documents General-Use Programming Interface and Associated Guidance Information. General-Use Programming Interfaces allow the customer to write programs that obtain the services of NetView Performance Monitor. General-Use Programming Interface and Associated Guidance Information is identified where it occurs by an introductory statement to a chapter or section or by the following marking:
Notices

General-Use Programming Interface

| General-Use Programming Interface and Associated Guidance Information. |

| End of General-Use Programming Interface |

Trademarks and Service Marks

The following product names are trademarks of Tivoli Systems or IBM Corporation: AIX, IBM, NetView, Tivoli Management Environment, and TME 10.

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

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

C-bus is a trademark of Corollary, Inc.

Java and Hot Java are trademarks of Sun Microsystems, Inc.

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

ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks or registered trademarks of Intel Corporation in the United States and other countries.

Other company, product, and service names mentioned in this document may be trademarks or servicemarks of others.

IBM Trademarks

The following terms, denoted by an asterisk (*) at their first occurrences in the NPM publications, are trademarks of IBM Corporation in the United States or other countries:

- APPN
- BookManager
- CICS
- Common User Access
- ESCON
- Extended Services for OS/2
- FFST/2
- First Failure Support Technology/2
- IBM
- MVS/XA
- Operating System/2
- OS/2 ES
- Personal System/2
- PS/2
- RETAIN
- System/370
- Virtual Machine/Enterprise Systems Architecture
- VTAM
- AS/400
- BookMaster
- CICS/MVS
- CUA
- ES/9000
- FFST
- First Failure Support Technology
- GDDM
- MVS/ESA
- NetView
- OS/2
- PerfDesk
- Presentation Manager
- RACF
- SAA
- SMP/E
- VM/ESA
- XGA
## Trademarks of Other Companies

The following terms, denoted by a double asterisk (**) at their first occurrences in the NPM publications, are trademarks of other companies:

<table>
<thead>
<tr>
<th>Term</th>
<th>Company Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helvetica</td>
<td>Linotype Company</td>
</tr>
<tr>
<td>IPX</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>Legent</td>
<td>Legent Corporation</td>
</tr>
<tr>
<td>NetSpy</td>
<td>Legent Corporation</td>
</tr>
<tr>
<td>NetWare</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>NetWare Loadable Module</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>NetWare Management Agent for NetView</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>NetWare for SAA</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>NLM</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>Novell</td>
<td>Novell, Inc.</td>
</tr>
<tr>
<td>OMEGAMON II</td>
<td>Candle Corporation</td>
</tr>
<tr>
<td>SAS</td>
<td>SAS Institute, Inc</td>
</tr>
<tr>
<td>Times New Roman</td>
<td>Monotype Corporation, Limited</td>
</tr>
<tr>
<td>TPX</td>
<td>Legent Corporation</td>
</tr>
</tbody>
</table>
About This Book

This book describes how to install, customize, and tune the Tivoli NetView® Performance Monitor (NPM) program.

Who Should Use This Book

This book is designed for the system programmer who will install, customize, and tune NPM. NPM installation requires the use of this book, the NPM Program Directory, and NPM Concepts and Planning.

Programming and system knowledge are required to install NPM. The person installing NPM should also have the following skills:

- Knowledge of System Management Facility (SMF)
- Ability to code job control language (JCL)
- Knowledge of System Modification Program/Extended (SMP/E*)
- Knowledge of Virtual Telecommunications Access Method (VTAM*)
- Knowledge of Virtual Sequential Access Method (VSAM) for session and review files
- Ability to operate system editing facilities
- Knowledge of assembler language

How This Book Is Organized

NPM Installation and Customization consists of 14 chapters. Part 1, “Installing NPM” on page 1, describes the steps needed to install NPM and contains the following chapters:

- Chapter 1, “Preparing for Installation” on page 3, contains information you should review before installing NPM.
- Chapter 2, “Installing NPM with the NPM Initialization Program” on page 11, describes the interactive, menu-driven NPM initialization program that enables you to install NPM without manually coding the NPM initialization parameters.
- Chapter 3, “Installing a Simple Version of NPM” on page 21, describes how to install a minimal version of NPM quickly.
- Chapter 4, “Installing NPM Manually” on page 27, describes how to perform a normal NPM installation using the NPM initialization statements. You can code these statements to meet your specific requirements, then use them to install all of NPM’s functions.
  
  Note: If you plan to use the NPM initialization program to install NPM, you do not need to read this chapter.

- Chapter 5, “Migrating to NPM V2R4 from V1 Releases” on page 75, describes how to migrate to NPM Version 2 Release 4 from Version 1 releases. It also describes how to install NPM Version 2 Release 4 and have it collect the same data you are currently collecting with a previous release. This chapter also describes the mapping changes that are required when migrating from NPM Version 1 Release 6 to NPM Version 2 Release 4.
Chapter 6, “Migrating to NPM V2R4 from Previous V2 Releases” on page 85, describes how to migrate to NPM Version 2 Release 4 from earlier Version 2 releases.

Chapter 7, “Installing the NPM Desk/2 User Interface” on page 89, contains installation instructions for NPM Desk/2.

Chapter 8, “Installing NPM Netware Resource Collection” on page 117, contains procedures for installing the NPM NetWare Agent and the associated code which allows for NPM to perform data collection on NetWare Resources.

Chapter 9, “Installing NPM Exits” on page 121, contains procedures for installing VTAM and SMF exit routines. The chapter also describes:

– How to install the VSAM local shared resource option to improve the performance of NPM VSAM files
– How to use SMP/E and NPM-provided MVS user modifications (USERMODs) to install message tables and VSAM definitions

Part 2, “Customizing NPM” on page 137, describes how to customize NPM, optimize its performance, and reduce its storage requirements. Part 2 contains the following chapters:

– Chapter 10, “NPM Data Collection” on page 141, contains information about how to use NPM commands to automate NPM data collection. It explains how to use FNMSRT, how to use, write, and process NPM EXECs, and how to issue commands from the system console.

– Chapter 11, “Starting and Stopping NPM and Verifying Installation” on page 175, describes how to start and stop NPM, how to verify NPM installation, and how clean up NPM’s common storage area (CSA) control blocks.

– Chapter 12, “Tuning NPM” on page 209, provides suggestions for improving NPM performance and reducing storage requirements.

– Chapter 13, “Customizing NPM Messages” on page 223, explains how to use general-use programming interfaces to modify the text and destination of the messages that NPM issues. It also explains how to determine message destinations and under what conditions messages are sent to these destinations.

– Chapter 14, “Customizing NPM Processing with Installation-Wide Exits” on page 237, describes the NPM installation-wide exit and the path information unit (PIU) analysis exit. The chapter describes the functions provided for each programming interface. It lists when the functions are called and the parameters that are passed.

A glossary, an index, and a reader’s comment form follow the chapters. A product bibliography is included in NPM Concepts and Planning.

Conventions Used in This Book

Information in this book applies to MVS unless otherwise stated.

**MVS** indicates information applying to MVS/XA* and MVS/ESA*.

The term data set is used in this book to refer to members of partitioned data sets for MVS.
In NPM panels, statements, and commands, the term LU means a terminal. Unlike the SNA definition, this definition of LU does not include applications. In NPM, the abbreviation APPL means application program.

New in NPM Version 2 Release 4

This section describes the new features in Tivoli NPM Version 2 Release 4 (V2R4).

New TCP/IP Session Collection Capability

A TELNET server is a gateway that enables clients and workstations on a TCP/IP network to access applications in an SNA network.

In addition to collecting session data on SNA sessions, NPM can now collect response time data for TELNET sessions to a mainframe application, including the IP network segment. This enhanced session collection capability is available for TN3270E servers that reside on the same host as NPM and are running OS/390 Release 5 or higher.

Multinode Persistent Sessions (MNPS) Data Collection

VTAM uses multinode persistent sessions (MNPS) to preserve sessions across application outages where hosts are connected through the S/390 coupling facility. MNPS provides for the recovery of hardware, VTAM, MVS, or other software failures with minimal impact to the user.

NPM collects the following data about MNPS application activity:

- Bytes written to CF for the MNPS application
- Percentage of blocks that are used by this application
- Percentage of blocks that are used by all MNPS applications
- Number of sessions recovered
- Number of HPR pipes recovered

Communications Storage Manager (CSM) Data Collection

Communications Storage Manager (CSM) is a component of VTAM that enables host applications to share data with VTAM and other CSM users without having to physically copy the data. CSM reduces CPU utilization and optimizes system performance during the transfer of bulk data by enabling applications to share buffers.

NPM can now monitor the use of storage managed by CSM. You can use NPM to monitor the following for specific applications that are using CSM-managed storage pools:

- Storage that is allocated to each pool
- Buffer that is available to each pool
- Buffer that is in use for each pool
- Fixed and ECSA storage that is in use
Rapid Transport Protocol (RTP) Statistics Support
The Rapid Transport Protocol (RTP) is the highly efficient mechanism used by High Performance Routing (HPR) to perform end-to-end error recoveries (selective retries) and flow control. HPR brings the improvements necessary to enable SNA to exploit the new high-speed (multi-megabit) links with very low bit error rates on LANs and WANs. It also enables SNA to exploit new switched virtual networking technologies, such as frame relay, SMDS, and ATM.

NPM can monitor the following RTP data:
- Bytes that are sent and received over the RTPs
- Path-switch attempts
- Active LU-LU sessions that are using RTP
- Hop-count statistics
- PIUs that are sent and received over the RTPs

What’s New in This Book
The following changes have been made to this book for NPM Version 2 Release 4:
- The following appendixes have been moved to NPM Reference:
  - NPM Initialization Statement Reference
  - NPM Samples
  - NPM Initialization Statement Parameter Panel Cross-Reference
  - NPM Macros
  - NetView Synergy Interface
- Chapter 12, “Tuning NPM” on page 209 has been updated to include the following:
  - A description of the new SAMP statement.
  - A description of the new TELNET parameter for the NPM statement.

Using Syntax Diagrams
This book uses syntax diagrams to illustrate the required syntax of commands and statements. This section describes how to use these diagrams.

Reading Syntax Diagrams
The syntax diagrams start with double arrowheads on the left (leftrightarrow) and move along the main line until you end with two arrowheads facing each other (leftrightarrow). To use a syntax diagram, follow any path from left to right. When you reach the end of a line, go to the beginning of the next line, if there is one. For whatever path you choose, code every item that appears on the path. All spaces, commas, and other characters are significant.

Abbreviating Keywords
In a syntax diagram, keywords are all or partly in uppercase. Where an abbreviation is possible, the abbreviation is shown in uppercase and the rest of the keyword is shown in lowercase. Variable values that you provide are shown in italics.
The previous diagram shows that you can code the SEND command in either of the following ways:

SE ‘message text’
SEND ‘message text’

Parameters

The following are types of parameters used in syntax diagrams:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>Required parameters are displayed on the main path.</td>
</tr>
<tr>
<td>Optional</td>
<td>Optional parameters are displayed below the main path.</td>
</tr>
<tr>
<td>Default</td>
<td>Default parameters are displayed above the main path.</td>
</tr>
</tbody>
</table>

Parameters are classified as keywords or variables. Keywords are displayed in uppercase letters and can be typed in uppercase or lowercase. For example, a command is a keyword.

Variables are italicized, appear in lowercase letters, and represent names or values you supply. For example, a file name is a variable.

In the following example, NSASOLCT is a command, the variable parameter is ncp_name, the keyword is CLOCK, and CLOCK’s variable is time. You replace the variables with your own values.

NSASOLCT ncp_name,CLOCK=time

Required Parameters

A stack of parameters with the first parameter on the main path means that you must choose only one from the stack.

In the following example, the required parameters are LU, GROUP, CDRM, or APPL.

SESSCOLL LU=luname
GROUP=gname
CDRM=cname
APPL=applname

Default and Optional Parameters

Items shown above the main line are defaults. Items shown below the main line are optional.

SEND ‘message text’

The previous diagram shows that if you do not specify a host, HOST=LOCAL is used. To send a message to a different host, for example NYC, code the SEND command as follows:

SEND ‘message text’,HOST=NYC
About This Book

Repeating Parameters
Items that can be repeated are shown as follows:

```
>> CRITERIA 'expression'
```

The previous diagram shows that the following are all valid ways of coding the CRITERIA statement:

- CRITERIA
- CRITERIA 'expression'
- CRITERIA 'expression1','expression2'
- CRITERIA 'expression1','expression2','expression3'
- CRITERIA 'expression1','expression2','expression3','expression4'

and so on.

Reading Fragments
Syntax diagrams can contain fragments. A fragment is indicated by vertical bars with the name of the fragment between the bars. The fragment appears after the main diagram, as shown in the following example.

```
>>> SEnd 'message text' ─ Route ─
```

Route:
```
,ROUTE=GLOBAL
,ROUTE=ALL
,ROUTE=CONSOLE
,ROUTE=EXTERNAL
```

The previous diagram shows that the following are all valid ways of coding the SEND command:

- SE 'message text'
- SE 'message text',ROUTE=GLOBAL
- SE 'message text',ROUTE=ALL
- SE 'message text',ROUTE=CONSOLE
- SE 'message text',ROUTE=EXTERNAL

Long Syntax Diagrams
When more than one row is needed for a syntax diagram, the continued line ends with a single arrowhead (▼) and following line begins with a single arrowhead (▲), as shown in the following example.

```
>>> SELECT DATES=(50-01-01,49-12-31)
```

```
,DATES=(start,end)
,TIMES=(00:00:00,24:00:00)
,DAILY=Yes
,DAILY=No
```
Where to Find More Information

The following list contains the names and order numbers of the information about NPM.

**Evaluation and Planning**

*NPM Concepts and Planning*  
GH19-6961-03

**Installation and Customization**

*NPM Installation and Customization*  
SH19-6964-03

**Operation**

*NPM Desk/2 User's Guide*  
SH19-6963-03
*NPM Graphic Subsystem*  
SH19-6967-00
*NPM Messages and Codes*  
SH19-6966-03
*NPM Reference*  
SH19-6965-03
*NPM User's Guide*  
SH19-6962-03

**Diagnosis**

*NPM Diagnosis*  
LY19-6381-03

**Softcopy CD-ROM**

*Tivoli NPM V2R4 Product Library Kit*  
SK2T-6949-00
*(available with the product)*

*IBM Networking Systems Softcopy Collection Kit*  
SK2T-6012-21
*(available after June 1998)*

*IBM Online Omnibus Edition MVS Collection Kit*  
SK2T-0710-23
*(available after June 1998)*

*IBM Online Omnibus Edition OS/390 Collection Kit*  
SK2T-6700-09
*(available after June 1998)*

**Packaging**

Standard binder  
SX80-0258-00
Binder inserts and labels  
GH19-4490-01
A package containing the NPM manuals. It also contains three binders and related binder inserts and labels.  
SBOF-8454-02

You can read more about IBM, Tivoli, and NPM on Internet home pages. The uniform resource locations (URLs) are:

*Table 1. NPM Information on the Internet*

<table>
<thead>
<tr>
<th>Home Page</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tivoli NPM for OS/390</td>
<td><a href="http://www.tivoli.com">http://www.tivoli.com</a></td>
</tr>
<tr>
<td>IBM information</td>
<td><a href="http://www.ibm.com">http://www.ibm.com</a></td>
</tr>
</tbody>
</table>

About This Book  xxi
Part 1. Installing NPM

Chapter 1. Preparing for Installation ........................................... 3
  Working with NPM Samples .................................................. 3
  Rules for Coding NPM Initialization Statements ......................... 4
  Special Characters on Non-US Keyboards .................................. 5
  Managing NPM Security ....................................................... 6
    Logon Processing .......................................................... 6
    Minimal Security .......................................................... 6
    Normal Security ........................................................... 7
    RACF Security ............................................................. 7
    User-Defined Security .................................................... 7
  Functional Security .......................................................... 7
  Graphic Subsystem Security .................................................. 8
  Console Security ............................................................. 8
  Defining Security for NPM-to-NPM Communication ........................ 8
    Sending Console Commands to a Remote NPM ............................ 8
    Using Remote Profiles .................................................... 8
  Using RACF with NPM .......................................................... 9
    Protecting NPM Data Sets ................................................ 9
    Controlling Access to NPM from Specific Terminals .................... 9
    Controlling Access to NPM ................................................. 9

Chapter 2. Installing NPM with the NPM Initialization Program ................ 11
  General Information ........................................................ 11
  Using the NPM Initialization Program .................................... 14
    Using PF Keys ............................................................ 14
    Using Commands .......................................................... 14
    Getting Help .............................................................. 14
  Getting Started ............................................................. 14
  Using the Minimum or Normal Initialization Options .................... 17
    Minimum Initialization .................................................. 17
    Normal Initialization .................................................... 17
  Installing the Minimum or Normal NPM Initialization Options .......... 18
  Retrieving Initialization Parameters .................................... 19

Chapter 3. Installing a Simple Version of NPM ................................ 21

Chapter 4. Installing NPM Manually ............................................. 27
  Installation Procedure (for MVS) .......................................... 27
  Step 1. Allocate NPM Data Sets ............................................. 28
  Step 2. Define NPM for MVS ................................................ 29
  Step 3. Define NPM to VTAM ................................................ 30
  Step 4. Define FNMINIT Initialization Statements ........................ 32
    Step 4a. Set General FNMINIT Definitions .............................. 32
    Step 4b. Install Network Collection .................................... 34
    Step 4c. Install Network Session and Gateway Accounting ............ 35
    Step 4d. Install Session Collection ..................................... 36
    Step 4e. Install VTAM Statistics Data Collection ....................... 42
    Step 4f. Install LAN Data Collection ................................... 44
    Step 4g. Install Netware Resource Data Collection ..................... 46
    Step 4h. Install RTM Collection ........................................ 48
Chapter 1. Preparing for Installation

This chapter contains information you should review before you install NPM.

The following preparation steps should be completed prior to using the NPM initialization program, or when installing NPM manually:

- Decide on a naming convention for the various data sets that NPM uses.
- Back up any existing NPM data sets.
- Gather the necessary information for your planned NPM configuration. See *NPM Concepts and Planning* for checklists of information on which you should decide before beginning to install NPM.
- Install the NPM program tape, following the instructions in the program directory supplied with the tape for MVS.

Working with NPM Samples

When installing NPM, you update statements in sample data sets and in data sets you create. The data set names used in this book are condensed from longer names that depend upon the operating system.

For example, in MVS, FNMINIT, which is used to define your NPM configuration using NPM initialization statements, is the FNMINIT member of the FNMPARM partitioned data set. Its full name is NPM.V2R4M0.SFNMJCL1(FNMINIT).

Table 2 gives the locations of the partitioned data sets and members of partitioned data sets that you will use most often. The table provides the following information for each member:

- The name of the member
- The data definition (DD) statement, in the NPM startup job control language (JCL), that points to the partitioned data set containing the member
- The name of the data set provided with NPM where you can find a sample for the member

<table>
<thead>
<tr>
<th>Member Name</th>
<th>DD Statement</th>
<th>Sample Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNMINIT</td>
<td>FNMPARM</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMINIT)</td>
</tr>
<tr>
<td>FNMOPER</td>
<td>FNMPARM</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMOPER)</td>
</tr>
<tr>
<td>FNMPROF</td>
<td>FNMPARM</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMPROF)</td>
</tr>
<tr>
<td>FNMSTRT</td>
<td>FNMPARM</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMSTRT)</td>
</tr>
<tr>
<td>FNMLUGRP</td>
<td>FNMSCMDS</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMLUGRP)</td>
</tr>
<tr>
<td>FNMESSMF</td>
<td>FNMSCMDS</td>
<td>NPM.V2R4M0.SFNMJCL1(FNMESSMF)</td>
</tr>
</tbody>
</table>

The NPM startup JCL contains pointers to these samples. If you change the names of any samples, you must modify the startup JCL as well.
Rules for Coding NPM Initialization Statements

You can find a description and syntax diagram for each initialization statement in NPM Reference and for each command in the NPM User’s Guide. Follow the syntax diagrams when coding the statements or commands. In addition, follow the rules listed below when coding the statements:

- Statements can start in any column, but cannot extend beyond column 71. You can continue statements on a subsequent line by ending the current line with a comma and starting the next logical line with the next operand.

- There should not be any blanks within the statement text on any line, unless they are part of a string enclosed in single quotes. In noncomment statements, the first blank to the right of the statement marks the end of the statement text on that line. Any text to the right of the blank is treated as a comment. Any line that is blank between columns 1 and 72 is ignored.

- Comment statements can start with an asterisk in any column on any logical line.

- Statements can appear in any order.

- Statements are scanned for syntax before the operands are processed. If a syntax error is found in a statement that you can use more than once (for example, APPL and PROFILE), NPM issues a message and ignores the entire statement.

If a syntax error is found in a statement that you can use only once (for example, PFKEYS and NPM), NPM issues a message and uses the defaults for the parameters not yet processed for that statement. You can use these messages, which are written to FNMILOG, to help you resolve initialization problems.

The description for each statement specifies whether you can use the statement more than once.

- Keywords can be specified only once in a statement.

- Parameters can be defined as strings enclosed in single quotes. They are delimited by commas. Literal strings can be defined with or without quotes. However, if the string contains blanks or special characters, single quotes are required.

- If you want to use the default values for all parameters of a statement, do not code the statement. If you code a statement without any associated parameters, the following message is issued and all default values are taken:
  FNM622E: SYNTAX ON STATEMENT IS NOT VALID

- Names can be from one to eight characters in length. Valid characters are A–Z, 0–9, and the at (@), pound (#), and dollar ($) symbols. The first character cannot be numeric.

  Note: NetWare server names can be from two to forty-seven characters. Valid characters are A–Z, 0–9, the dash (–), and the underscore (_).
Certain parameters allow multiple values (for example, the dynamic network collection (DNC) parameter on the NPM initialization statement). Multiple values should be separated by commas and enclosed within parentheses. For example, to specify SMF and NPMLOG as destinations for DNC records, code the NPM statement as follows:

```
NPM DNC=(SMF,NPMLOG)
```

If you specify only one value, the parentheses are optional.

### Special Characters on Non-US Keyboards

NPM requires a number of special characters in addition to A–Z and 0–9. When you use the interactive panels, your keyboard input is translated into the correct hexadecimal representation through a translation table. However, when you create files such as those used for initialization statements, no translation takes place.

If you are using a non-US keyboard to create the files described in this manual, you need to be aware of the hexadecimal representation that NPM expects. Table 3 lists special characters and their hexadecimal values required by NPM. When requested to enter one of these special characters, ensure that the character on your keyboard produces the correct hexadecimal representation. For example, if the instructions say to enter a dollar ($) symbol, enter the character on your keyboard that corresponds to X'5B'.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>¢</td>
<td>Cent</td>
<td>4A</td>
</tr>
<tr>
<td>.</td>
<td>Period</td>
<td>4B</td>
</tr>
<tr>
<td>(</td>
<td>Left parenthesis</td>
<td>4D</td>
</tr>
<tr>
<td>&amp;</td>
<td>Ampersand</td>
<td>50</td>
</tr>
<tr>
<td>$</td>
<td>Dollar</td>
<td>5B</td>
</tr>
<tr>
<td>*</td>
<td>Asterisk</td>
<td>5C</td>
</tr>
<tr>
<td>)</td>
<td>Right parenthesis</td>
<td>5D</td>
</tr>
<tr>
<td>¬</td>
<td>Not symbol</td>
<td>5F</td>
</tr>
<tr>
<td>,</td>
<td>Comma</td>
<td>6B</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td>6E</td>
</tr>
<tr>
<td>:</td>
<td>Colon</td>
<td>7A</td>
</tr>
<tr>
<td>#</td>
<td>Pound</td>
<td>7B</td>
</tr>
<tr>
<td>@</td>
<td>At sign</td>
<td>7C</td>
</tr>
<tr>
<td>'</td>
<td>Single quote</td>
<td>7D</td>
</tr>
<tr>
<td>=</td>
<td>Equals</td>
<td>7E</td>
</tr>
<tr>
<td>-</td>
<td>Dash</td>
<td>60</td>
</tr>
<tr>
<td>_</td>
<td>Underscore</td>
<td>6D</td>
</tr>
</tbody>
</table>
Managing NPM Security

NPM provides the means to control access to NPM facilities. This section provides an overview of NPM security management and the security procedures that are performed when an operator logs on. It also includes the following security information:

- Functional security
- Console security
- NPM-to-NPM communications security
- Special protection with Resource Access Control Facility (RACF*)

You manage NPM security at two levels:

- Global
- Operator

Global parameters for NPM security determine the types of checks performed by NPM when an operator attempts to log on to NPM.

At the operator level, you define operator IDs and a set of security profiles for each operator who uses NPM, including operators who use NPM-to-NPM communication from remote NPMs.

Each security profile defines a level of access to NPM functions. Each operator is defined with one to eight profiles, although only one profile can be used per session. If an operator wants to use a different profile, the operator must log off and then log on again to NPM, specifying the name of the new profile.

Logon Processing

When an operator logs on to NPM, NPM verifies the logon according to the type of security processing that you have defined. NPM then determines the profile to use for the operator. If the operator does not enter a profile name when logging on, NPM uses the first profile defined to the operator. If the operator specifies a profile name, NPM checks the list of profiles defined for the operator and verifies that the profile is correct. If the operator is not authorized for the specified profile name, NPM rejects the logon attempt.

You can choose one of four different types of security processing:

- Minimal
- Normal
- RACF
- User-defined

The following sections explain the logon processing performed by NPM for each of these security types.

Minimal Security

Operators must enter both an operator ID and a password to log on. NPM verifies that the ID is defined, but does not check the password.
Normal Security
Operators must enter both an operator ID and a password.

If you change an operator’s password while NPM is running, the operator must use the updated password on the next logon attempt. You do not need to restart NPM to change passwords.

You do need to restart NPM if additional operator IDs are added. Operators who have been added can log on after the addition using their new IDs and passwords.

RACF Security
RACF is a product that provides access control by:
- Identifying and verifying the users to the system
- Authorizing access to protected resources
- Logging the detected unauthorized attempts to enter the system
- Logging the detected accesses to protected resources

With RACF security, NPM does not perform the security check. The password check is done by using the RACF service, RACROUTE. In addition to passing the operator ID and password to RACF, NPM also passes the terminal identifier and the NPM application name. The operator ID must be defined to RACF, and the password must agree with the password defined to RACF.

If the operator specifies a new password, it is automatically passed to RACF. After RACF accepts the operator ID and password, the new password is in effect for the next logon attempt.

When the NPM operator ID is the same as the Time Sharing Option (TSO) operator ID, both NPM and TSO require the same RACF password. If a new password for RACF is used and accepted by RACF, the new password is effective for all applications using the RACF operator ID. You can also use RACF security to protect data sets from unauthorized access; see “Using RACF with NPM” on page 9.

Note: NPM provides logon control using RACF with the MVS or VM System Authorization Facility (SAF) feature. Because NPM uses the SAF interface, you can use any other security product that conforms to the SAF interface rather than RACF, but always specify that you want NPM to perform RACF security, regardless of the product that you are using.

User-Defined Security
You can perform your own security check through the NPM installation-wide exit. With user-defined security, NPM calls the operator-connect function of the NPM installation-wide exit to verify the operator ID, password, and security profile name of the operator.

Functional Security
NPM checks each request an operator makes to ensure that the operator is allowed to perform the requested function. The profile the operator specifies at logon lists which functions are allowed. If operators attempt to perform a function not allowed by their profiles, NPM rejects the request and issues a specific message.
NPM Security

Graphic Subsystem Security
TSO or conversational monitor system (CMS) operators who use the NPM graphic subsystem are also subject to NPM security checking when they request real-time monitor data. They must be defined to NPM with a profile that allows the collection of data for dynamic display with the graphic subsystem.

Console Security
You can specify the operator ID and profile of the system console, through the CONSOLE statement. NPM accepts only the commands permitted by the profile.

With MVS/ESA Version 4, you can add an additional level of security for console commands. Because an operator must log on to the system console in this version, you can specify that the operator’s profile, rather than the profile defined to the console, controls the commands that can be issued from the console. In this way, you can give operators different types of system console authorization by coding several different profiles.

Defining Security for NPM-to-NPM Communication
When an NPM operator makes requests from another NPM, the security level checks are defined in the remote NPM.

First, the remote NPM checks that the operator ID is defined. If it is not, the request is rejected. If the operator ID is defined and if the security is minimal in the remote NPM, no further checking is done and the request is accepted. Otherwise, the remote NPM performs the following checks:

- If a remote profile was specified for the operator, it is used to check the request. Otherwise, the operator must provide a profile name when sending the request.
- The profile name must be defined in the remote NPM and specified for the operator ID in the remote NPM.
- The profile must allow the operator to process the request.

If any of these checks fails, NPM sends an appropriate error message to the operator and rejects the request.

Sending Console Commands to a Remote NPM
An operator can send console commands to a remote NPM, but the commands are subject to NPM security at the remote NPM.

Using Remote Profiles
You can reduce the number of profiles required by your system and also control the access of remote operators to NPM functions by using remote profiles. You can also create one set of operator and profile definitions to be used by all NPMs in a complex environment and still maintain control of all NPM-to-NPM communication.
Using RACF with NPM

With RACF, you can protect specific NPM data sets from unauthorized access or the entire system from unauthorized users. The following sections explain the optional uses of RACF with NPM.

Protecting NPM Data Sets
You can protect your NPM data sets from unauthorized access by using RACF. For example, if you are working with normal security, you should protect the NPM data set that contains NPM operator IDs and passwords.

Controlling Access to NPM from Specific Terminals
During logon processing, NPM always supplies the terminal identifier to RACF. The RACROUTE service that NPM uses to verify the operator ID and password can also verify user access to the terminal. The return code supplied by RACF indicates whether access to the terminal is allowed.

Controlling Access to NPM
To prevent any RACF-defined user from logging on to NPM, NPM passes an application identifier to RACF during the logon procedure. RACROUTE checks the access list for the NPM application and grants or denies access accordingly. The Resource Access Control Facility (RACF) Security Administrator’s Guide contains specific information for implementing RACF security.
Chapter 2. Installing NPM with the NPM Initialization Program

This chapter describes how to use the NPM initialization program to install either a simple NPM system, using the minimum initialization option, or all of NPM's primary functions, using the normal initialization option.

The NPM initialization program is an interactive, panel-driven program that enables you to install NPM without manually coding the initialization parameters. The NPM initialization program is intended to help new users of NPM.

If you have installed previous releases with the NPM initialization program, you can use the NPM initialization program to migrate to Version 2 Release 4 of NPM. If you are migrating from a previous NPM release, you should review one of the following chapters before installing this NPM release.

- Chapter 5, “Migrating to NPM V2R4 from V1 Releases” on page 75.
- Chapter 6, “Migrating to NPM V2R4 from Previous V2 Releases” on page 85.

General Information

This section contains information you should review before using the NPM initialization program.

1. After using the NPM initialization program, you can install additional features using the NPM initialization statement process. See Chapter 4, “Installing NPM Manually” on page 27 for instructions on installing NPM using the NPM initialization statements.

2. The NPM initialization program does not support the installation of the following NPM features:

   - NPM Desk/2
   - Installation-wide exits

   **Note:** See Chapter 14, "Customizing NPM Processing with Installation-Wide Exits" on page 237 for more information about installing installation-wide exits, and *NPM Concepts and Planning* for a description of the difference between manual installation and installing NPM using the NPM initialization program.

   - System Management Facility (SMF) control function

   Although selective processing of SMF data cannot be installed using the NPM initialization program, the writing of data to SMF files is supported.

3. If you make manual changes to data sets after running the NPM initialization program and then run the program again, the manual changes are lost.

4. Ensure that the following data sets have been allocated before using the NPM initialization program:

   - NPM load module library (STEPLIB)
   - NPM panel library (SFNMPNL0)
   - NCP VTAMLST data set (VTAMLST)
   - NCP RRT data set (FNMLLIB)
   - NetView data set for alerts (CNMLINK)
Using the Initialization Program

5. Make the changes described below to the following SYS1.PARMLIB data set members to define NPM for MVS:

- Update the program properties table to enable the NPM program FNMAIN to execute in storage key 6 and to be nonswappable by adding the following statement to the SCHEDxx member:

  `PPT PGMNAME(FNMMAIN)
  NOSWAP
  KEY(6)`

  You can use the default values for all other keywords on the PPT statement.

  If you plan to use NetView Access Services (NVAS), you should also authorize the NSI by adding the following statement to the same SCHEDxx member:

  `PPT PGMNAME(FNMASMAN)
  NOSWAP
  KEY(6)`

- APF-authorize the load module library for NPM. The name of the load module library for MVS is NPM.V2R4M0.SFNMLMD1. To authorize the library, update the IEAAPFxx member to include the name and volume serial (`volser`) of the NPM load module library, as shown in the following example:

  `NPM.V2R4M0.SFNMLMD1, volser,`    

- If you are using NVAS, add the following statement to the IEFSSNNxx member to define NSI as a subsystem and to build the required control blocks in CSA:

  `NSI1`

  The EXEC statement that appears in the JCL to start NSI must begin with the same character string.

  **Note:** You must IPL MVS with the CLPA option for the changes to effect.

- If you installed NPM on an operating system earlier than OS/390 V2R5 and you want to run NPM on an operating system later than OS/390 V2R5, you must run the following two sample jobs:

  - `NPM.V2R4M0.SFNMJCL1(FNMTNLNK)`
  - `NPM.V2R4M0.SFNMJCL1(FNMTNMSG)`

6. Use the following procedure to define NPM to VTAM:

   **a.** Code the following statement to define the NPM application to VTAM in an application (APPL) major node member in the VTAMLST data set:

     `name_of_npm APPL AUTH=(SPO,ACQ,CNM),PRTCT=password`

     where:

     - `name_of_npm` Is the name of the NPM application you want to define to VTAM (for example, NPM01).
     - `password` Is the name of the PRTCT password for the NPM application you want to define to VTAM (for example, NPMPWS).
Note: If you are using VTAM V3R4.1 or an earlier release, use the following syntax to code the APPL major node member:

```
name_of_npm APPL AUTH=(SPO,ACQ)
```

d. If you are using VTAM V4R3 or later, and you are going to install VTAM statistics data collection, you must authorize the interface as a CNM application to VTAM. You must also authorize the interface as a secondary programmed operator. Code the following in an application (APPL) major node member in the VTAMLST data set:

```
name_of_interface APPL AUTH=(SPO,CNM)
```

c. Ensure that VTAM messages needed by NPM are not suppressed during session collection by using one of the following methods:

- Specify SUPP=NOSUP (the default) in the VTAM start list (ATCSTRxx) in SYS1.VTAMLST. See VTAM Resource Definition Reference for more information about coding the SUPP option.
- Specify SUPP=NEVER as the message suppression class for VTAM operator messages IST097I, IST511I, IST512I, IST513I, IST984I, and IST985I. You can create a user-defined USS table and specify the USSTAB=uss_table_name operand on the APPL statement that defines NPM to VTAM in SYS1.VTAMLST. See the description of the USS macroinstruction in VTAM Resource Definition Reference for more information about changing the message suppression class of these messages.

d. If you are using session collection, and if your version of VTAM is V3R4.1 or later, concatenate the NPM load library to the VTAM load library in the VTAM start procedure, as shown in the following example:

```
//NET PROC
//VTAM EXEC PGM=ISTINM,DPRTY=(14,),REGION=496,TIME=1440
//STEPLIB DD DSN=.........,DISP=SHR
//VTAMLST DD DSN=.........,DISP=SHR
//VTAMLIB DD DSN=.........,DISP=SHR
// DD DSN=NPM.V2R4M.SFNMLMD1,DISP=SHR
//NCPLOAD DD DSN=.........,DISP=SHR
//VTAMDUMP DD DSN=.........,DISP=SHR
```

The concatenation is needed to enable NPM to use the VTAM CNM interface.

Note: If you are using VTAM V4R3 or later, and you are going to install VTAM statistics data collection, you must concatenate the NPM load library to the VTAM load library in the VTAM start procedure to enable NPM to use the VTAM CNM interface.

NPM modifies the logmode entries for its use when starting a session. NPM supports LU type 0 and LU type 2 logmode entries for 3270 devices. See VTAM Resource Definition Reference for examples of logmode entries.

e. Code the TNSTAT keyword in VTAM’s start option list to enable NPM to monitor channel-activity data. Specify the TIME=n keyword after the TNSTAT keyword as TIME=1440 (1440 minutes) to provide accurate data from channel-attached resources.

Note: To ensure that buffer pool data is accurately reflecting collection of VTAM statistics data, do NOT issue the following VTAM commands or put them in the VTAM start option list ATCSTRxx:
Using the NPM Initialization Program

The NPM initialization program is panel-driven. To page through the panels, type the character that corresponds to the option you want to choose in the entry field and press ENTER. On some panels, you also need to type some information.

Using PF Keys

At the bottom of each panel in the NPM initialization program is a list of the function keys you can use on that panel. They include the following:

- **PF1 HELP**: Displays a help panel for the initialization program panel currently on your screen.
- **PF2 SPLIT**: Splits your ISPF screen.
- **PF3 EXIT**: Returns you to the main menu.
- **PF9 SWAP**: Swaps with another ISPF screen.
- **PF12 CANCEL**: Moves backward through the panels.

Using Commands

In addition to using PF keys, you can also use the following commands from any command line in the NPM initialization program:

- **COLORS**: Enables you to view or change the colors on your screen. To use colors with the NPM initialization program, you must log on to TSO with a color logmode.
- **INTERVAL**: Enables you to view or change the NPM base interval.
- **MSG xxx**: Enables you to view help text for messages, where xxx is the message number.
- **PANELID**: Displays the name of the panel in the upper left corner of the screen.

Getting Help

If you need help while using the NPM initialization program, press PF1 to displayed a help panel. Once the help panel is displayed, you can press PF11 to access indexed help for the topic about which you need more information. The help panels explain the parameters you are defining.

Getting Started

To begin installing NPM, start the NPM initialization program by completing the following steps:

1. Log on to TSO with your TSO user ID.
2. Run ISPF.
3. Select option 6 from the ISPF/PDF PRIMARY OPTION MENU.
4. Type the following commands:
Getting Started

a. ALLOC DA('NPM.V2R4M0.SFNMINS0') FI(SYSPROC) SHR REUSE

If this command is entered correctly, the system does not return any message and the command remains displayed. Erase this command and type the following command:

b. FNMYINST NPM.V2R4M0 SFNMINS0

Notes:

a. You can concatenate the NPM initialization program data set with your normal allocations by adding it to the list of system procedure library (SYSPROC) data set names by using the following command instead of the one in Step 4-a:

ALLOC DA('dsname1'.....'npm.v2r4m0.sfnnmins0') FI(SYSPROC) SHR REUSE

If you do not concatenate all of your normal allocations in SYSPROC, they are lost. The library with the largest blocksize should be first.

You can also add the NPM initialization program data set to your TSO logon procedure SYSPROC concatenation.

b. If you want to use the Japanese national language, substitute Step 4-b with the following command:

FNMYINST NPM.V2R4M0 SFNMINS0 SFNMINSJ

Step 5. When you see the opening panel of the NPM initialization program (FNMYPLG) (see Figure 1), press ENTER to continue.

Figure 1. NPM Initialization Program Opening Panel (FNMYPLG)

Step 6. If you are installing NPM for the first time, information panels (FNMYPIT, FNMYPQR, or FNMYPVP) are displayed. Page through each panel until you reach the Main Menu panel (FNMYPMN, see Figure 3 on page 17). Press ENTER after you have read each panel.

Note: If you are migrating from a previous release of NPM, the Table Migration panel (FNMYPMG) is displayed before the Main Menu panel (FNMYPMN) (see Figure 2).
Getting Started

Figure 2. Migration Table (FNPMYPMG)

Step 7. There are two possible responses to the questions on the Table Migration panel:

- **N**—No migration takes place, the NPM initialization program ends, and you are returned to ISPF option 6.

- **Y**—Migration from a previous release of NPM takes place. NPM performs the following processes:
  
  - Information from the current ISPF tables is copied to new tables and the current tables are deleted.
  
  - After the new ISPF tables are in place, several information panels (FNMPYPT, FNMPYPQR, or FNMPYPVP) are displayed in sequence to alert you to the steps you need to take before installing NPM. Press ENTER after you have read each panel.

  - The Main Menu panel (FNMPYMN) is displayed (see Figure 3).

Step 8. You can now begin the installation process. See “Installing the Minimum or Normal NPM Initialization Options” on page 18.
Using the Minimum or Normal Initialization Options

This section describes the steps you need to perform to install a minimum or normal NPM system using the NPM initialization program.

Minimum Initialization

You can use the minimum initialization option to install a simple NPM system that collects session data by application. When performing this type of data collection, NPM collects performance data from the host. It then calculates the host, network, and operator transit times for each transaction involving the specified applications. Analyzing this data can help you determine the source of possible performance problems.

The minimum option is designed for a quick installation of NPM. It provides a base for future installation modifications as you adjust the installation parameters to meet the needs of your configuration. As you become familiar with NPM using this simple configuration, you might want to add more functions, a few at a time. This way, you can experiment with each one and make informed decisions about the features you want to use. For general information about features you can install, see NPM Concepts and Planning.

As you work with session collection by application and recognize other needs for data collection in your system, see the other planning and installation chapters in this book to learn how you can meet these needs. “Normal Initialization” on page 18 and Chapter 4, “Installing NPM Manually” on page 27 provide instructions on installing each function of NPM, either through the automated NPM initialization program or by coding the NPM initialization statements manually.
Normal Initialization

Normal initialization allows you to install the following primary NPM functions:

- Accounting data collection
- Graphic subsystem
- LAN data collection
- Network data collection
- NetWare resources data collection
- Session data collection
- VTAM statistics data collection

You can only set initialization parameters for the following functions:

- Dynamic network collection
- RTM data collection

Installing the Minimum or Normal NPM Initialization Options

To install session collection by application (minimum initialization), or to install all of NPM’s primary functions (normal initialization), complete the following procedure after you have started the NPM initialization program:

Step 1. Select option 1: Define NPM Initialization Parameters from the Main Menu (FNMYPMN).

Step 2. Select one of the following options from the next panel displayed (FNMYPPM):

- For minimum initialization, select option 1: Define Minimum Initialization Parameters. The initialization program prompts you for information, such as your applications and operator IDs.
- For normal initialization, select option 2: Define Normal Initialization Parameters. The initialization program prompts you for information, such as your NPM configuration, the NPM functions you want to install, and the commands you want to process.

Step 3. Select option 2: Define NPM data sets from the Main Menu (FNMYPMN) to define your NPM data sets. The initialization program prompts you for information about your NPM data sets.

Step 4. Select option 3: Run NPM Initialization Programs from the Main Menu (FNMYPMN) to run the initialization program. You can monitor the progress of each part of the installation job on the Initialization Status panel (FNMYPIN). If the initialization program runs successfully, you see the Initialization Completion panel (FNMYPFI). You can browse the ISPF log if you need additional information about the executed steps. By doing this, you can check the ISPF log for errors and correct them.

Step 5. Select option S: Save Initialization Information from the Main Menu (FNMYPMN) to save the parameters you have defined for your installation for later retrieval. Enter a name for the set of parameters on the next panel displayed (FNMYPSV). A list of currently saved parameter sets is displayed to help you avoid duplicate names.

Step 6. Exit the NPM initialization program from panel FNMYPXT.

Step 7. Start NPM and verify its installation. See Chapter 11, “Starting and Stopping NPM and Verifying Installation” on page 175.
Retrieving Initialization Parameters

The NPM initialization program automatically saves the last set of parameters you entered. The next time you run the program, it automatically uses the last parameters specified unless you retrieve another set. Select option R: Retrieve Initialization Information from the Main Menu (FNMYPMN) to retrieve saved initialization parameters. Enter the name of the parameter set you want to retrieve on the next panel displayed (FNMYPRE). A list of the available parameter sets is displayed to help you remember the name you used when saving the parameters.

**Note:** Before retrieving a set of initialization parameters, be sure to save the values currently in use. If you do not save them, they are overwritten by the retrieved parameters. You receive a warning message before any current values are overwritten.

If you have updated your initialization statements manually since you saved the installation parameters, those changes are not retrieved. For example, suppose you add an operator directly to the FNMOOPER data set rather than using the initialization program. If you decide later to add other support using the initialization program, your new operator is not included in the retrieved parameters. You must use the initialization program to add the operator if you want to save the information.

See *NPM Reference* for a diagram of the NPM initialization program panel flows.
Chapter 3. Installing a Simple Version of NPM

This chapter describes how to install a simple, minimal version of NPM that collects session data by application. When performing this type of data collection, NPM collects performance data from VTAM in the host. NPM then calculates the host, network, and operator transit times for each transaction involving the specified application(s). Analyzing these calculations can help you determine the source of possible performance problems.

This simple version of NPM can be installed quickly. It provides a base for future installation modifications as you adjust the installation parameters to meet the needs of your configuration. After you become familiar with NPM by using this simple configuration, you can add new functions. This allows you to experiment with each additional NPM function and make informed decisions about the features you want to use.

**Note:** Additional information about the NPM initialization statements, keywords, and other parameters described in this chapter is available in the *NPM Reference*.

Use the following procedure to install a minimum NPM systems:

**Note:** A example of how to code the NPM initialization statements in FNMINIT for a minimum system is shown in Figure 4 on page 25. The parameters on the statements in this example use the default values provided in the installation samples supplied with NPM. These example statements contain more parameters than are specified in the following procedure.

**Step 1.** You must allocate the data sets in which NPM stores data and the profiles that it uses. Use the following procedure to allocate NPM data sets:

**Step a.** Create a backup copy of data set NPM.V2R4M0.SFNMJCL1.

**Step b.** Modify the sample JCL in the FNMLOGDS member of the NPM.V2R4M0.SFNMJCL1 data set by changing all statements written in lower case to match those of your required installation.

The following example FNMLOGDS JCL shows the modifications necessary to allocate NPM data sets.
**Installing a Simple Version of NPM**

```plaintext
//FNMLOGDS  JOB (accounting,information).programmer.name, //MSGLEVEL=1,MSGCLASS=A,CLASS=A //FNMLOGDS  PROC HLQ=npm,OUT='* ',REL=v2r4m0,UNIT=SYSDA, //OSER=xxxxxx,VOLID=xxxxxx ...

//***************************************************************************** //** THE FOLLOWING FIVE DATASETS ARE OPTIONAL FOR THE NPM //** GRAPHIC SUBSYSTEM. IF THE SUBSYSTEM WILL NOT BE USED, THESE //** ALLOCATIONS CAN BE DELETED. //***************************************************************************** //***************************************************************************** //** ALLOCATE THE NPM GRAPHICS PROFILE DATASET, ONLNPROF //***************************************************************************** //ONLNPROF DD DSN=&HLQ..&REL..ONLNPROF,UNIT=&UNIT, //* VOL=SER=&VOLID,SPACE=(6160,(50,10,50)), //* DCB=(LRECL=80,BLKSIZE=6160,RECFM=FB), //* DISP=(NEW,CATLG,DELETE) ...

//***************************************************************************** //** ZERO DD DSN=&HLQ..ZERO.RECORD,DISP=(OLD,DELETE,DELETE) //***************************************************************************** //PEND // EXEC FNMLOGDS,UNIT=SYSDA,VOLID=xxxxxx

//STEP1A.SYSIN DD *
DSD OUTPUT=(ZERO)
CREATE QUANTITY=1,FILL=X'00'
END

//STEP1B.SYSIN DD *
DELETE ('npm.v2r4m0.SESS/zerodot1') CLUSTER
SET MAXCC = 0
DEFINE CLUSTER( -
NAME(npm.v2r4m0.SESS/zerodot1) -
FILE(SESS) -
VOLUMES(xxxxx) -
RECORDSIZE(400 512) -
...
...

**Note:** If you do not want to use the optional graphic subsystem supplied with NPM, you must comment out any ALLOC statements related to the graphic work data sets, as shown in this example. You can allocate the graphic work data sets later by re-running FNMLOGDS.

**Step c.** Run FNMLOGDS.

**Step d.** Verify that the FNMLOGDS job has a return code of zero (RC=0) using SDSF.

**Step 2.** Make the changes described below to the following SYS1.PARMLIB data set members to define NPM for MVS:

**Step a.** Update the program properties table to enable the NPM program FNMMAIN to execute in storage key 6 and to be nonswappable by adding the following statement to the SCHEDxx member:

```plaintext
PPT  PGNAME(FNMMAIN)
    NOSWAP
    KEY(6)
```

You can use the default values for all other keywords on the PPT statement.
If you plan to use NetView Access Services (NVAS), you should also authorize the NSI by adding the following statement to the same SCHEDxx member:

```
PPT PGMNAME(FNMASMAN)
   NOSWAP
   KEY(6)
```

**Step b.** APF-authorize the load module library for NPM. The name of the load module library for MVS is NPM.V2R4M0.SFNMLMD1. To authorize the library, update the IEAAPFxx member to include the name and volume serial (volser) of the NPM load module library, as shown in the following example:

```
NPM.V2R4M0.SFNMLMD1 volser,
```

**Step c.** If you are using NVAS, add the following statement to the IEFSSNxx member to define NSI as a subsystem and to build the required control blocks in CSA:

```
NSI1
```

The EXEC statement that appears in the JCL to start NSI must begin with the same character string.

**Note:** You must IPL MVS with the CLPA option for the changes to effect.

**Step d.** If you installed NPM on an operating system earlier than OS/390 V2R5 and you want to run NPM on an operating system later than OS/390 V2R5, you must run the following two sample jobs:

```
/bulletmed NPM.V2R4M0.SFNMJCL1(FNMTNLNK)
/bulletmed NPM.V2R4M0.SFNMJCL1(FNMTNMSG)
```

**Step 3.** Use the following procedure to define NPM to VTAM:

**Step a.** Code the following statement to define the NPM application to VTAM in an application (APPL) major node member in the VTAMLST data set:

```
name_of_npm APPL AUTH=(SPO,ACQ,CNM),PRTCT=password
```

where:

- **name_of_npm** is the name of the NPM application you want to define to VTAM (for example, NPM01).
- **password** is the name of the PRTCT password for the NPM application you want to define to VTAM (for example, NPMPWS).

**Note:** If you are using VTAM V3R4.1 or an earlier release, use the following syntax to code the APPL major node member:

```
name_of_npm APPL AUTH=(SPO,ACQ)
```

**Step b.** If you are using session collection, and if your version of VTAM is V3R4.1 or later, concatenate the NPM load library to the VTAM load library in the VTAM start procedure, as shown in the following example:
Installing a Simple Version of NPM

Step 4. Locate the following sample initialization members in the NPM.V2R4M0.SFNMLMD1 data set:

- FNMEXEC
- FNMINIT
- FNMOPER
- FNMPROF
- FNMSTRT

Step 5. Code the following parameter on the NPM initialization statement in FNMINIT:

SESSINT=nnn

where:

nnn Specifies the session interval value (any number from 1–256). NPM determines how often to collect session data by multiplying the INTERVAL value (with an interval default value of 225 seconds) by the SESSINT value. The session interval default value is 1.

Step 6. Code the following parameter on a FILE statement in FNMINIT for each data set in which you will record session data:

NAME=dd_name

where:

dd_name Specifies the data definition (DD) name of one of the following data sets:

- **FNMLOGx** The NPM collection sequential (BSAM) data set. x is any unique alphanumeric or national character that you assign. You can allocate multiple FNMLOGx data sets.
- **FNMSESx** The session data VSAM data set. x is any unique alphanumeric or national character that you assign. You can allocate multiple FNMSESx data sets.

Note: If you code both the FNMLOGx and FNMSESx data set names on the FNMINIT file statement, the same data is stored twice. Although the data received from the FNMLOGx and the FNMSESx data sets is identical, it is used for different purposes by NPM. Data from the FNMLOGx data set is used for batch reporting. Data from the FNMSESx data set is used by the NPM panels.

Step 7. Use the defaults coded in the sample members for the remaining initialization statements. You can use an operator and security profile.
already coded in FNMOPER and FNMPROF to log on to NPM, or you can add your own definitions to these samples.

Step 8. Code the SESSCOLL command in FNMSTRT to automatically start and stop session data collection.

Note: This step is optional. If you omit this step, the NPM operator can still manually enter session data collection commands.


*INITIALIZATION PARAMETERS FOR NPM - THESE WILL GET YOU STARTED *
* YOU MUST REPLACE THE ????? WITH YOUR NAMES *
*INITIALIZATION PARAMETERS FOR NPM - THESE WILL GET YOU STARTED *

SYS SECURITY=NORMAL

NPM ALERT=YES, CMDS=NONE, CONFIG=YES, EVENT=NONE, HEADER='### NPM V2.4 ###', * Customize as needed
INACTVR=NO, INTERVAL=(15,MIN), LUGRPNUM=10, NETWORK=(VSAM,SMF), NETWORKX=(VSAM,ALERT), NSA=NONE, NSI=NO, * NO for TPX, YES for NV/AS
RETRY=5, SESSION=1, SESSION=(VSAM,SMF), VTAM=(VSAM,SMF), VTAMCOLL=YES, VTAMX=(VSAM,ALERT)
BUFFERS BUFSEGS=5120

IDENTIFY THIS NPM TO VTAM:

VTAM APPLID=?????,PMI=?????
HOST NAME=?????,MAXL=1/zerodot/zerodot/zerodot
CONTROL SMF=NO * This statement still allows NPM to write to SMF
CONSOLE CNSID=/zerodot1,COMMAND=YES,OPERATOR=CONSOLE

FILE NAME=FNMSES1,MACRF=LSR,WRAP=YES * File wrapping on
FILE NAME=FNMSES2,MACRF=LSR,WRAP=YES * File wrapping on
FILE NAME=FNMREV1,MACRF=LSR,WRAP=YES * File wrapping on
FILE NAME=FNMREV2,MACRF=LSR,WRAP=YES * File wrapping on
FILE NAME=FNMLOG1,WRAP=YES * File wrapping on
FILE NAME=FNMLOG2,WRAP=YES * File wrapping on
FILE NAME=FNMVLOG1,WRAP=YES * File wrapping on
FILE NAME=FNMVLOG2,WRAP=YES * File wrapping on

Figure 4 (Part 1 of 2). Sample FNMINIT Member
Installing a Simple Version of NPM

* APPL DEFINITIONS: *
* APPL NAME=??TSO,SYN=TSO,SYNMASK=TSO00001, SYNCOUNT=20,TYPE=TSO,MAXL=50
  APPL NAME=??TSO,SYNMASK=?????,SYNCOUNT=20,TYPE=TSO,MAXL=50
  APPL NAME=?????,TYPE=CICS,MAXL=50
  APPL NAME=?????,TYPE=IMS,MAXL=50
  APPL NAME=??TPX,MAXL=50,TYPE=SM,RSP=YES
  APPL NAME=??NPM,MAXL=5
  APPL NAME=??NETV,MAXL=50,SYNMASK=?????,SYNCOUNT=20
*
* SET UP DEFAULTS: *
* DEFAULTS XRES=YES *
* SET THRESHOLDS FOR RESPONSE TIMES: *
* THRESHOLD BVAL=(1,2,3,5),HOST=(0,2),NETWORK=(0,2),OPER=(0,4) *

Figure 4 (Part 2 of 2). Sample FNMINIT Member
Chapter 4. Installing NPM Manually

This chapter describes how to perform a normal, complete NPM V2R4 installation in an MVS environment.

Notes:

1. An alternate, minimal installation procedure using the FNMINIT member is described in Chapter 3, “Installing a Simple Version of NPM” on page 21. Use the FNMINIT procedure if you want to install a minimal version of NPM V2R4 quickly.

2. Additional information about the NPM initialization statements, keywords, and other parameters described in this chapter is available in the NPM Reference.

The following installation steps are provided in this chapter:

Step 1. Allocate NPM Data Sets. See page 28.
Step 2. Define NPM for MVS. See page 29.
Step 3. Define NPM to VTAM. See page 30.
   a. Set General FNMINIT Definitions. See page 32.
   b. Install Network Collection. See page 34.
   c. Install Network Session and Gateway Accounting. See page 35.
   d. Install Session Collection. See page 36.
   e. Install VTAM Statistics Data Collection. See page 42.
   f. Install LAN Data Collection. See page 44.
   g. Install NetWare Resource Data Collection. See page 46.
   h. Install RTM Collection. See page 48.
Step 8. Install the NPM Desk/2 User Interface. See page 54.
Step 9. Install NPM NetWare Resource Collection. See page 54.
Step 11. Install NPM Exits. See page 58.
Step 12. Make Changes to Other Related Products. See page 58.

Installation Procedure (for MVS)

Use the following procedure to install NPM in an MVS environment:
Step 1. Allocate NPM Data Sets

You must allocate the data sets in which NPM stores data and the profiles that it uses. Use the following procedure to allocate NPM data sets:

Step 1. Create a backup copy of data set NPM.V2R4M0.SFNMJCL1.

Step 2. Modify the sample JCL in the FNMLLOGDS member of the NPM.V2R4M0.SFNMJCL1 data set by changing all statements written in lower case to match those of your required installation.

The following example FNMLLOGDS JCL shows the modifications necessary to allocate NPM data sets.

```
//FNMLLOGDS JOB (accounting,information),programmer.name,
// MSGLEVEL=1,MSGCLASS=A,CLASS=A
//FNMLLOGDS PROC HLQ=npm,OUT='*',REL=v2r4m0,UNIT=SYSDA,
// OSER=xxxxxx,VOLID=xxxxxx
  ...
...
...
  ///c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197 ... 97/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197
  ///c5197/c5197 THE FOLLOWING FIVE DATASETS ARE OPTIONAL FOR THE NPM
  ///c5197/c5197 GRAPHIC SUBSYSTEM. IF THE SUBSYSTEM WILL NOT BE USED, THESE
  ///c5197/c5197 ALLOCATIONS CAN BE DELETED.
  ///c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197 ... 97/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197
  ///c5197/c5197 ALLOCATE THE NPM GRAPHICS PROFILE DATASET, ONLNPROF
  ///c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197 ... 97/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197
  ///c5197/c5197 ONLNPROF DD DSN=&HLQ..ONLNPROF,UNIT=&UNIT,
  ///c5197 VOL=SER=&VOLID,SPACE=(616/zerodot,(5/zerodot,1/zerodot,5/zerodot)),
  ///c5197 DCB=(LRECL=8/zerodot,BLKSIZE=616/zerodot,RECFM=FB),
  ///c5197 DISP=(NEW,CATLG,DELETE)
  ...
...
...
  ///c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197 ... 97/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197/c5197
  //ZERO DD DSN=&HLQ..ZERO.RECORD,DISP=(OLD,DELETE,DELETE)
  ...
...
...
  //PEND
  //EXEC FNMLLOGDS,UNIT=SYSDA,VOLID=xxxxxx
  //STEP1A.SYSIN DD *
  DSD OUTPUT=(ZERO)
  CREATE QUANTITY=1,FILL=X'000'
  END
  //STEP1B.SYSIN DD *
  DELETE ('npm.v2r4m0.SESS01') CLUSTER
  SET MAXCC = 0
  DEFINE CLUSTER( -
  NAME(npm.v2r4m0.SESS01) -
  FILE(SESS) -
  VOLUMES(xxxxxx) -
  RECORDSIZE(400 512) -
  ...
...
...

Note: If you do not want to use the optional graphic subsystem supplied with NPM, you must comment out any ALLOC statements related to the graphic work data sets, as shown in this example. You can allocate the graphic work data sets later by re-running FNMLLOGDS.

Step 3. Run FNMLLOGDS.

Step 4. Verify that the FNMLLOGDS job has a return code of zero (RC=0) using SDSF.
Step 2. Define NPM for MVS

Make the changes described below to the following SYS1.PARMLIB data set members to define NPM for MVS:

- **SCHEDxx**—Update the program properties table to enable the NPM program FNMMMAIN to execute in storage key 6 and to be nonswappable by adding the following statement to the SCHEDxx member:

  ```
  PPT  PGMNAME(FNMMMAIN)
  NOSWAP
  KEY(6)
  ```

  You can use the default values for all other keywords on the PPT statement.

  If you plan to use NetView Access Services (NVAS), you should also authorize the NSI by adding the following statement to the same SCHEDxx member:

  ```
  PPT  PGMNAME(FNMASMAN)
  NOSWAP
  KEY(6)
  ```

  See *MVS/XA System Programming Library: Initialization and Tuning* for more information about the SCHEDxx member of the SYS1.PARMLIB data set.

- **IEAAPFxx**—APF-authorize the load module library for NPM. The name of the load module library for MVS is NPM.V2R4M0.SFNMLMD1. To authorize the library, update the IEAAPFxx member to include the name and volume serial (volser) of the NPM load module library, as shown in the following example:

  ```
  NPM.V2R4M0.SFNMLMD1  volser,
  ```

  If you are going to use the NPM graphic subsystem, RTM data collection, or SMF control functions, authorize the NPM LPA library NPM.V2R4M0.SFNMLPA1 by adding the following statement to the IEAAPFxx member of SYS1.PARMLIB:

  ```
  NPM.V2R4M0.SFNMLPA1  volser,
  ```

  You must authorize this data set when it is added to the LPALSTxx member. See *MVS/XA System Programming Library: Initialization and Tuning* for more information.

- **IEFSSNxx**—If you are using NVAS, add the following statement to the IEFSSNxx member to define NSI as a subsystem and to build the required control blocks in CSA:

  ```
  NSI1
  ```

  The EXEC statement that appears in the JCL to start NSI must begin with the same character string.

  **Note:** You must IPL MVS with the CLPA option for the changes to effect.

- If you installed NPM on an operating system earlier than OS/390 V2R5 and you want to run NPM on an operating system later than OS/390 V2R5, you must run the following two sample jobs:
  - NPM.V2R4M0.SFNMJCL1(FNMTNLNK)
  - NPM.V2R4M0.SFNMJCL1(FNMTNMSG)
Step 3. Define NPM to VTAM

Use the following procedure to define NPM to VTAM:

Step 1. Code the following statement to define the NPM application to VTAM in an application (APPL) major node member in the VTAMLST data set:

```
name_of_npm  APPL AUTH=(SPO,ACQ,CNM),PRTCT=password
```

where:

- `name_of_npm` is the name of the NPM application you want to define to VTAM (for example, NPM01).
- `password` is the name of the PRTCT password for the NPM application you want to define to VTAM (for example, NPMPWS).

Note: If you are using VTAM V3R4.1 or an earlier release, use the following syntax to code the APPL major node member:

```
name_of_npm  APPL AUTH=(SPO,ACQ)
```

Step 2. Define the VTAM minor node for NPM Desk/2 workstation (LU 6.2) support by coding the following statement in an APPL major node member:

```
name_of_lu62appl  APPL AUTH=(ACQ,NVPACE),
                  APPC=YES,
                  PARSSEQ=YES,
                  ATNLOSS=LAST,
                  DLOGMOD=logmode_name,
                  MODETAB=modetab,
                  DMINWNL=1,
                  DMINWNR=1,
                  DSESLIM=2,
                  PRTCT=password,
                  SYNCLVL=CONFIRM
```

where:

- `name_of_lu62appl` is the name of the LU 6.2 component you want to define to VTAM (for example, NPM62).
- `logmode_name` is the name of the logon mode used for NPM sessions (for example, FNML6MOD).
- `modetab` is the name of the logon mode table (for example, NPM21TAB).
- `password` is the name of the PRTCT password for the LU 6.2 component you want to define to VTAM (for example, NPMPWS).

The password in both APPL statements must be the same.

Step 3. If you are using VTAM V4R3 or later, and you are going to install VTAM statistics data collection, you must authorize the interface as a CNM application to VTAM. You must also authorize the interface as a secondary programmed operator. Code the following in an application (APPL) major node member in the VTAMLST data set:

```
name_of_interface  APPL AUTH=(SPO,CNM)
```
Step 4. Ensure that VTAM messages needed by NPM are not suppressed during session collection by using one of the following methods:

- Specify SUPP=NOSUP (the default) in the VTAM start list (ATCSTRxx) in SYS1.VTAMLST. See VTAM Resource Definition Reference for more information about coding the SUPP option.
- Specify SUPP=NEVER as the message suppression class for VTAM operator messages IST097I, IST511I, IST512I, IST513I, IST984I, and IST985I. You can create a user-defined USS table and specify the USSTAB=uss_table_name operand on the APPL statement that defines NPM to VTAM in SYS1.VTAMLST. See the description of the USS macroinstruction in VTAM Resource Definition Reference for more information about changing the message suppression class of these messages.

Step 5. If you are using session collection, and if your version of VTAM is V3R4.1 or later, concatenate the NPM load library to the VTAM load library in the VTAM start procedure, as shown in the following example:

```
//NET PROC
//VTAM EXEC PGM=ISTINM/zerodot1,DPRTY=(14,/zerodot),REGION=4/zerodot96,TIME=144
//STEPLIB DD DSN=.........,DISP=SHR
//VTAMLST DD DSN=.........,DISP=SHR
//VTAMLIB DD DSN=.........,DISP=SHR
// DD DSN=NPM.V2R4M/zerodot.SFNMLMD1,DISP=SHR
//NCPLOAD DD DSN=.........,DISP=SHR
//VTAMDUMP DD DSN=.........,DISP=SHR
```

The concatenation is needed to enable NPM to use the VTAM CNM interface.

**Note:** If you are using VTAM V4R3 or later, and you are going to install VTAM statistics data collection, you must concatenate the NPM load library to the VTAM load library in the VTAM start procedure to enable NPM to use the VTAM CNM interface.

NPM modifies the logmode entries for its use when starting a session. NPM supports LU type 0 and LU type 2 logmode entries for 3270 devices. See VTAM Resource Definition Reference for examples of logmode entries.

Step 6. Code the TNSTAT keyword in VTAM's start option list to enable NPM to monitor channel-activity data. Specify the TIME=n keyword after the TNSTAT keyword as TIME=1440 (1440 minutes) to provide accurate data from channel-attached resources.

**Note:** To ensure that buffer pool data is accurately reflecting collection of VTAM statistics data, do NOT issue the following VTAM commands or put them in the VTAM start option list ATCSTRxx:

```
MODIFY TRACE,TYPE=SMS,ID=VTAMBUF
DISPLAY BFRUSE
```

Step 7. If you plan to collect data from LAN Network Manager or from NetWare resources, define NPM's NetView program DST application to VTAM by including the following statement in an application (APPL) major node:

```
FNMSYRN APPL AUTH=CNM
```
Step 4a. Set General FNMINIT Definitions

This definition allows the NPM DST to use the communication network management interface (CNMI) to solicit data from LAN Managers or NPM NetWare Agents.

Step 8. To customize the NPM host environment for NetWare, follow the directions in the installation and configuration instructions provided by Novell.

You must also make the following modifications to the VTAM list where the NPM service point is defined:

- Configure MAXDATA as MAXDATA=521.
- Configure MODETAB as MODETAB=NPM21TAB or a functionally equivalent table definition.
- Configure DLOGMODE as DLOGMODE=FNML6MOD.

The following example shows how the VTAM list should be customized.

```
TRSPNP4 PU ADDR=/zerodot4,
IDBLK=/zerodot17,
IDNUM=E/zerodot/zerodot/zerodot3,
CPNAME=TRSPNP4,
DISCNT=NO,
MAXOUT=1,
MAXDATA=521,
MODETAB=NPM21TAB,
MAXPATH=2,
VPACING=/zerodot,
PUTYPE=2,
XID=NO,
DLOGMODE=FNML6MOD,
USSTAB=USSTAB
```

Step 4. Define FNMINIT Initialization Statements

Use the following procedures (steps 4a through 4h) to define various initialization statements in the FNMINIT data set:

Step 4a. Set General FNMINIT Definitions

Use the following procedure to set the general definitions in the FNMINIT member to meet your specific needs:

**Note:** Many of the statements and keywords described in this step have default values assigned in the FNMINIT member. However, you should review each of the following statement and keyword descriptions to ensure they meet your specific needs.

Step 1. Code the SYS statement to meet your specific NPM security needs. Examine the LOGON, MAXOPS, and SECURITY keywords to ensure they meet your specific requirements.

An example of these SYS statement keywords is shown below:

```
SYS LOGON=5, 5 TRIES TO LOGON SUCCESSFULLY
MAXOPS=50, MAX 50 NPM OPERATORS
SECURITY=RACF RACF SECURITY
```

Step 2. Specify a name in the VTAM initialization statement (in FNMINIT) to match the NPM application name as shown in the example below. This
Step 4a. Set General FNMINIT Definitions

name cannot be the same as any other NPM resource ID or the NPM user ID.

For LU 6.2 support, you must also specify a name in the VTAM initialization statement to match the LU 6.2 component name (also shown in the example below). This name cannot be the same as any other LU 6.2 resource ID or the LU 6.2 user ID.

VTAM APPLID=NPM01, NPM VTAM APPLICATION NAME
LU62=NPM62, NPM LU6.2 COMPONENT NAME
NAME='HOST 01 NPM', DESCRIPTIVE NPM HOST NAME
PASSWD=NPMPWS NPM VTAM PASSWORD

Note: If you are using VTAM V4R3 or later, and you are going to install VTAM statistics data collection, you must specify the name_of_interface in the VTAM initialization statement in FNMINIT to match the name_of_interface, as shown in the following example:

VTAM APPLID=name_of_NPM
PMI=name_of_interface,

The names you code here must match the names coded in the VTAM APPL statements.

Step 3. Code the FILE initialization statement to define a data set(s) to NPM. You can code more than one FILE statement in the FNMINIT member. Code one statement for each data set you want to define.

An example of the FILE statement is shown below:

FILE NAME=FNMREV1 REVIEW FILE 1; NETWORK, VTAM, LAN DATA
NAME=FNMREV2 REVIEW FILE 2; NETWORK, VTAM, LAN DATA
NAME=FNMLOG1 LOG DATA SET 1
NAME=FNMLOG2 LOG DATA SET 2
NAME=FNMSES1 SESSION FILE 1; SESSION COLLECTION DATA
NAME=FNMSES2 SESSION FILE 2; SESSION COLLECTION DATA
NAME=FNMNLG1 NETWARE DATA
NAME=FNMVLOG1 VTAM LOG 1
NAME=FNMVLOG2 VTAM LOG 2

Step 4. Code the NPM statement to define processing options and values for the execution of NPM. You can change the parameters of the NPM statement to meet your specific collection needs.

For example, if you want NPM to send alerts to NetView, you can specify this in the NPM statement. The following example shows how to code the ALERT keyword:

NPM ALERT=YES, SEND ALERTS TO NETVIEW
VTAMX=(NPMLOG,ALERT) VTAM DATA EXCEPTION RECORDS TO FNMLOGX/NETVIEW

Step 5. You can use the system console to pass commands to NPM. The CONSOLE statement specifies the operator ID and profile of the system console to define console security. NPM accepts only the commands permitted by the profile.

See NPM Concepts and Planning for information about console security. See the NPM User's Guide for information about using the system console.
Code the CONSOLE statement in FNMINIT to provide the system console with a profile as shown in the following example:

```
CONSOLE OPERATOR=NPM1,PROFILE=CONSOLE
```

If you are using the extended MCS console function, you must log on to the system console. Your profile controls your authorization. Code the CONSOLE statement in FNMINIT with SECURITY=YES.

**Step 4b. Install Network Collection**

In this step, you define the most important initialization statements for the NPM network collection function. Define the following keywords on the NPM statement:

- Code the NETWORK keyword to specify the default destination of network records. The NETWORK keyword uses the following syntax:

```
NETWORK=(NPMLOG,VSAM)
```

where:

- **NONE** Specifies that there is no default destination for network performance records. If NONE is coded, you must specify a destination on the NETCOLL command.
- **NPMLOG** Specifies that network performance records are written to the data sets defined by the FILE initialization statement as FNMLOGx.
- **SMF** Specifies that network performance records are written to SMF (MVS only).
- **VSAM** Specifies that network performance records are written to the data sets defined by the FILE initialization statement as FNMREVx.

- Code the NETWORKX keyword to specify the default destination of network monitor exception data. The NETWORKX keyword uses the following syntax:

```
NETWORKX=(NPMLOG,VSAM,ALERT)
```

where:

- **ALERT** Specifies that default network monitor exception data is formatted into alerts and sent to the NetView program if you also code ALERT=YES on the NPM statement.
- **GLOBAL** Specifies that default network monitor exception data messages are broadcast to all NPM operators who are authorized with GLOBAL=YES. If no operators are logged on, the message goes to the system console.
NONE Specifies that there is no default destination for network monitor exception data records. If NONE is coded, you must specify a destination on the NETCOLL command.

NPMLOG Specifies that default network monitor exception data records are written to the data sets defined by the FILE initialization statement as FNMLOGx.

SMF Specifies that default network monitor exception data records are written to SMF (MVS only).

VSAM Specifies that default network monitor exception data records are written to the data sets defined by the FILE initialization statement as FNMREVx.

- Code the INTERVAL keyword to specify the base (smallest) interval over which network data can be collected. NPM then calculates six additional values, each value being double the previous value. The INTERVAL keyword uses the following syntax:

```
INTERVAL=(225,SEC)
```

where:

- `nnn` Specifies the number of minutes or seconds in the base interval.
- `MIN` Specifies that `nnn` is in minutes.
- `SEC` Specifies that `nnn` is in seconds.

The value you code for the INTERVAL keyword also affects the frequency of other types of data collection, such as session data and VTAM data.

**Step 4c. Install Network Session and Gateway Accounting**

Code the network session accounting (NSA) keyword on the NPM statement to specify the destination of both NSA and network gateway accounting (NGA) records. For NSA this keyword specifies the destination of network accounting records for boundary-attached LUs. For NGA this keyword specifies the destination of network accounting records for sessions passing through a gateway.

The NSA keyword uses the following syntax:

```
NSA=NPMLOG
```

where:

- NONE Specifies that no network accounting records are to be written.
- NPMLOG Specifies that network accounting records are written to the data sets defined by the FILE initialization statement as FNMLOGx.
Step 4d. Install Session Collection

SMF Specifies that network accounting records are written to SMF (MVS only).

Step 4d. Install Session Collection

Session data is performance data that is collected from VTAM in the host. You can collect session data for LUs, IP addresses, groups of LUs, nodes, and applications. Session data includes both host and network response times. See NPM Concepts and Planning for general information about session data collection.

Use the following procedure to install NPM session collection:

Step 1. Define the following keywords on the NPM statement:

- The HOSTCOLL keyword specifies whether this NPM supports data collection from the host. Code the HOSTCOLL keyword as shown in the following example:
  
  HOSTCOLL=YES
  
  where:
  
  YES Specifies that you want NPM to reserve common storage area (CSA) buffers that are needed for session data collection.

- The TELNET parameter enables you to collect data for TN3270 and TN3270E sessions by specifying a client IP address. Code the TELNET parameter using the following format:
  
  TELNET=YES
  
  The following conditions apply to the TELNET parameter:
  
  - The TELNET parameter is valid only if NPM is running on OS/390 V2R5 or later operating systems.
  
  - If the TELNET parameter is set to YES, check that the sample job NPM.V2R4M0.SFNMJCL1(FNMTNMSG) has already been run. If the sample job has not been run, run it before you execute the command.
  
  - If the TELNET parameter is set to YES, you must also set the HOSTCOLL and VTAMCOLL parameters to YES.

- Code the INTERVAL keyword (if you have not already coded it for network collection) to specify the base (smallest) interval over which network and VTAM statistics data can be collected. This value is multiplied with the SESSINT value to get a session collection interval. The INTERVAL keyword uses the following syntax:
  
  INTERVAL=(nnn,MIN)  (nnn,SEC)

  where:
  
  nnn Specifies the number of minutes or seconds in the base interval. NPM then calculates six additional values, each value being double the previous one.
Step 4d. Install Session Collection

MIN Specifies that \( nnn \) is in minutes.
SEC Specifies that \( nnn \) is in seconds.

- The SESSINT keyword specifies how many base intervals to wait before collecting session data. Code the SESSINT keyword as shown in the following example:

\[
\text{SESSINT} = nnn
\]

where:

\( nnn \) NPM determines how often to collect session data by multiplying the INTERVAL value by the SESSINT value. The valid range is 1–256.

- Code the SESSION keyword to specify the default destination of session interval records created by the session data collection feature of NPM. The SESSION keyword uses the following syntax:

\[
\text{SESSION}= (\text{NPMLOG, VSAM})
\]

where:

NONE Specifies that there is no default destination for session interval records. If NONE is coded, you must specify a destination on the SESSCOLL command.
NPMLOG Specifies that session interval records are written to data sets defined by the FILE initialization statement as FNMLOGx.
SMF Specifies that session interval and summary records are written to SMF (MVS only).
VSAM Specifies that session interval records are written to data sets defined by the FILE initialization statement as FNMSESx. Session interval records are written when a session collection interval expires. To enable online analysis of session data, you must specify VSAM as one of the destinations for session data.

- Code the SESSIONX keyword to specify the default destination of session monitor exception data. The SESSIONX keyword uses the following syntax:

\[
\text{SESSIONX}= (\text{NPMLOG, VSAM, ALERT})
\]

where:
Step 4d. Install Session Collection

ALERT  Specifies that session monitor exception data is formatted into alerts and sent to the NetView program if ALERT=YES is coded on the NPM statement. The ALERT option sends alerts to the NetView program only for the interval number set in the DEFAULTS initialization statement.

GLOBAL Specifies that session monitor exception data messages are broadcast to all NPM operators who are authorized with GLOBAL=YES. If no operators are logged on, the message goes to the system console.

NONE Specifies that there is no default destination for session monitor exception data. If NONE is coded, you must specify a destination on the SESSCOLL command.

NPMLOG Specifies that session monitor exception data is written to data sets defined by the FILE initialization statement as FNMLOGx.

SMF Specifies that session monitor exception data is written to SMF (MVS only).

VSAM Specifies that session monitor exception data is written to data sets defined by the FILE initialization statement as FNMSESx.

• (For NVAS users) Code the NSI keyword.

NSI=YES

where:

YES  Specifies that this NPM uses the NetView synergy interface (NSI). The NSI is used to collect data from session managers such as NetView Access Services. This causes NPM to load module FNMNSI, which allows NPM to connect to the NetView synergy interface.

NO  Specifies that this NPM does not use the NSI. This occurs when your session manager supports a session switch interface similar to TPX.

Step 2. Define the following keywords on the BUFFERS statement:

• Code the BUFFNO and SESSNO keywords as shown in the following example:

BUFFNO=nnnn,SESSNO=nnnn

where:

BUFFNO=nnnn  Specifies the number of buffers in CSA to be used to collect entire PIUs and RTM data. If you collect entire PIUs or RTM data, BUFFNO=200 should be adequate. If you do not need to collect entire PIUs or RTM data, code BUFFNO=0. The number you enter is used to calculate the number of pages of storage needed and is rounded up to allocate all of the last page.
Step 4d. Install Session Collection

SESSNO=nnnn  Specifies the number of transactions likely to reach NPM simultaneously from VTAM and SMF. This number is used to calculate the number of buffers in CSA that NPM uses to hold PIUs being processed. SESSNO=200 is adequate for most systems. To save CSA, you can start with SESSNO=100 and increase the value if you lose session data. NPM's performance measurement tables contain a count of any lost data. See the NPM User's Guide for more information about the performance measurement tables.

Note: You can override the value specified on this statement by using the BUFFNO and SESSNO keywords on the MVS EXEC JCL statement when NPM is started.

- If you are using a version of VTAM earlier than V3R4M1, code the DDRBUF keyword on the BUFFERS statement to allocate buffers needed for either DDR or DFC, as shown in the following example:

  DDRBUF=nnnnnn

  where:

    nnnnn  Specifies the maximum number of entries in either the DDR or DFC table. These tables have an entry for each outstanding response PIU that was generated by the DDR or DFC function. You can set DDRBUF to any number from 128–32768. The default for DDRBUF is 128. If you do not code any application with RSP=YES on the APPL statement, set DDRBUF to 0. If you are using VTAM 3.4.1 or a later release, NPM does not use the DDR table.

Step 3. Define your applications using one or more APPL statements:

- Use the NAME keyword to specify the application name used to open the VTAM access method control block (ACB). This name is in VTAM's VTAMLST data set.

  For example, if CICS APPL is the statement in VTAMLST, then you would code the NPM APPL initialization statement as APPL NAME=CICS.

- Code the SYN keyword if you are using TSO or any application that opens an ACB (ACBNAME=) with a name other than the name you specified on the NAME keyword, such as NetView or CICS. This allows NPM to resolve the application name.

- Code the RSP keyword for each application on which you want to enable DDR or DFC, using the following format:

  RSP=YES

  where:

    YES  Specifies that you want session collection to be started for an application that has SESSH=R coded on the SESSCOLL command for DDR or DFC.

- Code the TRANSIT keyword for each application on which you want to calculate network transit time. The TRANSIT keyword formats are shown in the following example:
Step 4d. Install Session Collection

TRANSIT=DR
or
TRANSIT=DFC

where:

DR Specifies that you want NPM to calculate network transit time only for user data PIUs which have definite responses.

DFC Specifies that you want NPM to include data control PIUs in the network transit time calculation.

- Code the TYPE keyword to specify special processing for certain types of applications. The TYPE keyword uses the following syntax:

```
,TYPE=CICS
   -IM
   -IMS
   -NVAS
   -SM
   -TSO
```

CICS Defines the Customer Information Control System (CICS) application.

IMS Specifies for NPM to ignore the clear screen and unlock keyboard messages when computing transit times. NPM still counts these messages in the volume statistics.

NVAS Defines session manager applications such as NetView Access Services.

SM Defines a session manager, such as TPX or a session switch interface similar to TPX, to allow NPM to correlate the session manager relay sessions.

TSO Defines the Time Sharing Option.

Some typical APPL statements are shown in the following example:

```
APPL NAME=TSO, APPLICATION NAME TSO
   MAXL=100, MAX 100 SESSION COLLECTIONS
   TYPE=TSO, IGNORE USER CLR & UNLOCK
   SYNCOUNT=50, 50 SYNONYMS
   SYNMASK=TSO0001 TSO0001 THRU TSO0050

APPL NAME=IMS, APPLICATION IMS
   MAXL=500, MAX 500 SESSION COLLECTIONS
   TYPE=IMS IGNORE USER CLR & UNLOCK

APPL NAME=CICS, APPL CICS RUNNING IN EXCEP RSP
   MAXL=500, MAX 500 SESSION COLLECTIONS
   TYPE=CICS, IGNORE USER CLR & UNLOCK
   RSP=YES USE NPM'S DEF RSP FUNCTION
```

Step 4. Code the HOST statement as shown in the following example:

```
HOST NAME=host_name, MAXL=nnnnn, SA=nnnnn
```

where:
Step 4d. Install Session Collection

**NAME** Specifies the name of the host running NPM. The name can contain from 1–8 alphanumeric characters. If you specify the host name VTAM, the HOSTPU keyword of the VTAMLST start member is used as the host name. The host name you specify cannot be the same as the SSCP name parameter of the VTAM start member. See *ACF/VTAM Installation and Resource Definition* for information about specifying the HOSTPU parameter.

**MAXL=nnnnn** Specifies the cumulative maximum number of LUs, local to the host specified on the HOST statement, on which NPM can collect session data.

**SA=nnnnn** Identifies the subarea number of the host specified in the HOST or CDRM statement. For the SA parameter, code the subarea number for the node. All nodes defined for session collection are required to have unique subarea numbers.

**Step 5.** Code the CDRM statement as shown in the following example:

```plaintext
CDRM NAME=cdrm_name,MAXL=nnnnn,SA=nnnnn
```

where:

**NAME** Specifies the name of the VTAM subarea node in the network. Unique names are required for all HOST and CDRM initialization statements and NCP commands that are defined for session collection.

**MAXL=nnnnn** Specifies the cumulative maximum number of LUs, local to the host specified on the HOST statement, on which NPM can collect session data.

**SA=nnnnn** Identifies the subarea number of the host specified in the HOST or CDRM statement. For the SA parameter, code the subarea number for the node. All nodes defined for session collection are required to have unique subarea numbers.

**Step 6.** Code the SESSH keyword on the DEFAULTS initialization statement to start session data collection using DDR or DFC. The SESSH keyword uses the following format:

```plaintext
SESSH=Y
```

where:

**SESSH** Specifies how data is compiled and written to the session file. The default is specified by the SESSH keyword on the DEFAULTS initialization statement.

**Y** Specifies that session data is collected and written to the session data set.
Step 4e. Install VTAM Statistics Data Collection

Use the following procedure to install VTAM statistics data collection:

**Step 1.** Define the following keywords on the NPM statement:

- Code the VTAMCOLL keyword to specify whether VTAM statistics data is collected. The VTAMCOLL keyword uses the following syntax:

  ![VTAMCOLL syntax diagram](image)

  where:
  - NO Specifies that VTAM statistics data is not collected.
  - YES Specifies that VTAM statistics data is collected.

- Code the INTERVAL keyword (if it was not previously defined) to specify the base (smallest) interval over which network and VTAM data can be collected. NPM then calculates six additional values, each value being double the previous value. The INTERVAL keyword uses the following syntax:

  ![INTERVAL syntax diagram](image)

  where:
  - \( nnn \) Specifies the number of minutes or seconds in the base interval.
  - MIN Specifies that \( nnn \) is in minutes.
  - SEC Specifies that \( nnn \) is in seconds.

- Code the VTAM keyword to specify the default destination of VTAM statistics data records. You can define more than one destination on this keyword. The VTAM keyword uses the following syntax:

  ![VTAM syntax diagram](image)

  where:
  - NONE Specifies that there is no default destination for VTAM statistics data records. If NONE is coded, you must specify a destination on the appropriate VxxxCOLL command.
  - NPMLOG Specifies that VTAM statistics data records are written to the data sets defined by the FILE initialization statement as FNMLGnx.
Step 4e. Install VTAM Statistics Data Collection

SMF Specifies that VTAM statistics data records are written to SMF (MVS only).

VSAM Specifies that VTAM statistics data records are written to the data sets defined by the FILE initialization statement as FNMREx.

- Code the VTAMX keyword to specify the default destination of VTAM statistics data exception records. You can specify more than one destination on this keyword. The VTAMX keyword uses the following syntax:

```
VTAMX=(VSAM,NPMLOG,ALERT)
```

where:

- ALERT Specifies that VTAM statistics data exception records are to be formatted into alerts and sent to the NetView program if ALERT=YES is coded on the NPM statement.

- GLOBAL Specifies that VTAM statistics exception data messages are broadcast to all NPM operators who are authorized with GLOBAL=YES. If no operators are logged on, the message goes to the system console.

- NONE Specifies that there is no default destination for VTAM statistics exception data messages. If NONE is coded, you must specify a destination on the appropriate VxxxCOLL command.

- NPMLOG Specifies that VTAM statistics data exception records are to be written to the data sets defined by the FILE initialization statement as FNMLOGx.

- SMF Specifies that VTAM statistics data exception records are to be written to SMF (MVS only).

- VSAM Specifies that VTAM statistics data exception records are to be written to the data sets defined by the FILE initialization statement as FNMREx.

Step 2. Code the SAMPVHI, SAMPVMD, and SAMPVLW parameters on the SAMP statement to specify sampling intervals (in seconds) used to collect VTAM statistics data for all resources with active collection. These keywords use the following syntax:

```
SAMPVHI=90
SAMPVHI=sample_interval
SAMPVLW=20
SAMPVLW=sample_interval
SAMPVMD=45
SAMPVMD=sample_interval
```
Step 4f. Install LAN Data Collection

where:

| SAMPVHI | Specifies the sampling interval used to collect VTAM data in the high category. This category is used when VTAM data collection is performed for application data, address space data, APPN data, MNPS data, and VTAM global data. The valid range is 60-120 seconds. |
| SAMPVMD | Specifies the sampling interval used to collect VTAM data in the medium category. This category is used when VTAM data collection is performed for device data, buffer pool, and CSM data. The valid range is 30-60 seconds. |
| SAMPVLW | Specifies the sampling interval used to collect VTAM data in the low category. This category is used when VTAM data collection is performed for virtual route data and RTP data. The valid range is 10-30 seconds. |

Step 3. Code the VTAMINT parameter on the DEFAULTS statement as shown in the following example:

VTAMINT=n

where:

n Specifies the default interval number for VTAM statistics data collection commands, and determines when NPM refreshes its list of available VTAM resources. This parameter is dependent on the VTAMCOLL and VTAM parameters coded on the NPM statement. You can set the interval to any number from 1–7. The default is 1.

Step 4f. Install LAN Data Collection

Use the following procedure to install LAN data collection:

Step 1. Code the following keywords on the NPM statement:

- Code the LANCOLL keyword to specify whether data can be collected. The LANCOLL keyword uses the following syntax:

  ![LANCOLL syntax diagram]

  where:

  YES Specifies that LAN data is collected. When you specify LANCOLL=YES, NPM loads the NetView program module CNMNETV to establish NPM’s connection to the NetView program-to-program interface.

  NO Specifies that no LAN data is collected.

  You must also specify the LBRGINT and LSEGINT keywords if you need to change the default values defined for the base intervals for LAN bridge and segment data collection.

- Code the LAN keyword to specify the destination of local area network (LAN) data. This field sends all collected LAN data to the listed data destinations. The LAN keyword uses the following syntax:
Step 4f. Install LAN Data Collection

where:

NPMLOG Specifies that LAN data is written to the data set defined by the FILE initialization statement as FNMLOGx.

SMF Specifies that LAN data is written to SMF (MVS only).

VSAM Specifies that LAN data is written to the data set defined by the FILE initialization statement as FNMREVx.

- Code the LANX keyword to specify the default destination of LAN monitor exception data. The LANX keyword uses the following syntax:

where:

ALERT Specifies that LAN monitor exception data is formatted into alerts and sent to the NetView program if ALERT=YES is coded on the NPM statement. The ALERT option sends alerts to NetView program only for the interval number specified on the DEFAULTS statement.

GLOBAL Specifies that LAN monitor exception data messages are broadcast to all NPM operators who are authorized with GLOBAL=YES. If no operators are logged on, the message goes to the system console.

NPMLOG Specifies that LAN monitor exception data is written to the data set defined by the FILE initialization statement as FNMLOGx.

SMF Specifies that LAN monitor exception data is written to SMF (MVS only).

VSAM Specifies that LAN monitor exception data is written to the data set defined by the FILE initialization statement as FNMREVx.

Step 2. Modify the QCB keyword on the BUFFERS statement to reflect the maximum number of LAN resources you expect to define to NPM. The QCB default calculation assumes that the maximum number of LAN resources is 200. If you do not use the QCB calculated default, you must calculate 1 QCB for each LAN resource on your system.
Step 4g. Install NetWare Resource Data Collection

Use the following procedure to install NetWare resource collection data:

Step 1. Code the following keywords on the NPM statement:

- Code the NWCOLL keyword to specify whether NetWare resources data can be collected. The NWCOLL keyword uses the following syntax:

```
[NWCOLL=NO] [NWCOLL=YES]
```

where:

YES Specifies that NetWare resources data is collected. When you specify NWCOLL=YES, NPM loads the NetView program module CNMNETV to establish NPM’s connection to the NetView program-to-program interface.

NO Specifies that data for NetWare resources data is not collected.

- Code the LWG keyword to specify the default destination of NetWare data. This field sends all the collected data to the listed data destinations. The LWG keyword uses the following syntax:

```
[LWG=(NPMLOG,VSAM)]
```

where:

NONE Specifies that there is no default destination for NetWare resources data. If NONE is coded, you must specify a destination on the LWGCOLL command in FNMSTRT. See “LWGCOLL” on page 154 for more information.

NPMLOG Specifies that NetWare resources collection data is written to the data set defined by the FILE initialization statement as FNMLLOGx.

SMF Specifies that NetWare resources collection data is written to SMF (MVS only).

VSAM Specifies that NetWare resources collection data is written to the data set defined by the FILE initialization statement as FNMLWGx.

- Code the LWGX keyword to specify the destination of NetWare resources exception data. The LWGX keyword uses the following syntax:
Step 4g. Install NetWare Resource Data Collection

where:

**ALERT**  Specifies that NetWare monitor exception data is formatted into alerts and sent to the NetView program if ALERT=YES is coded on the NPM statement. The ALERT option sends alerts to the NetView program only for the interval number specified on the DEFAULTS statement.

**GLOBAL**  Specifies that NetWare monitor exception data messages are broadcast to all NPM operators who are authorized with GLOBAL=YES. If no operators are logged on, the message goes to the system console.

**NONE**  Specifies that there is no default destination for monitor exception data for NetWare resources collections. If NONE is coded, you must specify a destination on the LWGCOLL in FNMSTRT. See “LWGCOLL” on page 154 for more information.

**NPMLOG**  Specifies that NetWare monitor exception data is written to the data set defined by the FILE initialization statement as FNMLWGx.

**SMF**  Specifies that NetWare monitor exception data is written to SMF (MVS only).

**VSAM**  Specifies that NetWare monitor exception data is written to the data set defined by the FILE initialization statement as FNMLWGx.

**Step 2.** If you need to change the default values defined for the base intervals for NetWare resources data collection, you must also specify the LWGIN T parameter on the DEFAULTS statement in FNMINIT. The following example shows how to code the LWGIN T parameter.

```
DEFAULTS DATE=1/zerodot2/zerodot98, DATE FORMAT
        GTF=NO, DO NOT PASS PIUs TO GTF
        INTERVAL=1, INTERVAL NUMBER FOR NETCOLL CMDS
        LWGIN T=2, NETWARE DATA COLLECTION INTV
...
```

**Step 3.** Modify the QCB keyword on the BUFFERS statement to reflect the maximum number of NetWare resources you expect to define to NPM. If you do not use the QCB calculated default, you must calculate 1 QCB for each NetWare resource on your system.
Step 4h. Install RTM Collection

Use the following procedure to install RTM collection:

Step 1. Code the following keywords on the NPM statement:
   - Ensure that HOSTCOLL=YES is coded to enable RTM data collection.
   - Code the RTM keyword to specify where RTM data should be logged. The data set must be allocated and defined to NPM.
   - Code the SYNCH and SESSINT keywords to cause session data to be collected at the same time you have instructed the NetView program to collect RTM data. This optional step enables you to correlate session and RTM data.

Step 2. Code the BUFFNO keyword on the BUFFERS statement:
   BUFFNO=nnnn
   where:
   nnnn Specifies the number of buffers to create in CSA that are used to collect RTM data and entire PIUs. If you collect RTM data or entire PIUs, BUFFNO=200 should be adequate.

Step 3. Code the RTM keyword on the CONTROL statement:
   RTM=YES
   where:
   YES Specifies that you want NPM to process RTM SMF records that are written by the NetView program. The RTM keyword on the NPM statement determines where the reformatted RTM records are written.

Step 5. Define NPM Operators

NPM operators use NPM's online panels to issue commands and view data summaries. You must identify operators to NPM and authorize them for the functions they can use. See the NPM User’s Guide for more information about using NPM's online panels.

Use the following procedure to define NPM operators:

Step 1. Code an OPERATOR statement in FNMOPER for each operator who uses NPM. The following example shows how to define three operators to NPM:

```
OPERATOR NAME=BILLT, MASTER CONTROLLER
    PASSWD=A1493Z,
    PROFILE=MSTCNTL
*
OPERATOR NAME=CONSOLE, SYSTEM OPERATOR
    PASSWD=D9242P,
    PROFILE=PROFILEP
*
OPERATOR NAME=BOTHALL, HOST and NPM Desk/2 OPERATOR
    PASSWD=NPMPW,
    PROFILE=PROFILEW
```

In this example, two operators (BILLT and CONSOLE) are defined for the host, and one operator (BOTHALL) is defined for both the host and the
Step 6. Define the NPM Start Procedure (FNMSNPM)

NPM Desk/2 workstation. These profiles define the NPM functions that the operator can use.

**Step 2.** Code one or more PROFILE statements in FNMPROF to define the NPM functions different operators are allowed to use. The following PROFILE statement example corresponds to the OPERATOR statement for PROFILEP shown in Step 1:

```
PROFILE NAME=PROFILEP,
  CONTROL=YES, CONTROL NPM SESSIONS
  MAXUSER=2, MAXIMUM NUMBER OF OPERATORS WHO CAN
  USE THIS PROFILE AT THE SAME TIME
  NETCOLL=YES, NETWORK COLLECTION
  NETANLY=YES, NETWORK ANALYSIS
  NSA=NO, NETWORK SESSION ACCOUNTING AND
  NETWORK GATEWAY ACCOUNTING
  PD=YES, PROBLEM DETERMINATION
  SMF=NO, SYSTEM MANAGEMENT FACILITIES
```

In this example, only two users at a time can access PROFILEP (MAXUSER=2), which allows both network collection and analysis (NETCOLL=YES, NETANLY=YES), the ability to bind and unbind resources (CONTROL=YES), and access to NPM’s problem determination facilities (PD=YES). However, the operators in this example cannot access NSA or NGA network accounting or System Management Facility control functions (NSA=NO, SMF=NO).

**Step 3.** Set up how NPM stores user profile information for each operator by allocating the FNMPROFS partitioned data set to store the user profiles. You can find an example of how to allocate the FNMPROFS data set in the sample FNMLOGDS JCL.

See *NPM Concepts and Planning* for additional information.

**Step 6. Define the NPM Start Procedure (FNMSNPM)**

The JCL needed to start NPM under MVS is supplied in the NPM.V2R4M0.SFNMJCL1(FNMSNPM) data set. This JCL is shown in the following example.
Step 6. Define the NPM Start Procedure (FNMSNPM)

```
//FNMSNPM PROC HLQ=NPM,REL=V2R4M,OUT='*',
   INIT=FNMINIT,OPER=FNMOPER,PROF=FNMPROF,
   CMD=FNMSRT

//******************************************************************************
/****
/****  PROCEDURE: FNMSNPM
/****
/****
/****  FUNCTION: STARTED TASK PROCEDURE TO INVoke NPM
/****
/****
/****  SYMBOLIC PARMs:
/****
/****  HLQ : DATASET PREFIX FOR NPM DATA SETs
/****
/****  REL : NPM RELEASE DATA SET QUALIFIER
/****
/****  OUT : SYSOUT CLASS
/****
/****  B : NUMBER OF CSA DATA BUFFER
/****
/****  S : NUMBER OF CSA SESSION BUFFERS
/****
/****  INIT : NAME OF INITIALIZATION STATEMENT MEMBER
/****
/****  OPER : NAME OF OPERATOR MEMBER
/****
/****  PROF : NAME OF PROFILE MEMBER
/****
/****  CMDS : NAME OF INITIALIZATION COMMANDS MEMBER
/****
/****
/****  NPM EXEC PGM=FNMMAIN,REGION=4,DPRTY=1,TIME=144,
   PARM=('INIT=&INIT,OPER=&OPER,PROF=&PROF',
   'CMDS=&CMDS')
//STEPLIB DD DSN=&HLQ..&REL..SFNMLMD1,DISP=SHR
//FNMLLIB DD DSN=&HLQ..&REL..SFNMLMD1,DISP=SHR
//SYSPRINT DD SYSOUT=OUT
//FNMPARM DD DSN=&HLQ..&REL..SFNMLMD1,DISP=SHR
//FNMLLOG DD SYSOUT=OUT,FREE=CLOSE
//FNMSNPLD DD DSN=&HLQ..&REL..SFNMLMD1,DISP=SHR
//FNMPROFS DD DSN=&HLQ..&REL..SFNMLMD1,DISP=SHR
//FNMPREV1 DD DSN=&HLQ..&REL..REV01,DISP=SHR
//FNMPREV2 DD DSN=&HLQ..&REL..REV02,DISP=SHR
//FNMSSES1 DD DSN=&HLQ..&REL..SESS01,DISP=SHR
//FNMSSES2 DD DSN=&HLQ..&REL..SESS02,DISP=SHR
//FNMLWG1 DD DSN=&HLQ..&REL..LWG01,DISP=SHR
//FNMLWG2 DD DSN=&HLQ..&REL..LWG02,DISP=SHR
//FNMLLOG1 DD DSN=&HLQ..&REL..LOG01,DISP=SHR
//FNMLLOG2 DD DSN=&HLQ..&REL..LOG02,DISP=SHR
//FNMSCMD5 DD DSN=&HLQ..&REL..FNMSCMD5,DISP=SHR
//FNMVLG1 DD DSN=&HLQ..&REL..VTAMLGT1,DISP=OLD
//FNMVLG2 DD DSN=&HLQ..&REL..VTAMLGT2,DISP=OLD
//VTAMLST DD DSN=SY1.VTAMLST,DISP=SHR
//SYSMDUMP DD DSN=HLQ..&REL..SYSMDUMP,DISP=OLD
//FNMSNAP DD SYSOUT=OUT
//FNMSDUMP DD DSN=HLQ..&REL..SYSMDUMP,DISP=OLD
```

Note: The JCL statements shown in this example are described on page 177.

Make the following updates to the NPM start procedure (FNMSNPM):

**Step 1.** Specify the following data sets containing the NCP resource resolution table (RRT) and the dummy RRTs generated for 3746 APPN network nodes (NNs) in FNMSNPM's FNMLLIB DD statement to allow NPM access to the NCP RRT:

```
//FNMLLIB DD DSN=ncp.rrt.linklib,DISP=SHR
// DSN=another.rrt.linklib,DISP=SHR
```

**Step 2.** Add the following DD statement for each VTAMLST data set containing either 1) NCP major node definitions, or 2) dummy major nodes generated for 3746 APPN network nodes, to the VTAMLST DD statement of the NPM startup procedure:

```
//VTAMLST DD DSN=SYS1.VTAMLST,DISP=SHR
// DSN=user.ncp.vtamlst,DISP=SHR
```
Step 7. Install NPM-to-NPM Communication

The VTAMLST statement identifies the data set containing the NCP major nodes and dummy major nodes for 3746 APPN NNs (generated using the FNM3746 EXEC). NPM must have access to these members and to the RRTs in order to define an NCP or 3746 APPN NN using the NCP command.

Step 3. Concatenate the NetView CNMLINK library to the STEPLIB DD statement in the NPM startup procedure JCL, as shown in the following example:

```
//STEPLIB DD DSN=&HLQ..V2R4M0.SFNMLMD1,DISP=SHR
// DD DSN=SYS1.CNMLINK,DISP=SHR
```

All of the libraries named in the STEPLIB DD statement must be authorized. If you change the default library names, and the NetView program CNMNETV module is not in the library SYS1.CNMLINK, substitute your library name for the library that contains CNMNETV.

Step 7. Install NPM-to-NPM Communication

NPM-to-NPM communication enables operators to perform NPM functions on an NPM that is remote from the NPM they are using. For example, they can log on to an NPM at one location, communicate with an NPM at another location, and use the data collection function as if they were logged on at the second location. They can start and stop collection or view collected data, but they only see monitor or event messages at the monitored NPM.

Before operators can use NPM-to-NPM communication, you must define their operator IDs at the other locations. Operators can have different authorizations at each location.

NPM operators can operate other NPMs using NPM-to-NPM communication. When an NPM operator makes requests from another NPM, the security level check is defined in the remote NPM. The first check requires that the operator ID be defined to the remote NPM. If it is not, the request is rejected. If the security is minimal in the remote NPM, no further checking is done. Otherwise, the following checks are performed:

- If you specified a remote profile for the operator (the RPROFILE keyword), it is used to check the request. Otherwise, the operator must provide a profile name when sending the request.

  **Note:** The RPROFILE profile name is taken from the operator statement defined in the remote NPM.

- You must define the profile name in the remote NPM and specify it in the OPERATOR statement for the operator ID in the remote NPM.

- The profile must allow the operator to process the request.

If any of these checks fail, an appropriate error message is sent to the operator.

For an operator to use NPM-to-NPM communication, the APPLID, PASSWD, and HOST parameters on the VTAM and ANPM initialization statements should reference each other, as shown in the following example.
Step 7. Install NPM-to-NPM Communication

<table>
<thead>
<tr>
<th>VTAM1</th>
<th>NPM1</th>
<th>APPL PRCT=X1Y2Z3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPM1</td>
<td>VTAM</td>
<td>APPLID=NPM1, PASSWD=X1Y2Z3</td>
</tr>
<tr>
<td>ANPM</td>
<td>APPLID=NPM2, HOST=VTAM2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VTAM2</th>
<th>NPM2</th>
<th>APPL PRCT=A1B2C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPM2</td>
<td>VTAM</td>
<td>APPLID=NPM2, PASSWD=A1B2C3</td>
</tr>
<tr>
<td>ANPM</td>
<td>APPLID=NPM1, HOST=VTAM1</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** NPM Version 2 systems can only perform NPM-to-NPM communication with another NPM Version 2 system.

Use the following procedure to install NPM-to-NPM communication:

**Step 1.** Code the following statements to define each remote NPM to your local NPM:

- Code an `ANPM` statement in `FNMINIT` to define each remote NPM. Specify the name of the remote NPM with the `APPLID` parameter. Specify the name of the host in which the remote NPM is running, with the `HOST` parameter.

- Code a `CDRM` statement in `FNMINIT` for each remote NPM. Specify the name of the host in which the remote NPM is running on the `NAME` parameter. This is the same name you specified on the `HOST` parameter of the `ANPM` statement in this step. Specify the subarea number of the host in which the remote NPM is running on the `SA` parameter.

- Code an `OPERATOR` statement in `FNMOPER` to define the operator ID of each operator on the remote NPM who can communicate with the local NPM. Specify the operator ID on the `NAME` parameter, and the name of the security profile for this operator on the `RPROFILE` parameter.

- Code a `PROFILE` statement in `FNMPROF` to define a security profile for remote operators communicating with the local NPM. The name of this profile is specified on the `RPROFILE` parameter of the `OPERATOR` statement in this step.

**Step 2.** Code the following statements to define your local NPM to each remote NPM:

- Code an `ANPM` statement in `FNMINIT` at each remote NPM to define your local NPM. Specify the name of the local NPM with the `APPLID` parameter. Specify the name of the local host with the `HOST` parameter.

- Code a `CDRM` statement in `FNMINIT` at each remote NPM to define your local NPM. Specify the name of your local host on the `NAME` parameter. This is the same name you specified on the `HOST` parameter of the `ANPM` statement in this step. Specify the subarea number of the local host on the `SA` parameter.
Step 7. Install NPM-to-NPM Communication

- Code an OPERATOR statement in FNOPER at each remote NPM to define the operator ID of each operator on your local NPM who can communicate with the remote NPM. Specify the operator ID on the NAME parameter, and the name of the security profile for the local operator on the RPROFILE parameter.

- Code a PROFILE statement in FNMPROF at each remote NPM to define a security profile for operators on your local NPM communicating with the remote NPM. The name of this profile is specified on the RPROFILE parameter of the OPERATOR statement in this step.

The following example shows the definitions required to install NPM-to-NPM communication between a local and remote host. In this example, the local NPM is running in HOST1 and the remote NPM is running in HOST2. The same security profile, OPER1R, is defined in both hosts.

**Local NPM FNMPARM initialization statements:**

**FNMINIT member:**
- ANPM APPLID=NPM2,HOST=HOST2 ...
- CDRM NAME=HOST2,SA=2 ...
- VTAM APPLID=NPM1,PASSWD=X3Y4Z5 ...

**FNOPER member:**
- OPERATOR NAME=OPER1,RPROFILE=OPER1R ...

**FNMPROF member:**
- PROFILE NAME=OPER1R ...

**Remote NPM FNMPARM initialization statements:**

**FNMINIT member:**
- ANPM APPLID=NPM1,HOST=HOST1 ...
- CDRM NAME=HOST1,SA=1 ...
- VTAM APPLID=NPM2,PASSWD=A1B2C3 ...

**FNOPER member:**
- OPERATOR NAME=OPER1,RPROFILE=OPER1R ...

**FNMPROF member:**
- PROFILE NAME=OPER1R ...

You can send console commands to a remote NPM, but the commands are subject to NPM security at the remote NPM. To set up NPM-to-NPM communication for console commands, code the CONSOLE statement at the local NPM, as shown in the following example:

**CONSOLE OPERATOR=NPMS,PROFILE=NPMCONS**

At the remote NPM, code the OPERATOR and PROFILE statements to authorize the local NPM to issue the commands, as shown in the following example:

**OPERATOR NAME=NPMS,PROFILE=NPMCONS**
**PROFILE NAME=NPMCONS,CONSOLE=YES,DFM=YES,PD=NO**

The PROFILE statement used by the system console does not have to be the same at the local and remote NPMs. The remote NPM uses its PROFILE statement to determine whether the sending NPM is authorized to issue the command.

**Note:** When you use the MVS console to issue a command from one host NPM to another host NPM using NPM-to-NPM communication, NPM uses the procedure name you assigned at NPM startup as the operator ID for security verification.
Step 10. Code Automated Commands in FNMSTRT

Therefore, you must ensure that you assign the same procedure name in FNMOPER for both host NPMs.

Step 8. Install the NPM Desk/2 User Interface

This step is optional. If you are using NPM Desk/2, see the installation procedure provided in Chapter 7, “Installing the NPM Desk/2 User Interface” on page 89.

Additional NPM Desk/2 installation information is provided in NPM Desk/2 User’s Guide.

Step 9. Install NPM NetWare Resource Collection

This step is optional. If you are installing NetWare Resource Collection, see the installation procedure provided in Chapter 8, “Installing NPM Netware Resource Collection” on page 117.

Step 10. Code Automated Commands in FNMSTRT

Use the following procedure to code the automated commands in FNMSTRT. See “Issuing NPM Commands at Startup (FNMSTRT)” on page 141 for additional information about FNMSTRT.

Note: All of the sub-steps listed in this step are optional.

Step 1. Code a separate NCP command in an EXEC member in FNMSCMDS for each NCP or 3746 APPN NN in your network for which you want to collect data. Use the following parameters on the NCP command:

\[\text{name} = \text{ncp\_name}, \text{npalu} = \text{lu\_name}, \text{istatus} = \text{ACTIVE}\]

where:

- \text{ncp\_name} Specifies the NCP from which network data is collected.
- \text{lu\_name} Specifies the primary network performance analysis logical unit’s (NPALU’s) alias name in this network. The default primary NPALU is the primary NPALU defined in the NCP generation definition. If you specify the name of an NPALU that has been defined as a backup NPALU in the NCP generation definition, the name you specify is ignored and the default name is used. If NPA=NO is specified in the NCP’s BUILD definition statement, the NPALU parameter is ignored. NPALU names must be unique in the network.
- \text{ACTIVE} Specifies that the primary NPALU should be bound when NPM starts up. If you do not code this parameter, specify \text{CONTROL=YES} in a PROFILE statement so that an operator can manually bind the NPALU.

For information about another way to bind NPALUs, see step 6 on page 67. To define the NCP at NPM startup, you must code an EXECUTE command in FNMSTRT to call this EXEC member.

The SNI=YES parameter is necessary on NCP commands that define 3746 APPN NNs (if more than one is to be defined) to avoid problems caused by duplicate resource names. Use the following parameters on the NCP command:

\[\text{name} = \text{node\_name}, \text{sni} = \text{YES}\]
Step 10. Code Automated Commands in FNMSTRT

where:

node_name Specifies the NCP or 3746 APPN NN from which network
data is collected.

See “Processing NPM EXEC Members (FNMSCMDS)” on page 160 for
more information.

Step 2. Code the NETCOLL command to start network data collection or
monitoring.  See the NPM User's Guide for more information about the
NETCOLL command.

Step 3. Code a SESSCOLL command to automatically start and stop session
data collection.  See the NPM User's Guide for more information.

Step 4. Code the following commands in FNMSTRT to control the collection of
NSA or NGA data:

- **For NSA**: Code the NSACNTL, NSAMDFY, and NSASOLCT
  commands in FNMSTRT to control collection of NSA data.
- **For NGA**: Code the NGACNTL, NGAMDFY, and NGASOLCT
  commands in FNMSTRT to control collection of NGA data.

Step 5. Code the following parameters on the LANMGR command in
FNMSCMDS or FNMSTRT for each LAN Manager in your network:

ADDRESS=address,
SP=name,
OPTION=ADD,

where:

address Specifies the adapter address used by the LAN
Manager.

name Specifies the PU name or the LU name of the LAN
Manager service point as defined to VTAM.

ADD Adds this LAN Manager to NPM.

See the NPM User’s Guide for more information.

Step 6. Code a LBRGCOLL and a LSEGCOLL command in FNMSTRT for LAN
data.  An NPM operator can also start LAN data collection using NPM’s
online panels, NPM Desk/2, an NPM EXEC, or NPM’s console support.

Step 7. Code the following parameters on the LWGRES command in
FNMSCMDS or FNMSTRT for each NetWare resource for which you
want to collect data.

The LWGRES command defines a NetWare resource to NPM for
collection purposes.  With the LWGRES command, you can add the one
NetWare resource you specify.  At least one of these resources must be
an NPM NetWare Agent.

- Code the SP keyword to specify the service point (SP) physical unit
  (PU) address.  The address is from 1 to 8 characters.  The SP
  keyword uses the following syntax:

  \[ SP=pu\_name \]

- Code the NETWORK and NODEADDR keywords to specify the IPX**
  network address and the physical node address of a NetWare
Step 10. Code Automated Commands in FNMSSTRT

resource that identifies the node in a connectivity ring. The network address is a hexadecimal value from X'1' to X'FFFFFFFE'. The node address is a hexadecimal value from X'1' to X'FFFFFFFFFE'.

These keywords use the following syntax:

- ,NETWORK=network_address─ ,NODEADDR=network_address─

• Code the SERVER keyword to specify the name of a server. The server name can be from two to forty-seven characters. Valid characters are A–Z, 0–9, the dash (–), and the underscore (_).

When you specify this keyword, you must also specify the RESOURCE keyword as NWSRV.

The SERVER keyword uses the following syntax:

- ,SERVER=server_name

• Code the AGENT keyword to specify the name of a server containing the NPM NetWare Agent. The name of the NPM NetWare Agent can be from two to forty-seven characters. Valid characters are A–Z, 0–9, the dash (–), and the underscore (_).

The AGENT keyword uses the following syntax:

- ,AGENT=agent_server_name

• Code the RESOURCE keyword to specify the type of resource you are defining for collection.

The RESOURCE keyword uses the following syntax:

- RESOURCE=NWAGE─
- RESOURCE=NWRTR─
- RESOURCE=NWSRV─

where:

NWAGE Defines the NetWare agent server for collection.
NWRTR Defines the NetWare external router for collection.
NWSRV Defines the NetWare server for collection.

• Code the HOST keyword to specify the host that receives this command. You code this keyword only if you issue the command from the system console.

The HOST keyword uses the following syntax:

- HOST=LOCAL─
- HOST=host_name─

• Code the L keyword to specify the MVS console that receives response messages. You code this keyword only if you issue the command from the system console.
The L keyword uses the following syntax:

\[
\text{\textbackslash{SM}590000} \quad \text{\textbackslash{SM}590000}\text{\textbackslash{SM}630000}
\]

- The OPTION parameter specifies the action to perform.

\[
\text{\textbackslash{SM}590000} \quad \text{\textbackslash{SM}590000}\text{\textbackslash{SM}630000}
\]

where:

- **ADD** Adds a NetWare resource. If you try to add a NetWare resource that is already defined, the command is ignored and a message is issued.

- **DELETE** Deletes a NetWare resource. If you try to delete a NetWare resource that is not defined, the command is ignored and a message is issued.

**Note:** You use only ADD at the time of installation.

The following examples show various ways you can code the LWGRES command keywords.

**Example:** Use the following command to add a server resource for collection.

```
LWGRES SP=NWSRV1,
    NETWORK=100B,NODEADDR=100004,SERVER=SERV_name,
    AGENT=AG_SERV_01,
    RESOURCE=NWSRV
```

**Example:** Use the following command to add an agent server resource for collection.

```
LWGRES SP=NWSRV1,
    NETWORK=100B,NODEADDR=100005,
    AGENT=AG_SERV_01,
    RESOURCE=NWAGE
```

**Example:** Use the following command to add a router resource for collection.

```
LWGRES SP=NWSRV1,
    NETWORK=100B,NODEADDR=100006,
    AGENT=AG_SERV_01,
    RESOURCE=NWRTR
```

See “LWGRES” on page 153 and the *NPM User’s Guide* for more information on how to code this command.

**Step 8.** Code an LWGCOLL command in FNMSTRT for NetWare resources data collection. See the “LWGCOLL” on page 154 for an example of how to code this command. An NPM operator can also start NetWare resources data collection using NPM’s online panels, NPM Desk/2, an NPM EXEC, or NPM’s console support. See the *NPM User’s Guide* for more information.
Step 11. Install NPM Exits

This step is optional. If you need to install NPM exits, see the installation procedure provided in Chapter 9, “Installing NPM Exits” on page 121.

Step 12. Make Changes to Other Related Products

The procedures provided in this step describe how to make changes to some of the related products and components when you install NPM. Installation information is provided for the following:

- NCP
- 3746 APPN network node
- NetView
- NVAS
- TPX
- CICS

Changes to the NCP

Use the following procedure to make changes to the NCP when you install NPM:

**Step 1.** You must update the NCP generation definitions to include NPM specific definitions. If network data is to be collected, you must also include a primary NPALU and optional backup NPALUs.

The NPALU is a special logical unit, defined in the NCP, that communicates with NPM. Network performance and accounting data is collected by the NCP and sent to NPM by the NPALU. See NCP Resource Definition Guide for a complete description of the NPM data collection functions in the NCP and a description of collection techniques.

NCP statements follow strict rules. Starting columns and continuations from line to line are not the same as for NPM statements. Read the syntax rules for NCP statements in the NCP, SSP, and EP Resource Definition Reference before coding the NCP statements in this section.

Use the following procedure to update the NCP generation definition:

- **a.** If you are using NCP V4R3 or a later release, or if NPM.xxxxx keywords are coded, perform the following modifications to the JCL. See step 2 on page 62 for more information.

  Modify the JCL used for the NCP/EP definition facility (NDF) by concatenating the following DD statement after the STEPLIB DD statement:

  ```plaintext
  // DD DSN=NPM.V2R4M0.SFNMLMD1,DISP=SHR
  ```

  The NPM.V2R4M0.SFNMLMD1 data set must contain the FNMNDFGN load module. If you rename the NPM load module library, change the name in the above DD statement to your new library name.

- **b.** Code the OPTIONS statement if you are using NCP V4R3 or a later release. Omit this step for NCP V4R2 and earlier releases if no NPM.xxxxx keywords are coded. See step 2 on page 62 for more information.

  Add the following statement as the first line of the NCP generation definition that is not a comment:
Step 12. Make Changes to Other Related Products

OPTIONS USERGEN=(FNMDFGN)

Note: For NPM V1R5 and later releases, FNMDFGN does not produce an NPM RRT, it only prevents NDF from flagging NPM.xxx parameters as errors. Therefore, if those keywords are not used, FNMDFGN is not necessary because the dynamic reconfiguration support has removed the need for an NPM RRT.

c. Code the BUILD statement in the NCP generation definition. There are six parameters on the BUILD statement that apply to NPM. The following descriptions explain how to code these parameters:

NPA

Code NPA=YES to enable NPM’s network data collection functions. Code NPA=(YES,DR) to enable collection on resources from the dynamic reconfiguration pool. This includes NTRI resources and ODLC LAN resources as well as dynamically reconfigured resources.

Code NPA=(YES,DRTP) to enable transmission priority support for SDLC lines and PUs, including PUs that are dynamically added.

See NCP, SSP, and EP Resource Definition Reference for more information about the NPA keyword. See also step 4 on page 67. If you use only NPM session data collection, code NPA=NO. NPM is not able to collect any network data for this NCP.

SESSACC

Code the SESSACC parameter to enable network session accounting and network gateway accounting. See step 3 on page 63.

GWSESAC

Code the GWSESAC parameter to enable network gateway accounting. See step 3 on page 63.

PUNAME

Specify the NCP PU name if you generate more than one NCP load module for a single CCU. By specifying the same PU name for each NCP generation, NPM commands can use the common PU name, and NPM records will write the common PU name.

However, the name that you specify on the NPM NCP command that defines an NCP to NPM must be the same as the name that is specified on the NEWNAME parameter of the NCP BUILD statement. This is because NPM uses this name to find the NCP major node in VTAMLST. You do not have to keep track of different NCP names for the same CCU. This parameter is optional.

MAXTP

Specifies the maximum number of resources on which NPM can collect performance data by transmission priority at one time. See NCP, SSP, and EP Resource Definition Reference for more information about coding the keyword, limits on its size, and default value. If you code a value that is too small, start collection requests are rejected by the NCP.
d. Follow the substeps listed below to code the GROUP, LINE, PU, and LU statements:

1) Specify NPACOLL=YES for each resource on which you want to collect NPM network data.
   - For generic NEO, X.25(NPSI, XI), frame-relay, ODLC LAN, and Ethernet, specify NPACOLL=(YES,EXT) for each resource on which you want to collect NPM network data.
   - The default for NPACOLL is passed down, where applicable, from the GROUP to the LINE to the PU (CLUSTER), and from there to the LU (TERMINAL) macros.

2) Specify the SPEED parameter for line resources. The SPEED parameter on the LINE statement is used to set the line capacity when the line utilization percentage is calculated. Specify two line speeds for duplex lines, and specify one line speed for half-duplex lines. Note that in some cases the NCP SPEED parameter cannot take two values; in this case, use the NPM SPEED parameter. NPM takes the speed value in the following order: from the NPM.SPEED, TRSPEED (for token-ring connections), and SPEED parameters.

   The way in which the ADDRESS parameter is coded for an SLDC line effects NPM's LINEUTL monitor. NPM uses the higher of the primary and secondary line utilizations for comparison with the user-defined upper threshold. The following explains the way in which NPM interprets each line definition.

   - If the ADDRESS parameter is coded as:
     ADDRESS=(xxx,FULL)
     for a line that is full-duplex (one transmit circuit and one receive circuit), code the NPM.SPEED parameter as:
     NPM.SPEED=(primary_line_speed,secondary_line_speed)
     The primary and secondary line speeds are equal to the declared speeds for the line. That is, a full-duplex of 19200 bps can transmit at 19200 bps and receive at 19200 bps simultaneously.

     NPM calculates both the primary and secondary line utilizations, and the larger of the two is compared to the upper threshold.

   - If the ADDRESS parameter is coded as:
     ADDRESS=(xxx,FULL)
     for a line that is not full-duplex (that is, only one physical circuit), you should code the NPM.SPEED parameter as:
     NPM.SPEED=(primary_line_speed,0)
     The zero in the secondary_line_speed field tells NPM to treat the line as half-duplex for the purpose of calculating line utilization.

     NPM understands that the line is half-duplex if the ADDRESS parameter is coded as:
Step 12. Make Changes to Other Related Products

ADDRESS=(xxx,HALF)

NPM calculates the total line utilization, and compares it to the upper threshold. In this case, code the NPM.SPEED parameter as:

NPM.SPEED=primary_line_speed

3) Code NPATP=YES for SDLC lines and PUs to enable transmission priority support.

e. For generic NEO applications, code the accounting entry point name on the UACCTNG parameter of the GENEND statement. If you do not code the UACCTNG parameter for X.25 (NPSI, XI), it is supplied automatically. NDF reads the first eight accounting entry point names from the UACCTNG parameter, which enables NPM to display and modify the accounting thresholds for a maximum of eight NEO applications.

Do not use NPSI or XI as an accounting entry point name on the UACCTNG parameter because these words are reserved by NPM.

f. Define the NPALU group, including one primary NPALU. For network session and gateway accounting, you can define up to seven additional backup NPALUs. Each NPALU name in every NCP that communicates with NPM must be unique.

The NPALUs are defined with an NPALU resource group definition. This group definition must follow the last bisynchronous group definition. The following example shows an NPALU resource group definition.

```
A03NPM GROUP LNCTL=SDLC,
    ** SDLC PROTOCOL **
    NPARS=NO,
    ** NPM LINK GROUP **
    VIRTUAL=YES, ** PROGRAMMED RESOURCE SUPPORT **

**
A03NPML LINE **
A03NPMPU PU **
A03NPMLU LU MAXCOLL=500,
    ** COLLECT ON 500 RESOURCES ** COLLECT ON 500 RESOURCES
    LOGAPPL=NPM01, ** AUTOMATIC LOGON TO NPM01 **
A03NPML2 LU ** NSA/NGA BACKUP NPALU **
A03NPML3 LU ** NSA/NGA BACKUP NPALU **
A03NPML4 LU ** NSA/NGA BACKUP NPALU **
A03NPML5 LU ** NSA/NGA BACKUP NPALU **
A03NPML6 LU ** NSA/NGA BACKUP NPALU **
A03NPML7 LU ** NSA/NGA BACKUP NPALU **
A03NPML8 LU ** NSA/NGA BACKUP NPALU **
```

In this example, A03NPMLU is the primary NPALU. A03NPML2 through A03NPML8 are the backup NPALUs.

The MAXCOLL keyword on the first NPALU statement specifies the maximum number of resources on which NPM can collect data at one time. If you specify MAXCOLL, its value must be greater than zero and less than or equal to the number of resources with NPACOLL=YES. Be careful not to choose a value greater than necessary because a high value can waste NCP storage space. The default value is the number of resources defined with NPACOLL=YES.
Step 2. After you have generated your NCP, you can override certain parameters by updating the VTAMLST member or file with NPM.xxxxx keywords. However, in this case, use FNMDNDFGN during the NCP generation.

a. For 3705 communication controllers with RPQs that have a cycle speed different from the standard speed, add the following NPM.CYCLE keyword to the BUILD statement to specify the correct speed so that 3705 utilization is calculated accurately:

\[ \text{NPM.CYCLE=nnnn} \]

where:

nnnn Specifies the cycle speed of the 3705 controller in nanoseconds.

b. You can change the SPEED value for a line by adding the following NPM.SPEED keyword to the GROUP or LINE statement. If the NPM.SPEED keyword is coded on the GROUP statement, the keyword applies to all the lines in that group:

\[ \text{NPM.SPEED=(speed1,speed2)} \]

where:

speedx Specifies that the speed values from this keyword replace any SPEED value specified for the line and apply to all resources associated with the line.

c. You can define NPM dynamic network collection and session collection options by coding either or both of the following NPM.DNC and NPM.SESSH keywords on the GROUP, LINE, PU, and LU statements. Values coded on the GROUP statement apply to the LINE statement. Values coded on the LINE statement apply to the PU (CLUSTER) statement, and from there to the LU (TERMINAL) statements.

\[ \text{NPM.DNC=} \]

where:

YES Specifies that you want NPM dynamic network collection on all LU resources. The default value is DNC=YES.

NO Specifies that you do not want NPM dynamic network collection on all LU resources.

\[ \text{NPM.SESSH=} \]

where:

X Specifies that you want to exclude NPM session collection for this resource.

You can use the X parameter on the NPM.SESSH keyword to prevent session data collection for specific resources, such as printers and NPALUs, when you collect session data by application. You can also dynamically update these parameters and others after the NCP has
Step 12. Make Changes to Other Related Products

been defined to NPM by using GENERIC, LINE, PU, and LU commands. For more information about these commands, see NPM User's Guide.

Step 3. You must code two statements in the NCP generation definition to enable network session accounting and network gateway accounting. The SESSACC statement is required for both network session and network gateway accounting. The GWSESAC statement is required for network gateway accounting only. In addition, you should usually define backup NPALUs to protect against lost accounting data.

Network session accounting is available for NCP V4R3 and later releases. Network gateway accounting is available for NCP V4R3.1 and later releases for 3725 communication controllers, and NCP V5R2.1 and later releases for 3720 and 3745 communication controllers. Network gateway accounting also requires VTAM V3R2 or later releases.

To collect network session accounting and network gateway accounting data, complete the following steps:

a. Code the SESSACC keyword on the BUILD statement in the NCP generation definition to enable network session accounting. You must also specify the SESSACC keyword to enable network gateway accounting.

See NCP, SSP, and EP Resource Definition Reference for additional information and default values for the SESSACC keyword.

The SESSACC keyword uses the following syntax:

```
SESSACC=(NO,DEFER,YES),DEFER(bytenum)
```

The following are the options for this keyword:

where:

NO Specifies that you do not want NPM to collect network session accounting for this NCP. If you specify NO, you cannot turn on accounting using NPM. The default is SESSACC=NO.

YES Specifies that you want NPM to collect network session accounting for this NCP. If you specify YES, you can turn accounting on and off using NPM.
DEFER Specifies that you want NPM to collect accounting data on both primary and secondary logical unit boundary session blocks, but defer collection until a modification command is received from NPM. The default value for this parameter is SESSACC=DEFER.

ALL Specifies that you want NPM to collect accounting data on both primary and secondary logical unit boundary session blocks.

PLU Specifies that you want NPM to collect accounting data on primary logical unit boundary session blocks.

SLU Specifies that you want NPM to collect accounting data on secondary logical unit boundary session blocks.

nscnum where:

nscnum Specifies the minimum number of NPM session counter control blocks (NSCs) to be generated by NCP. The number of NSCs actually created may be greater than the number you specified if additional storage is available after all NCP control blocks have been generated.

bytenum where:

bytenum Specifies the byte threshold that must be reached before the NCP sends NSA data to NPM. The byte threshold has a minimum value of 256 and must be greater than the PIU threshold value. The maximum value is 2147483646. The range that NPM allows for this parameter is smaller than the range that NCP allows.

piunum where:

piunum Specifies the PIU threshold that must be reached before the NCP sends NSA data to NPM. The PIU threshold has a minimum value of 64 and must be less than the byte threshold value. The maximum value is 32767. The range that NPM allows for this parameter is smaller than the range that NCP allows.

nscx where:

nscx Specifies the number of NSC extensions to be defined.
Step 12. Make Changes to Other Related Products

where:

\textit{piudist} Specifies up to six distribution byte-range values. If there are no PIU distributions defined for the SESSACC keyword, NPM cannot modify the PIU distribution. The \textit{piudist} parameter is available with NCP V4R3.1 and later releases.

\textbf{Note:} In the SESSACC keyword, parameters are positional. If you omit a parameter, you must still insert a comma as a place holder.

b. Code the GWSESAC keyword on the BUILD statement in the NCP generation definition to enable network gateway accounting.

See NCP, SSP, and EP Resource Definition Reference for additional information and default values for the GWSESAC keyword.

The GWSESAC keyword uses the following syntax:

\begin{verbatim}
GWSESAC = (NO, NODEFER, DEFER, ,nscnum, byteth)
\end{verbatim}

The following are the options for this keyword:

\begin{verbatim}
NO
YES
\end{verbatim}

where:

\textbf{NO} Specifies that you not want NPM to collect network gateway accounting data for this NCP. If you specify NO, you cannot turn accounting on using NPM. The default is GWSESAC=NO.

\textbf{YES} Specifies that you do want NPM to collect network gateway accounting data for his NCP. If you specify YES, you can turn accounting on and off using NPM.

\begin{verbatim}
,NODEFER, DEFER
\end{verbatim}

where:

\textbf{NODEFER} Specifies that this NCP should start collecting data as soon as the NCP comes up. The default value for this parameter is GWSESAC=NODEFER.

\textbf{DEFER} Specifies that this NCP should wait until a modification command is received from NPM before starting data collection.

\begin{verbatim}
,nscnum
\end{verbatim}

where:
Step 12. Make Changes to Other Related Products

**nscnum** Specifies the minimum number of NPM session counter control blocks (NSCs) to be generated by the NCP. The number of NSCs actually created can be greater than the number you specified, if additional storage is available after all NCP control blocks have been generated.

where:

**byteth** Specifies the byte threshold that must be reached before the NCP sends NGA data to NPM. The byte threshold has a minimum value of 256 and must be greater than the PIU threshold value. The maximum value is 2147483646. The range that NPM allows for this parameter is smaller than the range that NCP allows.

where:

**piuth** Specifies the PIU threshold that must be reached before the NCP sends NGA data to NPM. The PIU threshold has a minimum value of 64 and must be less than the byte threshold value. The maximum value is 32767. The range that NPM allows for this parameter is smaller than the range that NCP allows.

where:

**nscx** Specifies the number of NSC extensions to be defined.

where:

**piudist** Specifies up to six distribution byte-range values. If there are no PIU distributions defined for the SESSACC keyword, NPM cannot modify the PIU distribution. The **piudist** parameter is available with NCP V4R3.1 and later releases.

where:

**initialpcid** Specifies that the PCID received during session setup (SETCV) is not overwritten by the PCID received at the session establishment (BIND). You can also specify this suboperand as INIT. If it is not coded, the bind PCID continues to overwrite the SETCV PCID. The **initialpcid** parameter is available with NCP V4R3.1 and later releases.
Step 12. Make Changes to Other Related Products

**Note:** In the GWSESAC keyword, parameters are positional. If you omit a parameter, you must still insert a comma as a place holder.

**Step 4.** For NCPs prior to NCP V4R3, the NPALU collection in the NCP does not support dynamic reconfiguration of resources. When reconfiguration takes place for a resource, the relationship that exists between the name of the resource that NPM knows and the resource address that the NCP knows is lost. Resources generated with PUDR=YES or with LUDR=YES are still supported for network data collection, but network collection stops when the resources are reconfigured.

With NCP Version 4 Release 3, the NCP can send NPM the information it needs to update its configuration as PUs and LUs in the NCP are reconfigured. To enable the sending of the information to NPM, add the following parameter to the BUILD macro:

```
NPA=(YES,DR)
```

This function works independently of network session accounting.

Code `NPA=(YES,DRTP)` on the BUILD statement to enable transmission priority support for SDLC lines and PUs, including PUs that are dynamically added. See *NCP, SSP, and EP Resource Definition Reference* for more information about the NPA keyword.

NCP sends dynamic reconfiguration information for all PUs and LUs in the NCP, regardless of ownership.

**Step 5.** If you use an NCP that is connected to multiple hosts (for example, an MVS and a VM host), use the GENLEVEL parameter to enable NPM to run in both hosts. VTAM and NPM use the NCP date and time stamp to verify that the NCP loaded in a CCU matches the NCP defined to VTAM or NPM. When an NCP is generated on multiple hosts, the NCP date and time stamp match only for the last host that generated the NCP.

The GENLEVEL parameter enables you to replace the NCP date and time stamp with a unique character string that identifies a particular NCP generation. The NPM in each host can then bind to the NCP. Each time an NCP is changed, the value of the GENLEVEL parameter must also be changed to maintain system integrity.

**Step 6.** When the path between the NPALU and NPM is activated, VTAM drives NPM’s LOGON exit routine, which creates a session with the NPALU and, using the network command restart support, start collecting network data again without operator intervention. Message FNM065I is issued each time a session with the NPALU and the NCP is started.

To use the autologon support, add the following parameter to the definition of the NPALUs in the NCP generation definition:

```
LOGAPPL=applname
```

where:

**applname** Specifies the name of the NPM application.

An alternative method is to use the VTAM VARY command:

```
V NET,ID=npalu,LOGON=applname
```

If you use either method of activating NPALUs, you must specify ISTATUS=INACTIVE in the NCP command. For example:
Step 12. Make Changes to Other Related Products

NCP  NAME=ncpname,ISTATUS=INACTIVE

Specifying ISTATUS prevents NPM from attempting to establish a connection with the NPALU while VTAM is also attempting to establish the connection.

If an NPALU is defined with the following keyword, the NPM must have a matching NCP defined that contains the NPALU:

LOGAPPL=applname

Otherwise, the NPALU session is treated as an operator session; NCP rejects the NPM logo screen, NPM generates message FNM081E, and the session ends.

The discussion of autologon in this section applies to primary NPALUs. The ISTATUS parameter does not apply to backup NPALUs. NPM always tries to bind any backup NPALUs defined to it.

Changes to 3746 APPN Network Nodes

To collect data for resources connected to a 3746 APPN network node (NN), a dummy RRT and VTAMLST must be generated. An ISPF-based program that is supplied to do this makes the NN look and act like an NCP to NPM. Data for its resources is forwarded to NPM only if the NN is a 3746 Model 900 or a 3746 Model 950 at microcode level 830 V02.

Use the following procedure to make changes to 3746 APPN Network Nodes when you install NPM:

Data can be collected for resources attached to a 3746 APPN NN using the same interface that is used to collect data for NCP-controlled resources. For this to be possible, a dummy RRT and VTAMLST member must be generated for the NN. An ISPF-based program is supplied for this purpose.

Before this program is run for the first time, it is necessary to convert a supplied NPM sample RRT into load module format. This needs to be done only once.

Use the following procedure to generate 3746 APPN network node members:

**Step 1.** Go to ISPF Option 6, and issue the following command:

RECEIVE INDSN(NPM.V2R4M.0.SFNMINS0(FNM46RRT))

**Step 2.** When you are prompted for a target data set name, type the name of the NPM load module library:

DA(NPM.V2R4M.0.SFNMLMD1)

You may have to change the data set names to match your naming convention.

**Step 3.** Check that a new member, FNM46RRT, has been added to the NPM load module library.

The major node and RRT are created using a simple ISPF panel-driven dialogue, supplied with the enhancement code.

To invoke this dialogue, issue the following commands from ISPF Option 6:
Step 12. Make Changes to Other Related Products

ALLOC FI(SYSPROC) DA('NPM.V2R4M0.SFNMIN0') SHR REUSE

FN3746 NPM.V2R4M0.SFNMIN0

You may need to change the data set names to match your naming convention.

The ISPF panels prompt you to supply the following information:

- NPM load module library name. The default is NPM.V2R4M0.SFNMMLMD1
- VTAMLST data set name. The dummy VTAMLST member is placed in this data set, which must be defined in the VTAMLST DD statement of the NPM start procedure.
- RRT data set name. The dummy RRT member is placed in this data set, which must be defined in the FNMLLIB DD statement of the NPM start procedure, and which should be defined as Unformatted, lrecl 0, block size 6144.
- Node name. The name by which your 3746 APPN NN is known to NPM.
- Network ID. The name of the network in which your APPN NN resides.
- NPALU name. The name of the NPALU virtual LU to which NPM is to establish a session for the transport of performance data.

With this information, an appropriate dummy VTAMLST member is created and placed in the appropriate data set. A JCL job is also be created, which, when run, creates the dummy RRT.

Note that the VTAMLST member and RRT serve no purpose other than to enable the definition of your 3746 NN to NPM. Do not try to activate them to VTAM.

Step 4. Define the 3746 NPALU using the instructions provided with the 3746 product.

Changes to NetView

Use the following procedure to make changes to NetView when you install NPM:

Step 1. Ensure that NPM’s DST in the NetView program is set up to communicate with NPM, the NPM NetWare Agent, and the LAN Managers by performing the following steps. (This may already be done, if you have installed NetWare resources data collection.)

a. Define NPM’s NetView program DST application to VTAM by including the following statement in an application (APPL) major node:
   
   FNMSYRN APPL AUTH=CNM

   This definition allows the NPM DST to use the communication network management interface (CNMI) to solicit data from LAN Managers.

b. Allow the NetView program to have access to the LAN and NetWare resources data collection command processors (FNMRCVQ, FNMRCVR, FNMRCVT, FNMDLGFX, FNMNWUDR, and FNMDCNMI), the NPM message table (FNMMMSG10), and the DST
Step 12. Make Changes to Other Related Products

initialization exit (FNMDINTX) by concatenating NPM’s load library to
the STEPLIB DD statement in the NetView program startup procedure
JCL. The following example JCL shows how to provide this access:

```sql
//STEPLIB DD DSN=&SQ1..VTAMLIB,DISP=SHR
// DD DSN=NPM.V2R4M0..SFNMLMD1,DISP=SHR
```

c. Add the following CMDMDL statements to the NetView DSICMD
member:

<table>
<thead>
<tr>
<th>MOD</th>
<th>TYPE</th>
<th>RES</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNMRCVQ</td>
<td>R</td>
<td>Y</td>
</tr>
<tr>
<td>FNMRCVT</td>
<td>R</td>
<td>Y</td>
</tr>
<tr>
<td>FNMRCVR</td>
<td>R</td>
<td>Y</td>
</tr>
<tr>
<td>FNMDCNMI</td>
<td>D</td>
<td>Y</td>
</tr>
<tr>
<td>FNMNWUDR</td>
<td>D</td>
<td>Y</td>
</tr>
<tr>
<td>FNM62RDV</td>
<td>D</td>
<td>Y</td>
</tr>
</tbody>
</table>

d. Add the following TASK statement to the NetView DSIDMN member:

```
TASK MOD=DSIZDST,TSKID=FNMSYRN,MEM=FNMDSTD,PRI=3,INIT=Y
```

e. Add a member called FNMDSTD to the DSIPARM file that contains
the following DST initialization statements:

```
DSTINIT FUNCT=CNMI,XITDI=FNMDINTX
DSTINIT DSRBU=1,DSRBO=10,UNSOL=FNMNWUDR
```

Notes:

1) DSRBO limits the number of solicited and concurrent requests
that the task FNMSYRN can process. With LAN data collection,
the value of DSRBO determines the maximum number of LAN
Managers that the NetView program can queue concurrently. Set
DSRBO equal to the number of LAN Managers defined to NPM.
If you have already installed NetWare data collection, add the
number of LAN managers defined to NPM to the current value of
DSRBO. See the NetView Administration Reference for more
information about the DSTINIT statement and associated
operands.

2) The DSRBU specifies the number of unsolicited DSRBs that are
allocated to FNMSYRN. See NetView Administration
Reference for more information about the DSTINIT statement and
associated operands.

f. Update the NetView program DSIOPF member to include
the following autotask definition. (You may have already done this if you
have previously installed LAN data collection.)

```
FNMAUTO1 OPERATOR PASSWORD=FNMAUTO1
PROFLEN NPM1
```

g. Create member NPM1 in the DSIPRF data set to include the following
autotask profile:

```
NPM1 PROFILE IC=FNMRCVQ
AUTH CTRL=GLOBAL,MSGRECVR=NO
```

h. The NetView program-to-program interface must be active for NPM to
collect LAN data. If the program-to-program interface is not active,
NPM issues one of the following two messages:
Step 12. Make Changes to Other Related Products

FNM136E: SP=sp name BRG=bridge name LAN MANAGER
COMMUNICATION FAILED, TIMER EXPIRED

FNM136E: SP=sp name SEG=segment name LAN MANAGER
COMMUNICATION FAILED, TIMER EXPIRED

See NetView Application Programming Guide for more information about enabling the program-to-program interface.

i. Once the NetView program has been initialized, start the LAN data collection autotask by issuing the following command:

AUTOTASK OPID=FNMAUTO1

You can add this statement to a NetView program initial command list (for example, CNME1034) to automatically enable the LAN data collection autotask when you initialize the NetView program.

j. Ensure that the NPM DST has been started in the NetView program. Either code INIT=Y on the TASK statement used to define the DST, or issue the following command from the command facility in the NetView program:

START TASK=FNMSYRN

k. Ensure that the NetView DSI6DST task has been started in the NetView program. The task is required for the LU 6.2 transport function of NetView. If DSI6DST is not active, NPM issues the following message:

FNM133E: (DSI295I) OPTIONAL DATA SERVICE TASK DSI6DST IS NOT ACTIVE

l. Optionally code a LBRGCOLL and a LSEGCOLL command in FNMSTRT for LAN data. An NPM operator can also start LAN data collection using NPM’s online panels, NPM Desk/2, an NPM EXEC, or NPM’s console support.

Step 2. To use response time monitor (RTM) data collection with NPM, set up RTM data collection on the NetView program and install the SMF exit. See NPM Concepts and Planning for an explanation of RTM data collection.

Complete the following steps to install RTM data collection:

a. Set up NetView program command lists and performance class parameters to collect RTM data from resources that support RTM, such as 3174 and 3274 controllers.

b. Ensure that the SMFPRMxx member in SYS1.PARMLIB allows SMF to write SMF type 39 records from the STC class. NPM intercepts subtypes 1, 2, and 5. Code IEFU83 on the EXIT keyword in the SMFPRMxx member of SYS1.PARMLIB.

c. Use a NetView program command list or EXEC to issue the RTM collect commands to synchronize the writing of RTM data.

Note: See NetView Installation and Customization Guide (MVS) for information about collecting RTM data using the NetView program.
Step 12. Make Changes to Other Related Products

Changes to NVAS

Use the following procedure to make changes to NVAS when you install NPM:

Step 1. Concatenate the NPM load module library in the NetView Access Services (NVAS) startup procedure, as shown in the following example:

```
//NVAS1  PROC
//******************************************************************************
// START JCL FOR NETVIEW ACCESS SERVICES *
//******************************************************************************
// NVAS EXEC PGM=EMSMAIN,TIME=144/zerodot,REGION=2/zerodot/zerodot/zerodotK
// STEPLIB DD DSN=EMS.V2R1M1.SEMSLMD/zerodot,DISP=SHR
// DD DSN=NPM.V2R4M/zerodot.SFNMLMD1,DISP=SHR
// SYSPRINT DD SYSOUT=/c5197...
```

Step 2. Ensure that NSI has been defined as a subsystem in the IEFSSNxx member and that FNMASMAN has been defined in the SCHEDxx member of SYS1.PARMLIB.

Changes to TPX

Use the following procedure to make changes to TPX when you install NPM:

Step 1. Ensure that the Activate NetSpy** Interface Option on the TPX System Options Detail Panel is set to yes by coding the following:

```
Activate NetSpy Interface: Y
```

Step 2. Verify that the applications for which you want to collect data are not defined as SHR on the TPC Application Characteristics Detail panel. Applications should be defined as GRP, as shown in the following example:

```
TYPE (SHR,GRP,UNQ,TPX): GRP
```

See the online TPX Administration manual for more information.

**Note:** For virtual terminals that have created a session prior to NPM collection, NPM cannot correlate the control and relay session data until the users of these terminals switch out of the application sessions and then switch back into them.

Changes to CICS

NPM can use the standard Customer Information Control System (CICS*) send exit (XZCOUT) to control definite response at the end of a logical transaction. This allows NPM to collect network times for logical units in session with CICS applications.

Use the following procedure to install the CICS XZCOUT send exit:

Step 1. Link the exit into the CICS program library by using the sample JCL in member FNMDDR.

Step 2. Define the exit to CICS using the following parameters:
Step 12. Make Changes to Other Related Products

PROGRAM=FNMCOUT
GROUP=FNMGROUP (or any other valid group name)
LANGUAGE=ASSEMBLER
RELOAD=NO
RESIDENT=YES
RS1=PUBLIC
STATUS=ENABLED

Step 3. Include the following overrides in the CICS startup JCL:

EXITS=YES
ZCP=$

Step 4. Install the exit using the following command:

CEDA INSTALL GROUP(FNMGROUP)

See CICS Resource Definition Online Manual for more information about the CEDA command.

Step 5. Enable the exit using the following command:

CECI ENABLE PROGRAM(FNMCOUT) EXIT(XZCOUT) START GALENGTH(32)

Note: You must enable the exit prior to starting NPM session collection on any CICS resource; otherwise, session end can occur. Use the CICS program load table (PLT) to automatically enable FNMCOUNT at CICS startup, but before CICS is active. You must still code SESSH=R on the SESSCOLL command because this exit is used in conjunction with DDR, not in place of it.

You can disable the exit by using the following command:

CECI DISABLE PROGRAM(FNMCOUT)

You can re-enable the exit following a disable command (this reloads the exit) by using the following command:

CECI ENABLE PROGRAM(FNMCOUT) EXIT(XZCOUT) START

For more information about the CECI command, see CICS Application Programming Guide.
Chapter 5. Migrating to NPM V2R4 from V1 Releases

This chapter explains how to migrate to NPM Version 2 Release 4 from NPM Version 1 Release 5, 5.1, or 6. This procedure enables you to install NPM Version 2 Release 4 and have it collect the same data you are currently collecting with a previous release.

See Chapter 2, “Installing NPM with the NPM Initialization Program” on page 11 if you want to use the NPM initialization program (NIP) to migrate from a previous release of NPM.

After you migrate from a previous release of NPM, you can install the new NPM Version 2 Release 4 features. See Chapter 1, “Preparing for Installation” on page 3 for the items needed, such as the correct NPM definition to VTAM. See NPM Concepts and Planning for the new NPM function software requirements and for a general description of these functions.

Migrating from NPM V1R5, V1R5.1, or V1R6

Complete the following steps to migrate to NPM Version 2 Release 4 from NPM Version 1 Release 5, 5.1, or 6:

Step 1. Make a backup copy of all the previous release data sets.

Step 2. Make a backup copy of any VTAM exits (ISTEXCAA) you have installed for the previous release of NPM.

Step 3. Make a backup copy of any SMF exits (IEFU83 and IEFU84) you have installed for the previous release of NPM.

Step 4. Follow the steps in the program directory to allocate the NPM Version 2 Release 4 data sets and to install the program tape.

Step 5. For NPM Version 2, the NPM graphic work data sets, the SYSMDUMP data set, and the FNMPROFS partitioned data set are allocated by the FNMLOGDS job instead of the FNMALLOC job.

Step 6. Update the new startup procedure JCL to point to the appropriate data sets. The sample startup procedure JCL is in NPM.V2R4M0.SFNMJCL1(FNMSNPM).

Step 7. APF-authorize the load module library for NPM, if you have not already done so. The name of the load module library for MVS is NPM.V2R4M0.SFNMLMD1. To authorize the library, update the IEAAPFxx member of the SYS1.PARMLIB data set to include the name and volume serial (volser) of the NPM load module library as follows:

NPM.V2R4M0.SFNMLMD1 volser,

Step 8. If you are using VTAM V3R4.1 or a later release, you must concatenate the NPM load library to the VTAM load library. “Step 3. Define NPM to VTAM” on page 30 explains how to concatenate these load libraries.

Step 9. Copy members FNMINIT, FNMOPER, and FNMPROF from the previous release FNMPARM data set to the NPM Version 2 Release 4 FNMPARM data set.
Migrating from V1

Step 10. Reinstall any VTAM exits (ISTEXCAA). See Chapter 9, “Installing NPM Exits” on page 121 to see how to reinstall them. NPM Version 2 Release 4 cannot use any exits built for previous releases of NPM.

Step 11. Reinstall any SMF exits (IEFU83 and IEFU84). NPM Version 2 Release 4 cannot use any exits built for previous releases of NPM.

Step 12. Use the existing FNMIGX41 and IGX00041 graphics modules installed in SYS1.LPALIB.

Step 13. Use the existing FNMUEXIT. Be aware of changes to record formats if you have customized FNMUEXIT to process NPM records. See NPM Reference for more information about records.

Step 14. If you copied the module FNMXIVT0 to SYS1.LPALIB for the previous release, you must remove this module. You do not need this module in SYS1.LPALIB.

Step 15. Make any changes to programs that use NPM records if the record changes affect those programs.

Step 16. If you want to use a customized message table from a previous release of NPM, update it with the new and changed messages for this release. See “Message Table Changes for NPM V2R4” on page 235 to see a list of these new and changed messages.

Step 17. If you are using the NPM graphic subsystem, customize the TSO command list member FNMGMON in NPM.V2R4M0.SFNMJCL1. The TSO command list is used to start the NPM graphic subsystem.

Step 18. The START COLLECT, START DISPLAY, and START MONITOR network collection commands were replaced by the NETCOLL command for shared network collection in NPM Version 2.

Replace all references to these network collection commands in any user-defined or modified NPM EXECs with the NETCOLL command before you install and run NPM Version 2 Release 4.

See “Other NPM V1 Migration Changes Information” on page 77 for more information about migration changes involving the NETCOLL command, NPM Concepts and Planning for more information about shared network collection, and NPM User’s Guide for specific information about the NETCOLL command.

Step 19. The destination parameters for detail and monitor records for all data collections have been updated, and the definition for the base interval has been changed.

Changes have been made to the command syntax used for NPM network collection on the DEFAULTS and NPM initialization statements to support shared network collection for NPM Version 2. You need to update any applicable user-defined batch commands in your system to reflect these changes.

See “Other NPM V1 Migration Changes Information” on page 77 for more information about migration changes and NPM Reference for specific information about the changes to these initialization statements.

Step 20. Start NPM Version 2 Release 4 and verify that the functions you were using with the previous release are working correctly.

Step 21. Install the NPM Version 2 Release 4 functions as appropriate:
• Add the initialization statements for the new functions you are installing. See *NPM Concepts and Planning*.

• Update the NPM installation-wide exit, FNMEEXIT, to process new record formats or to use new exit functions. See “The NPM Installation-Wide Exit” on page 237 to see how to update this exit.

• If you plan to install the NetWare resources data collection function, you must also install (unload) the NPM NetWare Agent code onto the host and download it to the servers.

**Step 22.** Perform the following steps to run NPM Version 2 Release 4 after running Version 1 Release 5, 5.1, or 6:

- **a.** Execute the FNMCCLNUP utility to free NPM’s CSA control blocks. FNMCCLNUP is executed by the sample job FNMCCLNPJ, located in NPM.V2R4M0.SFNMJCL1. Execute this utility only when NPM is not running. See “Cleaning Up NPM’s CSA Control Blocks” on page 207.

- **b.** If you have installed any of the NPM Version 2 Release 4 functions and plan to use the VSAM data sets with Version 1 Release 5, 5.1, or 6, reset the VSAM (FNMRVEx and FNMSEx) data sets. NPM Version 1 Release 5, 5.1, and 6 online analysis functions cannot read the VSAM records written by the NPM Version 2 functions.

**Step 23.** After all functions of NPM Version 2 Release 4 are operating correctly, delete the previous release libraries.

**Other NPM V1 Migration Changes Information**

**NPM-to-NPM Communication Changes**
An NPM Version 2 system can only perform NPM-to-NPM communication with another NPM Version 2 system.

**NPM List Panel Changes**
For NPM Version 2, on all NPM list panels, the TOP OF DATA and BOTTOM OF DATA messages have been replaced with ‘−’ and ‘+’ symbols for panel scrolling, and the REFRESH operator command has been added for displaying new list panel data.

**Changes to FNMINIT**
The following tables show the statement changes you must make to keep all the settings you used in NPM Version 1 Release 6 (V1R6). These tables enable you to perform the migration from NPM V1R6 to NPM Version 2 Release 4.

Table 4 on page 78 shows the parameters that have been changed, deleted, or added to the statements in FNMINIT for NPM Version 2. The changes made for NPM Version 2 Releases 1, 2, 3, and 4 are labeled (V2R1), (V2R2), (V2R3), and (V2R4), respectively.
Table 4. FNMINIT - NPM V1R6 to NPM Version 2 Release 4 Statement and Parameter Comparison

<table>
<thead>
<tr>
<th>Statement</th>
<th>Changed Parameters</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULTS</td>
<td>INTERVAL</td>
<td>DISPLAY</td>
<td>VTAMINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOG</td>
<td>LWGIINT (V2R2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REVIEW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XALERT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XDISPLAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XLOG</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XREVIEW</td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td>CMDS</td>
<td>LOG</td>
<td>INTERVAL (V2R3)</td>
</tr>
<tr>
<td>NPM</td>
<td></td>
<td>SAMPVADR (V2R4)</td>
<td>LAN (V2R1)</td>
</tr>
<tr>
<td></td>
<td>DNC</td>
<td>SAMPVAPL (V2R4)</td>
<td>LANX (V2R1)</td>
</tr>
<tr>
<td></td>
<td>EVENT</td>
<td>SAMPVAPN (V2R4)</td>
<td>LUGROUP (V2R3)</td>
</tr>
<tr>
<td></td>
<td>NETWORK</td>
<td>SAMPVBPL (V2R4)</td>
<td>LWG (V2R2)</td>
</tr>
<tr>
<td></td>
<td>RTM</td>
<td>SAMPVDEV (V2R4)</td>
<td>LWGX (V2R2)</td>
</tr>
<tr>
<td></td>
<td>SESSION</td>
<td>SAMPVGBL (V2R4)</td>
<td>MAXWS (V2R1)</td>
</tr>
<tr>
<td></td>
<td>SYNCH</td>
<td>SAMPVVRR (V2R4)</td>
<td>NETWORKX (V2R1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NWCOLL (V2R2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>REFRESH (V2R1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SESSIONX (V2R1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TELNET (V2R4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VTAM (V2R1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VTAMCOLL (V2R1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>VTAMX (V2R1)</td>
</tr>
<tr>
<td></td>
<td>SAMP (V2R4)</td>
<td></td>
<td>SAMPVHI (V2R4)</td>
</tr>
</tbody>
</table>

Required Processing on FNMINIT

You must process each one of the changed or deleted NPM V1R6 parameters shown in Table 4. You must also map the deleted NPM V1R6 parameters before deleting them from the NPM Version 2 statements for parameters that might pertain to the same statement.

Table 5 through Table 7 on page 79 show NPM Version 2 parameters that must be coded differently from NPM V1R6.

Table 5 (Page 1 of 2). Mapping NPM V1R6 DEFAULTS Statement Parameters to NPM Version 2

<table>
<thead>
<tr>
<th>NPM V1R6 Parameter</th>
<th>NPM Version 2 Statement</th>
<th>NPM Version 2 Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERVAL=8</td>
<td>DEFAULTS</td>
<td>INTERVAL=7</td>
</tr>
<tr>
<td>LOG=NO</td>
<td>NPM</td>
<td>NETWORK=NPMLOG</td>
</tr>
<tr>
<td>REVIEW=NO</td>
<td>NPM</td>
<td>NETWORK=VSAM</td>
</tr>
</tbody>
</table>

Note: Both INTERVAL parameters show the maximum collection interval. The maximum collection interval for NPM Version 2 is 7. You can set INTERVAL to any number from 1–7. 1 is the default.
Table 5 (Page 2 of 2). Mapping NPM V1R6 DEFAULTS Statement Parameters to NPM Version 2

<table>
<thead>
<tr>
<th>NPM V1R6 Parameter</th>
<th>NPM Version 2 Statement</th>
<th>NPM Version 2 Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>XALERT=YES</td>
<td>NPM</td>
<td>NETWORKX=ALERT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SESSIONX=ALERT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANX=ALERT</td>
</tr>
<tr>
<td>XDISPLAY=YES</td>
<td>NPM</td>
<td>LANX=GLOBAL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NETWORKX=GLOBAL</td>
</tr>
<tr>
<td>XLOG=YES</td>
<td>NPM</td>
<td>NETWORKX=NPMLOG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANX=NPMLOG</td>
</tr>
<tr>
<td>XREVIEW=YES</td>
<td>NPM</td>
<td>NETWORKX=VSAM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LANX=VSAM</td>
</tr>
</tbody>
</table>

Table 6. Mapping NPM V1R6 NPM Statement Parameters to NPM Version 2

<table>
<thead>
<tr>
<th>NPM V1R6 Parameter</th>
<th>NPM Version 2 Statement</th>
<th>NPM Version 2 Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMD</td>
<td>NPM</td>
<td></td>
</tr>
<tr>
<td>DNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EVENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NETWORK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMDS</td>
<td>NPM</td>
<td>If you explicitly coded one of these parameters on the NPM statement for NPM V1R6, use the same value for NPM Version 2. If you did not explicitly code one of these parameters for NPM V1R6, but you did explicitly code the LOG parameter (which has been eliminated in NPM Version 2), code these parameters using the same value coded on the LOG parameter. If you did not explicitly code either these parameters or the LOG parameter, do not code these parameters for NPM Version 2.</td>
</tr>
<tr>
<td>DNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EVENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NETWORK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYNCH=(8,mm)</td>
<td>SYNM</td>
<td>SYNM=(7,mm)</td>
</tr>
<tr>
<td>Note: Both SYNCH parameters show the maximum synchronization intervals for NPM network and session data collection. The maximum collection interval for NPM Version 2 is 7.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7 (Page 1 of 2). Mapping NPM V1R6 TASK Statement Parameters to NPM Version 2 for VTAM Statistics Collection

<table>
<thead>
<tr>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO TASK STATEMENT CODED</td>
<td>DO NOT CODE ANY STATEMENT</td>
</tr>
</tbody>
</table>
Migrating from V1

Table 7 (Page 2 of 2). Mapping NPM V1R6 TASK Statement Parameters to NPM Version 2 for VTAM Statistics Collection

<table>
<thead>
<tr>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK STATEMENTS CODED</td>
<td>FNMVDC00 has been added to the TASK statement as the default task for VTAM statistics data collection in NPM Version 2. FNMVDC00 must have a priority value of 255. If you coded 255 as the maximum priority value on a PRTY parameter for NPM V1R6, you must decrease all priority values on all NPM V1R6 PRTY parameters by at least 1. If you coded a PRTY parameter with a maximum value that was less than 255 in NPM V1R6, you must code the following parameters on the TASK statement for NPM Version 2:</td>
</tr>
<tr>
<td></td>
<td>TASK NAME=FNMVDC00,PRTY=255</td>
</tr>
<tr>
<td></td>
<td>In NPM Version 2 Release 2, a parameter called NETWARE was added to FNMFST00. If you code it as NETWARE=YES, you can write NetWare resources data collection records. The default is NETWARE=NO.</td>
</tr>
</tbody>
</table>

Changes to the FNMSCMDS and FNMSTRT Data Sets

The START COLLECT and START MONITOR commands that were used for network collection in NPM V1R6 have been replaced by the NETCOLL command in NPM Version 2 releases.

Table 8 shows the new or changed commands for NPM V2 and lists the affected parameters.

**Note:** See NPM User's Guide for more information about the parameters of these new commands.

Table 8 (Page 1 of 2). FNMSTRT and FNMSCMDS - NPM V2 R1, R2, R3 to NPM V2R4 Command and Parameter Comparisons

<table>
<thead>
<tr>
<th>Command</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>START COLLECT</td>
<td>NCP_name, interval_number</td>
<td>NCP</td>
</tr>
<tr>
<td></td>
<td>DISPLAY</td>
<td>INTERVAL</td>
</tr>
<tr>
<td></td>
<td>LOG</td>
<td>DEST</td>
</tr>
<tr>
<td></td>
<td>REVIEW</td>
<td>OPTION</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
</tr>
<tr>
<td>START MONITOR</td>
<td>NCP_name, interval_number</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>CRITERIA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DATA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DISPLAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REVIEW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XALERT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XDISPLAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XLOG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XREVIEW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XROUTE</td>
<td></td>
</tr>
</tbody>
</table>
### Table 8 (Page 2 of 2). FNMSRT and FNMSCMDS - NPM V2 R1, R2, R3 to NPM V2R4 Command and Parameter Comparisons

<table>
<thead>
<tr>
<th>Command</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETCOLL</td>
<td>N/A</td>
<td>BACKCONG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BUFUSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYTERAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYTERCVD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYTETRAN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CCUUTIL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONGEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DEST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DESTX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DISDATAG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DLFRAMES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DUTIL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ERRCNTS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FWDCONG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FREEBUFS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INTERVAL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LINEUTL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MORECOLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSGRATE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NCP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEGPOLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ONECOLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OPTION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PDUSDISC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PDURCVD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PDUTRAN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POSPOLLO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PUTIL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>QUELEN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REXMITB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REXMIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RNRRCVD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RNRTRAN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RTIMEOUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SLOWDOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUTIL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TICUTIL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TRANSDEF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XRES</td>
</tr>
<tr>
<td>INTERVAL (V2R3)</td>
<td>SAMPVADR (V2R4)</td>
<td>SESSINT (V2R3)</td>
</tr>
<tr>
<td></td>
<td>SAMPVAPL (V2R4)</td>
<td>SAMPVHI (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVAPN (V2R4)</td>
<td>SAMPVMD (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVBL (V2R4)</td>
<td>SAMPVWL (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVDEV (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMPVGBL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMPVVVR (V2R4)</td>
<td></td>
</tr>
<tr>
<td>LUGROUP (V2R3)</td>
<td></td>
<td>NAME (V2R3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OPTION (V2R3)</td>
</tr>
<tr>
<td>VAPNCOLL (V2R3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VCSMCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VMNPCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VRTPCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Required Processing for FNMSCMDS and FNMSTRT

In NPM Version 2, shared network collection processing enables you to issue one NETCOLL command per interval for each resource. To migrate FNMSTRT, or any other NPM EXEC, you must accumulate a number of START COLLECT and START MONITOR commands into one NETCOLL command. In NPM Version 2, a NETCOLL command is uniquely identified by the following parameters:

- Resource name
- NCP name (if coded)
- Interval number
- Start time
- Stop time

Table 9 shows how to map the NPM V1R6 START COLLECT and START MONITOR command parameters that have been deleted for NPM Version 2 into parameters for the NETCOLL command.

<table>
<thead>
<tr>
<th>NPM V1R6 Parameter</th>
<th>NPM Version 2 Parameter</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCP_name</td>
<td>NCP</td>
<td>NCP_name is a positional optional parameter. If you coded the NCP parameter in NPM V1R6, you must code NCP=NCP_name for NPM Version 2.</td>
</tr>
<tr>
<td>interval_number</td>
<td>INTERVAL</td>
<td>interval_number is a positional optional parameter. If you coded an interval number that was less than 8 in NPM V1R6, use the same interval number in NPM Version 2. However, if you coded an interval number that was equal to 8 in NPM V1R6, you may want to use 2 as the interval number for NPM Version 2.</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>LOG</td>
<td>DEST</td>
<td>If you coded LOG=YES in NPM V1R6, or did not code the DEST parameter, code DEST=NPMLOG for NPM Version 2.</td>
</tr>
<tr>
<td>REVIEW</td>
<td>DEST</td>
<td>If you coded REVIEW=YES in NPM V1R6, code DEST=VSAM for NPM Version 2.</td>
</tr>
<tr>
<td>ROUTE</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Table 9 (Page 2 of 2). Mapping of NPM V1R6 Command Parameters to NPM Version 2

<table>
<thead>
<tr>
<th>NPM V1R6 Parameter</th>
<th>NPM Version 2 Parameter</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA=value</td>
<td>BACKCONG</td>
<td></td>
</tr>
<tr>
<td>CRITERIA=(low limit, high limit)</td>
<td>BUFUSE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYTERAT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYTERCVD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYTETRAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CCUUTIL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONGEST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DISDATAG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DLFRAMES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ERRNTS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FWDCONG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FREEBUF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LINEUTL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MORECOLL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MSGRATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NEGPOLL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ONECOLL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDUSDISC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDURCVD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDUTRAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>POSPOLL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>QUELEN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REXMITB</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REXMIT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RNRCVCD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RNRTRAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RTIMEOUT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SLOWDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SLOWDOWN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TICUTIL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRANSDEF</td>
<td></td>
</tr>
</tbody>
</table>

You should code a MONITOR TYPE parameter equal to the following values for each pair of data and criteria parameters:

- If you explicitly coded a low limit in NPM V1R6, code `low threshold=low limit` for NPM Version 2.
- If you did not explicitly code a low limit in NPM V1R6, code `low threshold=0` for NPM Version 2.
- If you explicitly coded a high limit in NPM V1R6, code `high threshold=high limit` for NPM Version 2.
- If you did not explicitly code a high limit in NPM V1R6, code `high threshold=0` for NPM Version 2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DESTX</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>XALERT</td>
<td>DESTX</td>
<td>If you coded XALERT=YES in NPM V1R6, code DESTX=ALERT for NPM Version 2.</td>
</tr>
<tr>
<td>XDISPLAY</td>
<td>DESTX</td>
<td>If you coded XDISPLAY=YES in NPM V1R6, or if you did not code XDISPLAY, code DESTX=GLOBAL for NPM Version 2.</td>
</tr>
<tr>
<td>XLOG</td>
<td>DESTX</td>
<td>If you coded XLOG=YES in NPM V1R6, or if you did not code XLOG, code DESTX=NPMLOG for NPM Version 2.</td>
</tr>
<tr>
<td>XREVIEW</td>
<td>DESTX</td>
<td>If you coded XREVIEW=YES in NPM V1R6, or if you did not code XREVIEW, code DESTX=REVIEW for NPM Version 2.</td>
</tr>
<tr>
<td>XROUTE</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Note: At the end of mapping, if you did not code the DEST or DESTX parameters on the NETCOLL command, code DEST=None and DESTX=None for NPM Version 2.
Chapter 6. Migrating to NPM V2R4 from Previous V2 Releases

This chapter explains how to migrate to NPM Version 2 Release 4 from NPM Version 2 Release 1, 2, or 3. This procedure enables you to install NPM Version 2 Release 4 and have it collect the same data you are currently collecting with a previous release. See Chapter 2, “Installing NPM with the NPM Initialization Program” on page 11 if you want to use the NPM initialization program (NIP) to migrate from a previous release of NPM.

After you migrate from a previous release of NPM, you can install the new NPM Version 2 Release 4 features. See Chapter 1, “Preparing for Installation” on page 3 for the items needed, such as the correct NPM definition to VTAM. See NPM Concepts and Planning for the new NPM function software requirements and for a general description of these functions.

Migrating from NPM V2R1, V2R2, or V2R3

Complete the following steps to migrate to NPM Version 2 Release 4 from Version 2 Release 1, 2, or 3.

Step 1. Make a backup copy of all the previous release data sets.

Step 2. Make a backup copy of any VTAM exits (ISTEXCAA) you have installed for the previous release of NPM.

Step 3. Make a backup copy of any SMF exits (IEFU83 and IEFU84) you have installed for the previous release of NPM.

Step 4. Follow the steps in the program directory to allocate the NPM Version 2 Release 4 data sets and to install the program tape.

Step 5. Update the new startup procedure JCL to point to the appropriate data sets. The sample startup procedure JCL is in NPM.V2R4M0.SFNMJCL1(FNMSNPM).

If you are migrating from NPM Version 2 Release 1, 2, or 3 you can customize your existing JCL to include the new Version 2 Release 4 information. Figure 18 on page 177 shows the startup procedure JCL.

Step 6. APF-authorize the load module library for NPM, if you have not already done so. The name of the load module library for MVS is NPM.V2R4M0.SFNMLMD1. To authorize the library, update the IEAAPFxx member of the SYS1.PARMLIB data set to include the name and volume serial (volser) of the NPM load module library as follows:

NPM.V2R4M0.SFNMLMD1 volser,

Step 7. If you are using VTAM V3R4.1 or a later release, you must concatenate the NPM load library to the VTAM load library. "Step 3. Define NPM to VTAM" on page 30 explains how to concatenate these load libraries.

Step 8. Copy members FNMINIT, FNMOPER, and FNMPROF from the previous release FNMPARM data set to the NPM Version 2 Release 4 FNMPARM data set.

Step 9. (If you are migrating from NPM Release 2.1), you must make the following changes if you want to collect LAN or NetWare resource data:
Migrating from Previous V2 Releases

• Change NPM's NetView program DST definition from FNMDST to FNMSYRN in the VTAMLST to define NPM's NetView program DST to VTAM. Code:

  FNMSYRN       APPL AUTH=CNM

• Add the CMDMDL statement FNM62RDV to the NetView DSICMD member. Code:

  FNM62RDV CMDMDL MOD=FNMDCNMI,TYPE=D,RES=Y,PARSE=N

• Change the TASK statement FNMDST to FNMSYRN in the NetView DSIDMN member. Code:

  TASK        MOD=DSIZDST,TSKID=FNMSYRN,MEM=FNMDSTD,PRI=3,INIT=Y

• To ensure that the NPM DST has been started in the NetView program, code either INIT=Y on the TASK statement used to define the DST, or issue the following command from the command facility in the NetView program:

  START TASK=FNMSYRN

**Step 10.** Reinstall any VTAM exits (ISTEXCAA). See Chapter 9, “Installing NPM Exits” on page 121 to see how to reinstall them. NPM Version 2 Release 4 cannot use any exits built for previous releases of NPM.

Reinstall any SMF exits (IEFU83 and IEFU84). NPM Version 2 Release 4 cannot use any exits built for previous releases of NPM.

**Step 11.** Use the existing FNMIGX41 and IGX00041 graphics modules installed in SYS1.LPALIB.

**Step 12.** Use the existing FNMUEXIT. Be aware of changes to record formats if you have customized FNMUEXIT to process NPM records. See *NPM Reference* for more information about records.

**Step 13.** If you copied the module FNMXIVT0 to SYS1.LPALIB for the previous release, you must remove this module. You do not need this module in SYS1.LPALIB.

**Step 14.** Make any changes to programs that use NPM records if the record changes affect those programs.

**Step 15.** If you want to use a customized message table from a previous release of NPM, update it with the new and changed messages for this release. See “Message Table Changes for NPM V2R4” on page 235 to see a list of these new and changed messages.

**Step 16.** If you are using the NPM graphic subsystem, customize the TSO command list member FNMGMON in NPM.V2R4M0.SFNMJCL1. The TSO command list is used to start the NPM graphic subsystem.

**Step 17.** Start NPM Version 2 Release 4 and verify that the functions you were using with the previous release are working correctly.

**Step 18.** Install the NPM Version 2 Release 4 functions as appropriate:

  a. Add the initialization statements for the new functions you are installing. See *NPM Concepts and Planning*.

  b. Update the NPM installation-wide exit, FNMUEXIT, to process new record formats or to use new exit functions. See “The NPM Installation-Wide Exit” on page 237 to see how to update this exit.
c. If you plan to install the NetWare resources data collection function, you must also install (unload) the NPM NetWare Agent code onto the host and download it to the servers. See Chapter 8, “Installing NPM Netware Resource Collection” on page 117 to see how to perform the installation and download procedures.

**Step 19.** Perform the following steps to run NPM Version 2 Release 4 after running NPM Version 2 Release 1, 2, or 3:

a. Execute the FNMCNUP utility to free NPM's CSA control blocks.

   FNMCNUP is executed by the sample job FNMCNUPJ, located in NPM.V2R4M0.SFNMCNUPJL. Execute this utility only when NPM is not running. See “Cleaning Up NPM’s CSA Control Blocks” on page 207.

b. If you have installed any of the NPM Version 2 Release 4 functions and plan to use the VSAM data sets with Version 1 Release 5, 5.1, or 6, reset the VSAM (FNMRVXX and FNMSX) data sets. NPM Version 1 Release 5, 5.1, and 6 online analysis functions cannot read the VSAM records written by the NPM Version 2 functions.

**Step 20.** After all functions of NPM Version 2 Release 4 are operating correctly, delete the previous release libraries.

### Migration Changes from Previous NPM Version 2 Releases

The tables in this section show the statement changes you can make to keep all the settings you used in NPM Version 2 Release 1, 2, or 3. These tables enable you to perform the migration from NPM V2 R1, R2, or R3 to NPM Version 2 Release 4. If you are migrating from Version 2 Release 1, you need to make those changes that are labeled for Version 2 Releases 2, 3, and 4. If you are migrating from Version 2 Release 2, you need to make those changes that are labeled for Version 2 Releases 3 and 4. If you are migrating from Version 2 Release 3, you need to make only those changes that are labeled as belonging to Version 2 Release 4.

### Changes to FNMINIT Statements

Table 10 shows the parameters that have been changed, deleted, or added to the FNMINIT statements for NPM Version 2. The changes made for NPM Version 2 Release 2 are labeled (V2R2). The changes made for NPM Version 2 Release 3 are labeled (V2R3). The changes made for NPM Version 2 Release 4 are labeled (V2R4).

<table>
<thead>
<tr>
<th>Statement</th>
<th>Changed Parameters</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULTS</td>
<td>INTERVAL</td>
<td>DISPLAY</td>
<td>VTAMINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOG</td>
<td>LWGINT (V2R2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REVIEW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XALERT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XDISPLAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XLOG</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XREVIEW</td>
<td></td>
</tr>
<tr>
<td>FILE</td>
<td>WRAP (V2R3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Migrating from Previous V2 Releases

Table 10 (Page 2 of 2). FNMINIT - NPM V2 (R1, 2, & 3) to NPM V2R4 Statement and Parameter Comparison

<table>
<thead>
<tr>
<th>Statement</th>
<th>Changed Parameters</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPM CMDS</td>
<td></td>
<td>LOG</td>
<td>INTERVAL (V2R3)</td>
</tr>
<tr>
<td></td>
<td>DNC</td>
<td>SAMPVADR (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EVENT</td>
<td>SAMPVAPL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NETWORK</td>
<td>SAMPVAPN (V2R3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NSA</td>
<td>SAMPVBPL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RTM</td>
<td>SAMPVDEV (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SESSION</td>
<td>SAMPVGBL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYCH</td>
<td>SAMPVVR (V2R4)</td>
<td></td>
</tr>
<tr>
<td>SAMP (V2R4)</td>
<td></td>
<td></td>
<td>SAMPVHI (V2R4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SAMPVHI (V2R4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SAMPVHI (V2R4)</td>
</tr>
</tbody>
</table>

**New Commands for NPM V2R4**

Table 11 shows new or changed commands and their parameters that have been added or deleted for NPM Version 2 Release 4.

Table 11. FNMSSTR and FNMSCMDS - NPM V2 R1, R2, R3 to NPM V2R4 Command and Parameter Comparisons

<table>
<thead>
<tr>
<th>Command</th>
<th>Deleted Parameters</th>
<th>Added Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERVAL (V2R3)</td>
<td>SAMPVADR (V2R4)</td>
<td>SAMPVHI (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVAPL (V2R4)</td>
<td>SAMPVMD (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVAPN (V2R4)</td>
<td>SAMPVLW (V2R4)</td>
</tr>
<tr>
<td></td>
<td>SAMPVBPL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMPVDEV (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMPVGBL (V2R4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMPVVR (V2R4)</td>
<td></td>
</tr>
<tr>
<td>SESSCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VCSMCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VMNPCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VRTPCOLL (V2R4)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 7. Installing the NPM Desk/2 User Interface

This chapter describes the two separate procedures required to install NPM Desk/2 for NPM:

- You must configure the NPM host to communicate with an NPM OS/2 workstation. See the procedure provided in “Define Desk/2 Host Configuration.”
- You must install NPM Desk/2 on your NPM OS/2 workstation. See the procedure provided in “NPM Desk/2 Workstation Installation” on page 92.

Note: You cannot use NPM Desk/2 to collect session data with NPM V2R4.

Define Desk/2 Host Configuration

Use the following procedure to set up the host configuration. NPM host configuration is required for the host to communicate with an NPM OS/2 workstation using an LU 6.2 session. The NPM OS/2 workstation interfaces are NPM Desk/2 and NPM Performance Monitor of SNA Networks.

Step 1. Customize your existing logmode table entry to allow NPM to communicate with an NPM OS/2 workstation through an LU 6.2 session.

The following example shows how to code the logmode table entry to enable either of the OS/2 workstation interfaces to use an LU 6.2 session.

```
FNML6MOD MODEENT LOGMODE=FNML6MOD,
    FMPROF=X'13',
    TSPROF=X'07',
    PRIPROT=X'B0',
    SECPROT=X'B0',
    COMPROT=X'50B5',
    RUSIZES=X'8686',
    SSNDPAC=X'00',
    SRCVPAC=X'03',
    PSNDPAC=X'03',
    TYPE=0,
    PSERVIC=X'06020000000000000000000F00'
```

An example of how to code the logmode table definition is provided in NPM.V2R4M0.SFNMJCL1(FNML6MOD), which is provided with this release of NPM. An example showing how to assemble and link-edit the logmode table is provided in NPM.V2R4M0.SFNMJCL1(FNMASMLK).

See VTAM Programming and VTAM Resource Definition Reference for more information about how to assemble and link-edit the logmode table.

If you are transmitting large amounts of data or are experiencing an overload on the communication link, you can increase the RU sizes to 2048 bytes. See VTAM Resource Definition Reference for more information about how to increase RU sizes.

Step 2. You must establish and activate a session between the host and the NPM OS/2 workstation on your network to enable the NPM OS/2 workstation to use an LU 6.2 session.

To establish communication between the host and the NPM OS/2 workstation interface you are using, you must define the type of
workstation-to-host connections in your VTAM and NCP definitions. The following list provides two example types used for this connection:

a. LAN-attached token-ring connections  
b. SDLC line connections

**Step 3.** You must update VTAM and the NCP generation EXEC to define an NPM OS/2 workstation to a LAN to use an LU 6.2 session. See NCP Resource Definition Guide for more information.

The following example shows how an NPM OS/2 workstation would be defined in an NCP major node.

```  
A/zerodot1PGRP1 GROUP ECLTYPE=(PHYSICAL,PERIPHERAL),  
    TYPE=NCP,  
    DIAL=NO,  
    LNCTL=SDLC,  
    LEVEL2=ECLNARL2,  
    LEVEL3=ECLNARL3,  
    LEVEL5=NCP,  
    TIMER=(ECLNART1,,ECLNART2,ECLNART3),  
    XIO=(ECLNARXL,ECLNARXS,ECLNARXI,ECLNARXK),  
    USERID=(5668854,ECLRBDT,NORECMS,,ECLNMVT),  
    MAXPU=1,  
    SPEED=9600,  
    PUTYPE=1,  
    PUDR=NO,  
    COMPTAD=YES,  
    COMPSWP=YES,  
    COMPOWN=YES  
```

```  
A/zerodot1L/zerodot88 LINE ADDRESS=(1/zerodot88,FULL),  
    PORTADD=/zerodot,  
    LOCADD=4/zerodot/zerodot/zerodot1FFFFF3F,  
    RCVBUFC=4/zerodot95,  
    MAXTSL=2/zerodot12,  
    ADAPTER=TIC2,  
    TRSPEED=16,  
    UACB=(X$P1AX,X$P1AR)  
A/zerodot1PU88 PU MAXOUT=7,  
    ADDR=01,  
    ANS=CONTINUE  
A/zerodot1LU88 LU ISTATUS=INACTIVE,  
    LOCADDR=/zerodot  
```

The following example shows how to define LAN-attached token-ring NPM OS/2 workstations in a switched major node to correctly identify them to VTAM to use an LU 6.2 session. The LOCADDR=00 LU is used to define the LU 6.2 host connection for either of the NPM OS/2 workstation interfaces. The other four LUs (LU 2.0s) are used to define the 3270 emulation screens.

See VTAM Resource Definition Reference for more information about the configuration represented in the example.
Define Desk/2 Host Configuration

Step 4. When you define an NPM OS/2 workstation with a synchronous data link control (SDLC) connection, you need to update the NCP major node. The following example shows how to update the NCP major node. The LOCADDR=00 LU is used to define the LU 6.2 connection that the NPM OS/2 workstation interfaces use. The four other LUs (LU 2.0s) are used to define the 3270 emulation screens. See VTAM Resource Definition Reference for more information about the configuration represented in the following example.

```
***********************************************************************
TKSMN  VBUILD  TYPE=SNET,MAXGRP=30,MAXNO=200
*
  *  NPM Desk/2 number 1
A01LP01  PU  IDBLK=050,IDNUM=FFF01,ADDR=01,MAXPATH=1,MODETAB=AMODETAB,
  X
  USSTAB=AUSSTAB,DLOGMOD=MSDLQ
  PATH  DIALNO=00:44000:FFFFF1
A01LT010  LU  LOCADDR=00,MODETAB=NPM21TAB,DLOGMOD=FNML6MOD
A01LT011  LU  LOCADDR=01
A01LT012  LU  LOCADDR=02
A01LT013  LU  LOCADDR=03
A01LT014  LU  LOCADDR=04
***********************************************************************
```

**Step 5.** Update the NPM Initialization Statement for NPM OS/2 Workstations to Use an LU 6.2 Session.

---

**Note:** The LU names that you specify here must also be specified when you define the OS/2 Communications Manager workstation configuration. See NPM Desk/2 User's Guide and Using Performance Management of SNA Networks for more information.
The following example shows how to define the MAXWS keyword on the NPM initialization statement in FNMINIT, which specifies the maximum number of workstations that can be attached to one NPM simultaneously. The default value is 3.

```
NPM ALERT=NO,
CMDS=(NPMLOG),
CONFIG=YES,
MAXWS=3
```

You can set MAXWS to any number from 0–99. This value must be lower than or equal to the value specified on the MAXOPS keyword on the SYS statement. If the value on the MAXWS keyword is greater than the value specified on the MAXOPS keyword on the SYS statement, the MAXWS value is internally reset to the value specified on the MAXOPS keyword.

Setting MAXWS to 0 means that you do not want an LU 6.2 connection. If you code MAXWS=0, no control blocks are created for LU 6.2 management, and no ACB is opened (if you specified the ACB_name on the VTAM statement).

MAXWS is internally reset to 0 if you do not specify the LU62 keyword on the VTAM statement.

---

**NPM Desk/2 Workstation Installation**

The following steps describe how to install NetView Performance Monitor (NPM) Desk/2 on your OS/2 workstation. The detailed installation steps include:

- Downloading and unpacking the product files
- Defining the NPM Desk/2 attended or unattended installation programs
- Generating the NPM Desk/2 setup profile file
- Using the unattended installation program
- Using the attended installation program

Use the following procedure to install NPM Desk/2 on your OS/2 workstation:

**Step 1.** Verify that your hardware and software meet the specific NPM Desk/2 requirements/recommendations listed below:

- NPM Desk/2 requires an IBM Personal System/2* (PS/2*) or compatible computer with the following configuration:
  - 6MB of random access memory (RAM)
  - 8MB of available space on the hard disk
  - 10MB of available disk space for memory swapping, in addition to the requirements of other applications (such as OS/2 2.1)
  - VGA, XGA*, or SVGA video interface
  - 386 20MHz processor (with an 80386 chip-step level higher than B0 or B1)
  - Communications card

- The following IBM PS/2 (or other compatible hardware configuration) is recommended to increase NPM Desk/2 performance:
  - 10MB of RAM
NPM Desk/2 Workstation Installation

- 8MB of available space on the hard disk
- 10MB of available disk space for memory swapping, in addition to the requirements of other applications (such as OS/2 2.1)
- 386/DX 33MHz processor (with an 80386 chip-step level higher than B0 or B1)
- Mouse device

Although not required, you should partition the workstation hard disk so that you have at least two separate drives. OS/2 system files should reside on one drive, and the NPM Desk/2 program files and other applications should reside on a separate drive. This disk partitioning simplifies OS/2 system upgrades.

The selection of the PS/2 (or compatible) model size, DASD, and the amount of memory depends on performance factors. These factors include the size of the supported network, the number of Configuration and DataView windows, and the amount of network activity.

- NPM Desk/2 has the following software requirements:
  - OS/2 Version 2.1 with the Restructured Extended Executor (REXX) language feature installed, and one of the following:
    - Communications Manager/2 Version 1.x
    - Extended Services (ES) for OS/2 Version 1.0 with the Utilities feature installed
  - If you are installing NPM Desk/2 from a host running under MVS, NPM Desk/2 requires the following software at the host:
    - For English, the 3270 PC File Transfer Program Version 1 Release 1 Modification level 1 or a later release
    - For Japanese, the PS/55 File Transfer program

**Step 2.** Download and unpack the NPM Desk/2 product files. NPM Desk/2 is on the NPM host product distribution tape and is contained in two compressed files. One file contains NPM Desk/2's product files and the files unique to the English language version. The second file, which is optional, contains the files for other languages (National Language Support). Load the files from the distribution tape to the host using the system modification program/extended (SMP/E) program for MVS. See the NPM program directory for more information about downloading the distribution tape files to your host.

**Step 3.** Choose a method for installing NPM Desk/2. Installation methods for NPM Desk/2 differ according to the level of Communications Manager present on your workstation. NPM Desk/2 provides installation programs for both the Extended Services for OS/2 (OS/2 ES) and CM/2 Version 1.x environments.

The installation procedure customizes Communications Manager and the CONFIG.SYS file.

**Note:** Attended and unattended installation methods are a part of the NPM Desk/2 installation program.
**Warning:** The NPM Desk/2 installation program updates your Communications Manager configuration file automatically. If you have a highly customized Communications Manager configuration file, you should use the instructions to install NPM Desk/2 manually. Your customized Communications Manager configuration might be adversely affected by using the NPM Desk/2 installation program.

The available installation methods are:

- **Unattended installation**—The unattended installation method does not require an operator to be present at the workstation where NPM Desk/2 is being installed. Use this type of installation to install NPM Desk/2 on multiple workstations where the customization is done after installation.

- **Attended installation**—The attended installation method requires an operator to be present to initiate the installation program and to respond to the various prompts that are displayed during the installation and configuration processes. Use this type of installation to install NPM Desk/2 on single workstations where the customization is done during installation.

- **Manual installation**—Manual installation is an optional alternative to the standard attended or unattended installation methods.

After you have reviewed the installation requirements and downloaded and unpacked the NPM Desk/2 product files to your workstation, see the *NPM Desk/2 User’s Guide* for the changes you need to make to your Communications Manager configuration file and CONFIG.SYS file.

**Step 4.** You must configure the startup profile to be able to log on to the NPM host. If you use the attended installation and select full customization, the installation program will help you to customize the setup profile. In all other cases, you must first start NPM Desk/2 (see the *NPM Desk/2 User’s Guide*). Also, before you log on to the NPM host, you must configure the setup profile (see the *NPM Desk/2 User’s Guide*).

**Step 5.** Use the following procedure to create a directory in which to store the NPM Desk/2 program files before you download the files to your workstation:

- **a.** Determine the drive on which you will install NPM Desk/2.

- **b.** Make the drive on which you are installing NPM Desk/2 the current drive by typing the letter of the drive followed by a colon, at the OS/2 command prompt. For example, type C: and press Enter to make C the current drive.

- **c.** Create a directory for NPM Desk/2 by typing `md IBMNPM`, and pressing Enter.

- **d.** Make the IBMNPM directory the current directory by typing `cd IBMNPM`, and pressing Enter.

**Step 6.** Use the following procedure to download the NPM Desk/2 product files to your workstation:

- **a.** Start Communications Manager.

- **b.** Establish an OS/2 session.

- **c.** Establish a 3270 emulation session.
d. From your emulator session, log on to TSO and place your TSO session in a ready state.

Note: Specify a log mode that meets the requirements of the 3270 PC File Transfer program. To establish a session between the host and your workstation, this log mode must have the queryable bit (in the PSERVIC part of the mode entry) set in the logon mode.

e. Return to the OS/2 session and ensure that you are in the directory where you want to download the files. Download the host files as binary files.

For MVS, use the RECEIVE command to download NPM Desk/2 (DCZVEUI) from the host data set (SFNMEUI1) to the workstation where it is renamed to DCZVEUI.ZIP. For example:

```
RECEIVE DCZVEUI.ZIP a:'NPM.V2R4M0.SFNMEUI1(DCZVEUI)'
```

where `a` is the terminal emulator session you are using for TSO.

f. To install the Japanese feature from an MVS environment, you need to receive both the base English file and the Japanese file. Also, add a left bracket at the end of the RECEIVE command. For example:

```
RECEIVE DCZVEUI.ZIP a:'NPM.V2R4M0.SFNMEUI1(DCZVEUI)'
RECEIVE DCZVEUIJ.ZIP a:'NPM.V2R4M0.SFNMEUI1(DCZVEUIJ)'
```

You can switch from the Japanese version of NPM to the English version of NPM and back again using the IBMNPMSW command. To switch from Japanese to English, type the following command from the directory where NPM is installed:

```
IBMNPMSW J2E
```

Type the following command to switch from English to Japanese:

```
IBMNPMSW E2J
```

Note: To use the IBMNPMSW command, you must have the NPM library files in the directory on which you unpacked them.

g. Unpack the received files by entering the following command at the OS/2 command prompt:

```
PKUNZIP2 -d -o DCZVEUI.ZIP [targetdir]
```

where:

- `-d` Enables you to extract the subdirectories.
- `-o` Enables you to overwrite an existing file.

Note: If you have customized the default setup profile (IBMNPM.PRF), the response file (IBMNPM.RS), or the default PerfDesk files, you cannot save your existing files. You can copy these customized files to another name to prevent them from being overwritten.

**DCZVEUI.ZIP**

Specifies the product file downloaded from the host.

**targetdir**

Specifies the path where the file is unpacked and from which the NPM Desk/2 installation program will be started.

Note: If you do not specify `targetdir`, the file is unpacked in the current directory. This path cannot be the root.
directory and must be at the first level under the root. For example, \IBMNPM is a valid path.

The downloaded files are arranged on `targetdir` in the following subdirectories:

- `targetdir`: Contains all the NPM Desk/2 files except the IBM–supplied sample PerfDesk files, help files, and the response file.
- `targetdir\HELP`: Contains the help files.
- `targetdir\PERFDESK`: Contains IBM-supplied sample PerfDesks that are shipped with the product. This directory specifies the default working directory for the PerfDesks.
- `targetdir\INSTALL`: Contains the sample response file (SAMPLE.RS), the response file (IBMNP.M.RS), and all the backups of the files changed during the NPM Desk/2 installation procedure. This directory is also the default target directory for the NPM Desk/2 installation program.

h. From the `targetdir`, run the following command to unpack the IBM-supplied sample PerfDesks files and the default profile (IBMNP.M.PRF):

```
DCZVXPLD
```

### Step 7. Identifying NPM Names during Installation

Table 12 is a cross-reference table describing the NPM names you must use during installation of NPM Desk/2 for both OS/2 ES and CM/2 Version 1.x.

<table>
<thead>
<tr>
<th>VTAM</th>
<th>NPM FNMINIT</th>
<th>CM</th>
<th>NPM Desk/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A50NPU1</td>
<td>A50NPU</td>
<td>A50NPU</td>
<td></td>
</tr>
<tr>
<td>NPMA50</td>
<td>NPMA502</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 12 (Page 1 of 2). NPM Names
Define the NPM Desk/2 attended and unattended installation programs.

Note: If you are using the manual installation method, see the NPM Desk/2 User's Guide to continue your NPM Desk/2 installation.

Make sure that the following is true before continuing the NPM Desk/2 installation:

a. (For OS/2 ES Users) Verify that the OS/2 ES Utilities feature is installed on your machine by completing the following steps:
   1) Double-click on the Extended Services folder from your OS/2 Desktop.

Table 12 (Page 2 of 2). NPM Names

<table>
<thead>
<tr>
<th>VTAM</th>
<th>NPM FNMINIT</th>
<th>CM</th>
<th>NPM Desk/2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DCZA50</td>
<td></td>
<td>DCZA50</td>
</tr>
</tbody>
</table>

Note:
1. Specifies the name coded on the NAME keyword of the HOST statement in the FNMINIT member or file.

Use the HOST statement to define the VTAM node in which NPM is executing. The syntax for the HOST statement is:

```
HOST ...... NAME=VTAM | host_name
```

The NAME keyword specifies the name of the host running NPM. If you specify NAME=VTAM, the HOSTPU parameter of the VTAMLST start member is used as the host name. This is also the name you specify in the NPM Settings notebook on the workstation. See the NPM Desk/2 User’s Guide for more information about the NPM Settings notebook.

To specify this name on the workstation, see 2.

2. Specifies the name coded on the APPLID keyword of the VTAM statement in the FNMINIT member or file. The APPLID keyword specifies the application name by which this NPM is known to VTAM. The name that you specify is the Application_Id defined in application definition member of the VTAMLST data set (for example, A50APPLS).

3. Specifies the name coded on the LU62 keyword of the VTAM statement in the FNMINIT member or file. The LU62 keyword specifies the LU 6.2 name by which this component is known to VTAM. The name that you specify is the Application_Id defined in the LU 6.2 application definition member of the VTAMLST data set (for example, APAPLA50).

This is also the name you specify in the Communications Manager configuration for the Partner LU.

The syntax for the VTAM statement is:

```
VTAM ..... APPLID=NPM | application_name,
   LU62 = lu62_name
```

You must specify the same name on the workstation as in the Communications Manager profile. This name represents the partner LU with which NPM Desk/2 communicates through the LU 6.2 session.

4. Specifies the LU name by which the Partner LU is known throughout the SNA network.

This is the name that you have specified for the LU62 keyword in the FNMINIT member or file.

The alias name by which the transaction program within this node can refer to the Partner LU being defined. Use the name that you specified for the LU name portion in the LU name field. Use this name in the NPM Settings notebook.
2) Double-click on the **Add or Remove Features** icon to display the Extended Services Install/Remove Menu.

3) Select **Install Communications Manager**.

4) Select **Additional features** from the Communications Manager Install Menu.

5) Verify that the Utilities selection is marked YES. In case it is marked NO, install the Utilities feature before continuing the NPM Desk/2 installation.

b. (For CM/2 Users) Make sure that the CMRECORD and CMSETUP utilities are not running in any OS/2 session.

The NPM Desk/2 installation program automatically updates the CONFIG.SYS file that resides in the boot drive you specify. If you do not specify the drive, the current boot drive is assumed.

The NPM Desk/2 installation program creates a backup copy of your CONFIG.SYS in the `targetdir\INSTALL` directory. This file is named `CONFIG.xxx`, where `xxx` is an incremental number. The NPM Desk/2 installation program automatically modifies the CONFIG.SYS file for OS/2 2.1 by adding corresponding values to the configuration statements. The installation program also adds the configuration information needed to establish a connection to the NPM host to the Communications Manager’s node definition file.

The NPM Desk/2 installation program uses sections and keywords to represent the input values necessary for installation. The sections and keywords that are used for the Communications Manager configuration definitions are in the NPM Desk/2 installation program’s response file. See the *NPM Desk/2 User’s Guide* for a list of all the installation sections and keywords.

Table 13 contains the statements that are updated and the values that are added to the configuration statements.

<table>
<thead>
<tr>
<th>Configuration Statement</th>
<th>Value Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIBPATH</td>
<td><code>targetdir</code></td>
</tr>
<tr>
<td>SET PATH</td>
<td><code>targetdir</code></td>
</tr>
<tr>
<td>SET DPATH</td>
<td><code>targetdir</code></td>
</tr>
<tr>
<td>SET DPATH</td>
<td><code>targetdir\PERFDESK</code></td>
</tr>
<tr>
<td>SET HELP</td>
<td><code>targetdir\HELP</code></td>
</tr>
</tbody>
</table>

**Note:** The `targetdir` is the fully qualified directory containing the NPM Desk/2 code.

**Step 9.** Generate the NPM Desk/2 setup profile.

The NPM Desk/2 installation program generates the setup profile for the NPM Desk/2 program. The setup profile contains parameters that define how NPM Desk/2 functions on your workstation. You can customize these parameters using the NPM Settings notebook. See the *NPM Desk/2 User’s Guide* for more information about this notebook.
If you are using the attended installation program, you can also modify the setup profile during the installation process. The NPM Desk/2 installation program uses an existing profile file during the installation process. This can be a file that you have previously customized (in an earlier installation or in the NPM Settings notebook), or the sample profile file that is supplied with the NPM Desk/2 installation program.

The attended and unattended installation programs process the setup profile that is supplied as input to the installation programs as follows:

- **Unattended installation**—The unattended installation changes the parameters in the setup profile that are relevant for the installation and that were found to have different values during installation. The installation program uses the working directory and the list of NPM hosts in the response file when processing the setup profile.

  If you have NPM hosts defined in the existing setup profile, it is removed during the installation program if the response file does not contain the appropriate information to configure the host to Communications Manager. This information includes the NPM logical unit name and alias name, and the network ID for the host. You can add this information to the response file using a text editor, or you can use an existing response file that has been obtained as output from a previous attended installation.

- **Attended installation**—In addition to the actions performed in the unattended installation, the attended installation process enables you to edit the contents of the setup profile using the Full Customization window. See step 11 on page 112 for more information about modifying the setup profile.

  **Note:** The attended installation program provides an additional method for defining NPM hosts in the response file. See step 9 on page 110 for more information about defining NPM hosts in the response file.

The NPM Desk/2 installation program can encounter different values for the same parameter from the various input sources. The priority of the source information that is entered during the installation program is as follows:

a. Options for the unattended installation, or user input for the attended installation
b. Response file
c. Existing setup profile
d. Program defaults

Therefore, the program defaults are overwritten by information in the existing setup file, information in the existing setup file is overwritten by information in the response file, and so forth.

The response file keywords are checked for validity as well as their values. If values that are not valid are found, you receive error messages and the NPM Desk/2 installation program ends.

**Step 10.** (Optional) Install NPM Desk/2 using the unattended program.

If you are using the unattended installation program, you do not need to be present at the workstation where NPM Desk/2 is being installed. You can invoke the installation program from a software distribution manager.
(SDM) product. To enable the SDM to dictate when the installation should begin and what should be installed, you must have a distribution agent installed on the workstation. This agent is started when your boot your workstation. In an unattended installation, the error messages are logged and transmitted back to the NPM host and no error messages are displayed on the workstation.

Notes:

a. Be sure you are in the target directory containing the unpacked code.

b. If you receive system error SYS1804, ensure that you have the \\; path coded at the beginning of the LIBPATH statement in your CONFIG.SYS file.

The DCZVICL command starts the installation program. The syntax is as follows:

DCZVICL [Options]

where Options can be one or more of the following:

? Issues a list of all the available options and a brief description of each of them.

/T:target_dir
Identifies the directory where NPM Desk/2 is installed. The directory name must be fully qualified and only one directory level is allowed.

If you do not specify target_dir, DCZVICL looks for the target directory in the response file.

/B:boot_drive
Identifies the drive that contains the CONFIG.SYS file. The boot drive must be a local drive. If you did not specify a boot drive, the current system boot drive is the default.

/R:response_file
Identifies the path and the file name of the installation response file where input data is provided. The response file is an ASCII file that contains keywords and their values. The supported keywords are defined in the NPM Desk/2 User’s Guide (a sample response file is also provided). The supported keywords are defined in the NPM Desk/2 User’s Guide (a sample response file is also provided).

You can modify the keywords values in the response file using a text editor. You can also specify the name of a response file that was created from a previous attended installation.

If you do not specify response_file, the file name IBMNPM.RS is used along with the directory name from which you issued the DCZVICL command. You can customize the supplied SAMPLE.RS file and then rename it to IBMNPM.RS. The installation program uses the first directory level in the path and names the installation subdirectory INSTALL. For example, if the directory from which you started the installation program is C:\IBMNPM, the target directory assumed is C:\IBMNPM\INSTALL\IBMNPM.RS.
NPM Desk/2 Attended Installation

/L:log_file Identifies the name of the NPM Desk/2 logging file. The installation procedure logs the actions performed during the installation process and any errors that might occur. If you do not specify log_file, IBMNPM.LOG is used with the directory name from which you issued the DCZVICL command. The installation program uses the first directory level in the path and names the installation subdirectory INSTALL. For example, if the directory from which you started the installation program is C:\IBMNPM, the target directory assumed is C:\IBMNPM\INSTALL\IBMNPM.LOG.

/I:input_file Identifies the file name of the NPM Desk/2 setup profile. If you do not specify input_file, the file name IBMNPM.PRF is used. The installation program uses the directory from which you issued the DCZVICL command.

/W:working_dir Identifies the directory where the IBM–supplied sample PerfDesks reside. If you do not specify working_dir, the installation program looks for the directory in the response file.

/P:output_file Identifies the file name generated during the NPM Desk/2 installation process. If you do not specify output_file, the file name IBMNPM.PRF is used. The installation program uses the directory from which you issued the DCZVICL command.

Note: The default is the same directory and file name as the input file. If you do not specify a different name, the input file is overwritten.

/S:source_path Specifies the path to the source diskette images. This option indicates where the CM/2 diskette images reside, in case you need to install any additional CM/2 features during the configuration process.

If you type DCZVICL with no options, the system defaults are assumed for all the file names.

To start the installation program, at the OS/2 command prompt, type:

DCZVICL

followed by one or more options, and press Enter.

When the program returns you to the OS/2 command prompt, installation of NPM Desk/2 is complete.

(Optional) Install NPM Desk/2 Using the Attended Installation Program

The attended installation requires an operator to be present to initiate the installation program and to respond to the prompts that are displayed during the installation and configuration processes. The attended installation program enables you to modify the response file and the input file to create a new setup profile (output file), or use an existing setup profile to customize NPM Desk/2 for your workstation.
NPM Desk/2 Attended Installation

Notes:

1. Be sure you are in the target directory containing the unpacked code.

2. If you receive system error SYS1804, ensure that you have the .\; path coded at the beginning of the LIBPATH statement in your CONFIG.SYS file.

Step 1. The INSTALL command allows you to start the attended installation program on a target workstation. To start the installation program, type:

   INSTALL

at the OS/2 command prompt. Click on the OK push button on the IBM logo screen to proceed.

   Note: For the remainder of the installation program windows, the fields in the windows contain the default values that are specified in the response file and input file.

Step 2. Specify the installation profile names. You can change the installation file names that are specified in the response file or the input file. If you do not make any changes in this window, the file names currently specified are used.

   The parameters in this window are the same as those for the unattended installation. See step 10 on page 99 for additional information on these parameters.

The Installation Program window is shown below.

![Installation Program Window](image)

The fields for this window are:

**Existing profile (input)**

Identifies the file name for the NPM Desk/2 setup profile. If you do not specify another name, the default file name IBMNPM.PRF is used.

See the *NPM Desk/2 User’s Guide* for an example of the input file.

**Generated profile (output)**

Identifies the file name generated during the NPM Desk/2 installation process. If you do not specify another name, the default file name IBMNPM.PRF is used.
If the name you specify already exists, you receive a message. If you click on the **OK** push button, the existing file is overwritten. If you click on the **Cancel** push button, the Installation Program Window is redisplayed, enabling you to enter a new output file name.

**Response file**

Identifies the file name of the installation response file where input data is provided. The response file is an ASCII file that contains the keywords and their values. The supported keywords are defined in the *NPM Desk/2 User’s Guide*.

You can modify the keywords values in the response file during installation or by using a text editor. You can also specify the name of a response file that was created from a previous attended installation.

If you do not specify another name, the default file name IBMNPM.RS is used.

**Log file**

Identifies the name of NPM Desk/2’s installation logging file. The installation procedure logs the actions performed during the installation process and any errors that might occur. If you do not specify another name, the default file name IBMNPM.LOG is used.

The push buttons in this window are:

- **Find** Enables you to display the file selection window to find and select the proper file. If you click on this push button, the window shown in the Find a File Name Window is displayed.

  The **Find** push button displays either the **Find a File Name** window or the **Select a Directory** window, depending on the type of information in the field next to the push button. For example, the **Find** push button next to the Existing profile field in the Installation Program Window displays a list of possible file names, while the **Find** push button next to the Target directory field in the Find a File Name window (shown below) displays a list of the possible directories.

- **OK** Enables the installation program to read the input file and the response file. The values of the keywords in these files are verified.

  If you click on the **OK** push button, the installation program proceeds to 5 on page 105.

- **Cancel** Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.

- **Help** Provides help for the fields in this window.

  Click on the **OK** push button to proceed to step 5 on page 105.

**Step 3.** If you click on the **Find** push button next to a field requesting a file name, the following window is displayed.
The fields in this window are:

**Open filename**
- Displays the current file name. The Open filename field is updated when you double-click on a file name listed in the File list box.

**Type of file**
- Displays the file type for the file you want to find.

**Drive**
- Displays your current drive. Click on the arrow to the right of this field to display a list of the drives available to your system.
- When you select a drive from the list, the directories contained in the new drive are entered in the Directory list box.

**File**
- Displays a list of files on a directory in the Directory list box.
- Double-click on a directory name in the list box to display the files for that directory in the File list box.

**Directory**
- Displays the path to your currently selected directory and all its subdirectories. Double-click on a directory in this list box to update the File list box.

The push buttons on this window are:

**OK**
- Enters the new file information on the window from which you invoked the Find a File Name window and closes the window.

**Cancel**
- Closes the dialog without making any changes and returns you to the window from which you invoked the Find a File Name window.

**Help**
- Provides help for the fields on this window.

**Step 4.** Select a directory window. If you click on the Find push button next to a field requesting directory information, the following window is displayed.
Use the **Select a Directory** window to select an existing directory path for the NPM Desk/2 installation program. The fields for this window are:

**Current directory**
- Displays your current directory. This list box is read-only and you cannot enter information. The directory list box is updated when you double-click on a directory listed in the Directories list box.

**Drive**
- Displays your current drive. Click on the arrow to the right of this list box to display a list of the drives available to your system.

When you select a drive from the list, the Current directory list box is updated to reflect the root directory under that drive and the Directories list box is updated to reflect the directories under that drive.

**Directories**
- Displays the path to your currently selected directory and all its subdirectories. Double-click on a directory in this list box to update the Current directory list box. Then click on the **Select** push button to return to the **Installation Information** window.

The push buttons for this window are:

- **Select** Returns you to the window from which you invoked the **Select a Directory** window. The window you are returned to contains the directory changes you made.

- **Cancel** Closes the dialog without making any changes and returns you to the window from which you invoked the **Select a Directory** window.

- **Help** Provides help for the fields in this window.

If you click on the **Select** push button, the installation program returns to the window from which you invoked the **Select a Directory** window.

**Step 5.** Specify the installation directories. The **Installation Information** window enables you to specify the boot drive and the target and working directories for NPM Desk/2. If you do not make any changes in this window, the file names currently specified are used.
The **Installation Information** window is shown in the following figure.

The fields for this window are:

**Boot drive**
Identifies the drive that contains the CONFIG.SYS file. The boot drive must be a local drive. If you do not specify another boot drive, the installation program uses the boot drive in the response file. If there is no boot drive specified in the response file, the installation program defaults to the system boot drive.

**Target directory**
Identifies the directory where NPM Desk/2 is installed. If you do not use the default supplied, the directory name must be fully qualified. Only one directory level is allowed.

**Working directory**
Identifies the directory where the IBM-supplied sample PerfDesk files reside. If you do not specify another name, the default directory PERFDESK is used.

The push buttons on this window are:

**Find**
Enables you to display the **Select a Directory** window to find and select the proper directory. See step 4 on page 104 for more information about the **Select a Directory** window.

**OK**
Validates the input data and proceeds to the next window.

**Cancel**
Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.

**Help**
Provides help for the fields in this window.

After you click on the **OK** push button on the **Installation Information** window, the installation program proceeds to 6.

**Step 6**. Define the Communications Manager changes. Use the windows listed below to define the changes you want to make to your Communications Manager files:

- Communications Parameters
- DLC Parameters
- Define NPM Partners
Step  7. Define the communication parameters. Use Communication Parameters window to specify the communication parameters. These parameters are updated in your Communications Manager configuration file during the NPM Desk/2 installation program.

The Communication Parameters window is shown in the following figure.

```
NPM Desk/2 – Communication Parameters

<table>
<thead>
<tr>
<th>Local Node</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Local PU Name</td>
<td>AFFLP11</td>
<td></td>
</tr>
<tr>
<td>Local PU Net ID</td>
<td>NETA</td>
<td></td>
</tr>
<tr>
<td>Local PU Node ID</td>
<td>FFF11</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Local LU</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Local LU Name</td>
<td>AFFLT11B</td>
<td></td>
</tr>
</tbody>
</table>

Mode Name: FNML6MOD

Configuration File: C:\CMLIB\TKRING.CFG

OK Cancel Help
```

The fields for this window are:

**Local Node**

Specifies the information for the local physical unit (PU). This information is used to update the Communications Manager file.

**Local PU Name**

Specifies the 8-character name of the local physical unit. The PU name must match the host definition for this workstation.

**Local PU Net ID**

Specifies the 8-character ID name of the network to which the local PU (this workstation) belongs.

**Local PU Node ID**

Specifies the hexadecimal network node ID for the local PU.

**Local LU Name**

Specifies the name by which the local logical unit (LU) is known throughout the network. This name is used to update the Communications Manager file.

**Mode name**

Specifies the 8-byte EBCDIC name for the mode. This mode is defined by application programs to establish the network properties for a group of sessions having various partner LUs. The mode name is the name by which the mode is known throughout the network.

The logmode name must be defined in the VTAM logmode table. See “Define Desk/2 Host Configuration” on page 89 for information about logmode definition.
The logmode name must match both the DLOGMOD parameter in the LU statement that defines the local LU to VTAM and the APPL parameter that defines the host NPM LU 6.2 application to VTAM.

**Configuration File**
Indicates the name of the Communications Manager configuration file that is updated by the NPM Desk/2 installation program. To use a different configuration file type a new file name, or click on the Find push button to access the Find a File Name window, where you can locate and select one of the available CM configuration files.

The push buttons for this window are:
- **Find** Enables you to display the Find a File window to find and select the proper file. See step 3 on page 103 for more information about the Find a File window.
- **OK** Validates the input data and proceeds to the next window.
- **Cancel** Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.
- **Help** Provides help for the fields in this window.

If you click on the OK push button, the installation program proceeds to step 8.

**Step 8.** Define the Data Link Control parameters. The data link control (DLC) profile is the SNA layer consisting of link stations that schedule the transfer of data over a link between two nodes, and performs error control for the link. This section contains the changes necessary for defining the connection between the workstation that is running NPM Desk/2 and the target host.

The **DLC Parameters** window is shown in the following figure.

<table>
<thead>
<tr>
<th>NPM Desk/2 - DLC parameters.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enter the following parameters:</strong></td>
</tr>
<tr>
<td><strong>Link Name</strong></td>
</tr>
<tr>
<td><strong>Destination</strong></td>
</tr>
<tr>
<td><strong>Directory Entry</strong></td>
</tr>
<tr>
<td><strong>DLC Type</strong></td>
</tr>
<tr>
<td><strong>Adapter Number</strong></td>
</tr>
</tbody>
</table>

The fields for this window are:
Link name
- Specifies a 1- to 8-character SNA name. The valid characters for this name are A–Z, 0–9, #, $, and @. The first character cannot be @.

APPN link
- Check this box if the link between the workstation and the host is an APPN link. To use an APPN link, your NPM must reside on a host that is defined as an APPN network node. Your workstation will be defined as an APPN end node.

Destination
- Specifies the 12-character, hexadecimal address of the adapter on the partner node to which you are defining this connection. Enter one of the following, depending on your network:
  - The node address or medium access control (MAC) address defined in the configuration for the controller's network adapter.
  - A permanent address that is encoded in the controller's network adapter by the manufacturer.
  - An address assigned within your network to the controller's network adapter.

This field is valid only for the following adapter types:
- Token-ring or other LAN types
- PC Network
- Ethernet (ETHERAND)

Directory entry
- Specifies the name of the X.25 directory entry that you want to use for this connection. If you have not defined an entry for the LU with which you want to communicate, define the entry in the X.25 directory profile. This field is valid only for X25DLC.

DLC type
- Is a 1- to 8-byte ASCII string. This combination box contains the following IBM-supported DLC names:
  - SDLC
  - IBMTRNET
  - ETHERAND
  - IBMPCNET
  - X25DLC
  - TWINAX

Adapter number
- Specifies the adapter number. Because you can have two adapters active at the same time, use the values in the combination box to specify which adapter corresponds to the DLC. If you specify 0, the DLC corresponds to the primary adapter. If you specify 1, the DLC corresponds to the alternate adapter.

The push buttons for this window are:
Find
- Enables you to display the Find a File window to find and select the proper file. See step 3 on page 103 for more information about the Find a File window.
OK Validates the input data and proceeds to the next window.
Cancel Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.
Help Provides help for the fields in this window.

If you click on the OK push button, the installation program proceeds to 9.

Step 9. Define NPM partners. The Define NPM Partners window contains the host names that you defined to NPM Desk/2 in the response file (.RS).

Use the Define NPM Partners window to specify all the hosts that will be available for logging on. Enter the host name, the logical unit definition and a text description, and then press the Add push button for each host you want to add.

Note: When the window is opened, it contains the NPM hosts that are defined in the response file. When the installation process is complete, the hosts you specify in this window are saved in the output profile to be displayed on the list in the NPM Desk/2 Logon to Host NPM window.

The Define NPM Partners window is shown in the following figure.

The fields for this window are:

Host name
This combination box contains the names of the hosts that you specified. When you click on a host name in this list, the partner LU values for that host are displayed in the Partner Information fields.

Use the Delete push button to delete a host from this window. To delete an entry, click with MB1 on the entry in the Host name field and click on the Delete push button.

Note: The Delete push button is disabled if you change any of the values in the Partner Information fields.

Partner information
Specifies the information for the partner LU.
Host name
Specifies the name of the NPM host. The name that NPM Desk/2 uses to identify a host is the name specified on the NAME parameter on the HOST statement in the FNMINIT member or file (for example, HOST NAME=A50NPU). If you specify the host name as HOST NAME=VTAM, the HOSTPU parameter of the VTAMLST start member is used as the host name.

To add a new host name to the list or to add an NPM host with new partner information, enter the information and click on the Add push button. The Add push button is enabled after you enter the partner information.

Note: You can use an NPM host name more than once if you specify unique NPM partners.

Host NPM net ID
Specifies the NPM host network ID. This information is used to update the Communications Manager file.

Host NPM LU name
Specifies the NPM host LU name and must match the LU62 parameter of the VTAM* statement in the FNMINIT member or file. This information is used to update the Communications Manager file.

Host NPM LU alias
Specifies the alias name by which the transaction program within this node can refer to the partner LU. To avoid confusion, set this to the actual LU name. This information is used to update the Communications Manager file.

The values you enter in these fields are set and the entry is added to the window. If an entry with the same LU alias, network ID, or LU host name already exists, you receive an error message.

Description
Enables you to enter a text description of the host. The maximum length for this field is 60 characters.

The push buttons for adding additional NPM hosts are:

Add Enables you to add a host with the values you entered in the Partner information and Description fields. After you change the information in these fields and press the Add push button, the values you assigned are set and a new host name entry is added to the Host name combination box. If you attempt to add a host that already exists, you receive an error message.

The Add push button is disabled until you add a value in one of the Partner information fields. Entering a value in these fields also disables the OK and Delete push buttons.

Click on the Reset push button when you are finished adding host names.
NPM Desk/2 Attended Installation

Reset  Resets the Partner information and Description fields to the values they were for the Host name before you made changes. Selecting Reset enables the OK and Delete push buttons and disables the Add and Reset push buttons.

The push buttons for the Define NPM Partners window are:

OK  Validates the input data and proceeds to the next window.
Cancel  Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.
Help  Provides help for the fields in this window.

If you click on the OK push button, the installation program proceeds to step 10.

Step 10. Customize the setup profile. After you click on the OK push button on the Define NPM Partners window, the Edit All Parameters window is displayed (shown below). This window enables you to choose whether you want to customize the values on the keywords in the setup profile.

If you click on the No push button, the installation program proceeds to step 12 on page 114.
If you click on the Yes push button, the installation proceeds to step 11.

Step 11. Browse or modify parameter settings from the Full Customization window. The Full Customization window enables you to browse and modify all the parameters in the NPM Desk/2 setup profile. The parameters are grouped in the following sections:

- Application
- HostNpm
- FFST
- ConfigurationManager
- DataViewManager

The Full Customization window is shown in Figure 5 on page 113.
Click on the + sign of an icon for an installation section to display the keywords and values for that section. An example of an expanded window is shown in Figure 6.

To display a description of a section or keyword, click on that item with MB1. A multiline field is displayed in the lower right corner of the window containing a description of the section or keyword.

Clicking on an item with MB1 also displays the value for that section or keyword and the Presentation Manager window control in the upper right corner of the window. The window control is the type of field (entry field, spin button and so on) for that keyword. Enter the new value and then click on the Apply push button to change the parameter.

You can click with MB2 to open a context menu for the selected resource. This function is enabled only if the resource you clicked on lets you define...
more than one instance of the same section and keyword. The context menu has two choices:

**Add**
- Inserts another object in the tree. The object is inserted without a value and the object is selected.
- When a section is inserted, the keywords for that section are displayed automatically, but without the keyword values. Add the values to the keywords before proceeding.
- If you click on the OK push button, the NPM Desk/2 installation program verifies that all the added keywords have values assigned. If a keyword without a value is found, you receive an error message.

**Delete**
- Deletes the selected object. NPM Desk/2 does not issue a confirmation message.

See the *NPM Desk/2 User’s Guide* for a list of the sections and the keywords.

The push buttons for this window are:

**OK**
- Enables the installation program to proceed.

**Apply**
- Causes the values specified in the window control box to take effect.

**Reset**
- Resets the Current value field to the value currently in memory for the selected keyword.

**Cancel**
- Interrupts the installation program and closes the dialog. You receive a confirmation message verifying that you want to end the installation program.

**Help**
- Provides help for the fields in this window.

If you click on the OK push button from the Full Customization window, the installation program proceeds to step 12.

**Step 12.** You can save all the input that you entered during the installation process in the response file to reuse the file in another installation. The *Save as New Response File* window is shown below.

![NPM Desk/2 - Save as New Response File](image)

Press 'Yes' to save.
Press 'No' to continue without saving.

The response file field is blank when the window opens. If you enter the name of a response file that already exists, you receive a message asking if you want to continue. If you click on the OK push button of this message, the existing response file is overwritten. If you click on the
Cancel push button on this message, you return to the Save as New Response File window where you can reenter a response file name.

The push buttons on the window are:

**Find** Enables you to display the entries in the Response file field to find and select the proper response file.

**Yes** Starts the file update process.

**No** Continues the installation program without saving a new response file.

**Cancel** Interrupts the installation program. You receive a confirmation message verifying that you want to end the installation program.

**Help** Provides help for the fields in this window.

**Step 13.** Complete the installation process. After you click on the **Yes** push button on the Save as New Response File window, you receive a series of confirmation or error messages as the NPM Desk/2 installation program proceeds. You receive messages for the following steps:

a. The NPM Desk/2 installation program generates the response file. This file contains the keywords and the sections that you specified during the previous installation steps.

b. The CONFIG.SYS file is updated.

c. The installation program ends successfully. Reinitialize the workstation for these changes to take effect.

The following steps are related to OS/2 ES:

- The setup profile is generated.
- The REXX procedure, DCZVCM.CMD, is generated.
- The REXX procedure, DCZVCM.CMD, is started in the Communications Manager directory. At the end of this step, the Configuration Manager profile contains the definitions that are required by the NPM Desk/2 communication component.

The following step is related to CM/2:

- The file DCZV.RSP is created and passed to the CM/2 CMSETUP utility which is automatically started. At the end of this step, the Configuration Manager profile contains the definitions that are required by the NPM Desk/2 communication component.

If the Communications Manager reports an error while verifying the updates that the NPM Desk/2 installation program makes to its configuration, you receive a message notifying you that the installation process completed with errors. This can happen if you supply a configuration file to the NPM Desk/2 installation program that already contains network definitions that conflict with the definitions that the installation program is attempting to add.

If these errors occur, run a verification of the configuration file using the standard Communications Manager Configuration utility to determine what caused the failure. After you identify the problem, make the necessary changes in the network definitions to enable them to coexist in the same configuration.
When you correct the inconsistencies in the configuration file that caused the errors, you do not need to run the installation program again.

After the installation program completes successfully, and you have reinitialized your workstation, an NPM Desk/2 folder is added to the Desktop. This folder contains:

- An icon representing NPM Desk/2 with the starting parameters that were set during the last installation process.
- An icon representing the NPM Desk/2 attended installation.

You can restart the NPM Desk/2 installation program by clicking on the Desktop icon. If you run the NPM Desk/2 installation program more than once, the object definitions in the folder are replaced without any confirmation message.
Chapter 8. Installing NPM Netware Resource Collection

The NetWare resources data NPM NetWare Agent can collect the following types of data:

- Server global
- Server volume
- Router global
- LAN board
- Communications

This data is collected through a data service task (DST) that runs in the NetView program. However, this data is sent to NetView by an NPM NetWare Agent on an unsolicited basis. NetWare data collection requires that this DST and at least one NPM NetWare Agent be running.

If you want to use the NetWare resources data collection function to collect NetWare resources data, you must first install the NPM NetWare Agent and associated code. Activating NetWare resources data collection requires the following procedures:

- Unloading the NPM NetWare Agent and associated code from the delivery tape to the MVS or VM host.
- Downloading the NPM NetWare Agent and associated code to the NetWare servers.
- Customizing the NetWare environment.
- Customizing the host environment.
- Activating the NetWare server to NPM host connection.

Use the following procedure to install the NPM NetWare Agent code:

**Step 1.** Unload the file from the tape to the NPM host.

Use the procedure you normally use to transfer files from a tape to the host. When the unloading is completed, the file will be available to download to the servers.

**Step 2.** Download the NPM NetWare Agent code to the NetWare servers.

The following example shows one way to download the code to the servers; using the TSO file transfer program in conjunction with the RECEIVE command on an OS/2* session. Whatever download program you use, be sure to do a **binary** download (the default for the RECEIVE command).

Type the following command to receive the NPM NetWare Agent code on a diskette:

```
RECEIVE A:\IBMNPMA\G.NLM a:'NPM.V2R4M0.SFMNEUII(DCZVAGEN)'
```

where

- **A:** Specifies the name of the diskette on which you are receiving the code.
- **IBMNPMA\G.NLM** Specifies the name of the code that you are downloading.
Installing NPM Netware Resource Collection

a: Specifies the terminal emulator session you are using for TSO.

‘NPM.V2R4M0.SFNMEUI1(DCZVAGEN)’

Specifies the source location of the NPM NetWare Agent code. ‘NPM.V2R4M0.SFNMEUI1’ is an example of a data set name. The data set name you use depends on the name you used when you received the NPM NetWare Agent code.

Step 3. Download SS.NLM, which is shipped with the NPM code. To download it, follow the same procedure that you used to install the NPM NetWare Agent code, but include the following changes:

- Substitute SS.NLM for all instances of IBMNPMAG.NLM
- Substitute DCZVSS for all instances of DCZVAGEN.

Step 4. Use the following procedure to copy the NPM NetWare Agent NLM (IBMNPMAG) and SS.NLM to the system directory of the SYS volume of the NetWare server:

a. Type the following command from a NetWare requester workstation:

   LOGIN server_name\user_name

   where

   server_name Specifies the name of the server on which you want to install the NPM NetWare Agent code.

   user_name Specifies the name of a user with the required level of authorization. (For example, SUPERVISOR.)

b. Enter the user password when prompted to do so.

c. Enter the following command if you do not have a disk mapped to server_name\SYS:SYSTEM:

   MAP n:=server_name\SYS:SYSTEM

   This command addresses the directory SYSTEM of the SYS volume of the server_name as workstation disk n:

d. Type the following commands to complete the procedure:

   COPY A:IBMNPMAG.NLM n:
   COPY A:SS.NLM n:

   This copies the NPM NetWare Agent code and SS.NLM to the server.

Step 5. Perform the following steps to use the fast path to customize the NetWare environment. (Go to Step 6 if you want to use the detailed procedure.)

a. Identify one or more collection-point file servers for each network that you want to manage as a separate entity. The collection-point file server will be the NPM service point that provides NetWare data to the host.

b. When you configure the prerequisite NetWare software on each server, be sure to identify the network’s collection-point file server that you identified in the previous step.

Step 6. If you did not choose the fast path procedure described in the preceding step, you can use the following detailed procedure to customize the NetWare environment:
a. Ensure that one of the following products is installed on each of the servers on which you want to install the NPM NetWare Agent:
   - NetWare for SAA** (NWSAA)
   - NetWare Management Agent for NetView** (NMA-NV)

b. If you want to group multiple end-point servers under one host session, ensure that the collection-point file server is defined as the collection point for each NPM NetWare Agent server.

   You can choose how many collection-point file servers to have in each network. For example, you can have one for each network or you can configure each server to function as a collection-point file server. For detailed instructions, see the installation and configuration instructions provided by Novell**.

c. Specify the following on each of the servers (including the collection-point file servers):
   - NETVIEW MANAGEMENT=YES
   - GENERATE ALERTS=YES
   - PROCESS COMMANDS=YES
   - NETVIEW SECURITY=YES/NO

Note: If you are using NetWare for SAA, you have the option of coding either NETVIEW SECURITY=YES or NETVIEW SECURITY=NO. If you code NO, security is handled by NetView through operator logon. If you code YES, you need to define the NPM as a valid file-server operator to enable the NetView management security feature. To do this, define a new user named NPM with a security equivalence of supervisor on the server. See the section of the NetWare Utilities Reference manual that explains how to create users for information on how to give the user a security equivalence of supervisor.

   If you are using the NetWare Management Agent for NetView, NetView security is always enabled. (NetWare Management Agent for NetView does not give the user the choice of enabling or disabling NetView security.) You need to define the NPM as a valid file-server operator. To do this, define a new user named NPM with a security equivalence of supervisor on the server. See the section of the NetWare Utilities Reference manual that explains how to create users, for information on how to give the user a security equivalence of supervisor.

   See the installation and configuration instructions provided by Novell for detailed instructions on how to customize the NetWare environment:

d. On the collection-point file server, specify a value of 521 in the Maximum Frame Size to Transmit Inbound field.
e. Before you use the NPM NetWare Agent code, ensure that the following NetWare Loadable Modules** (NLMs) have been loaded (in the following order):

- CLIB.NLM
- STREAMS.NLM
- TLI.NLM
- IPXS.NLM
- SPPS.NLM
Chapter 9. Installing NPM Exits

The following procedures describe how to install VTAM and SMF exits. These procedures also describe how to install the VSAM local shared resource (LSR) option to improve the performance of NPM's VSAM files, and how to use SMP/E and NPM-provided MVS user modifications (USERMODs) to install message tables and VSAM definitions. SMP/E simplifies the installation process by automating the assembly and link-editing of existing and other customized modules.

See Chapter 14, “Customizing NPM Processing with Installation-Wide Exits” on page 237 for information about how to use NPM user exits.

VTAM and SMF Exits

NPM provides exit routines to both VTAM and SMF. Although these exits provide different functions, they are presented together because you can install them at the same time. If you do not need the functions provided by one of these exits, do not install this exit.

For more information about coding a VTAM session manager exit routine, see *NPM Installation and Customization*. For more information about coding an SMF exit routine, see *MVS/Extended Architecture System Programming Library: System Management Facilities*.

The VTAM Session Manager Exit

The VTAM ISTEXCAA session manager exit provides the following functions to NPM:

- Transmits the necessary information to the dynamic network data collection (DNC) function
- Writes session manager records to SMF or to the NPM log (FNMLOGx) each time a session starts or stops

NPM supplies four routines for the VTAM session manager exit, which are described in the following list. You install one or two of these routines, based on the functions you need. NPM also supplies a sample router program that you can use to call each of the exit routines that you specify.

<table>
<thead>
<tr>
<th>Exit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNMCAAX1</td>
<td>Formats a record for each session start and session end and writes that record to SMF. This exit routine is valid only in an MVS environment.</td>
</tr>
<tr>
<td>FNMCAAX2</td>
<td>Formats a record for each session start and session end and writes that record to the NPM log data set. This exit routine writes records only when NPM is active. Although this exit routine is valid in all environments, it is designed to be used in the VM/GCS environment, where SMF is not available. The DNC parameter of the APPL and CONTROL statements and the NCP command does not affect the writing of data by FNMCAAX2.</td>
</tr>
</tbody>
</table>
Installing NPM Exits

**FNMCAAX3** Provides session start and session stop information to NPM’s DNC function. This exit routine provides information only when NPM is active.

**FNMCAAX4** Combines FNMCAAX2 and FNMCAAX3 to write a record to the NPM log data set for each session start and session end, and to provide session start and session stop information to NPM’s DNC function. This exit routine provides information only when NPM is active.

Complete the following steps to select which VTAM session manager exit routines to install:

**Step 1.** Decide whether to use DNC. See *NPM Concepts and Planning* for information about this function.

**Step 2.** Decide whether to use VTAM session manager data collection. See *NPM Concepts and Planning* for information about this function.

**Step 3.** If you plan to use VTAM session manager data collection, choose the destination for the records that are collected. MVS users can write VTAM session manager records either to SMF or to the NPM log data set, VM users can write session manager records to the NPM log data set only.

**Step 4.** Use Figure 7 to determine the routines you need to install, based on the decisions you made in Steps 1 through 3.

*Figure 7. Selecting VTAM Session Manager Exit Routines*
SMF Exit
When SMF is about to write a record to the SMF log, it gives control to one of two SMF exit routines, IEFU83 or IEFU84. You can install an NPM-supplied SMF exit in one of these exit routines to control the writing of SMF records or to collect response time monitor (RTM) data from the NetView program.

NPM supplies two SMF exit routines, which are described in the following list. NPM also supplies two sample router programs, FNMU83 and FNMU84, which you can use to call these routines.

<table>
<thead>
<tr>
<th>Exit Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FNMU83X1</strong> Controls the writing of SMF records. You can control SMF records written by all sources, not just the SMF records written by NPM.</td>
</tr>
<tr>
<td><strong>FNMU83X2</strong> Collects RTM data from the NetView program through SMF. When the NetView program sends RTM data to SMF, NPM intercepts the data using this exit routine and copies it to NPM’s data sets.</td>
</tr>
</tbody>
</table>

Complete the following steps to select the SMF exit routines to install:

**Step 1.** Decide whether to use the SMF control function.
**Step 2.** Decide whether to use RTM data collection.
**Step 3.** Use Figure 8 to determine the routines you need to install, based on the decisions you made in Steps 1 and 2.

![Figure 8. Selecting SMF Exit Routines](image-url)
Sample Router Programs

NPM provides sample router programs, which you can use to call NPM's exit routines for VTAM and SMF. These router programs can call any number of routines. However, you must modify the sample router programs to call only the routines that you want and comment out any routines that you do not want to call. The sample router programs are assembled and link-edited with the selected routines to create load modules that are stored in the product library for VTAM and SMF.

Figure 9 shows an example of the control flow within the two SMF exits. Each time SMF is about to write a record, it gives control to the IEFU83 exit or the IEFU84 exit, depending on how SMF was called. In this example, the IEFU83 and IEFU84 exits are load modules built from NPM's sample router programs and exit routines. SMF calls the exit by name (IEFU83 or IEFU84) and passes control to the router entry point.

Figure 9. Control Flow for SMF Exits IEFU83 and IEFU84

The FNMU83 router program in this example has been modified to call both of NPM's SMF routines and to call one exit routine named YOUREXIT. The FNMU84 router program has been modified to call FNMU83X1 and YOUREXIT. IEFU84 does not call FNMU83X2.
When SMF calls the IEFU83 exit, the FNMU83 router entry point gets control. The router first passes control to the FNMU83X1 routine. NPM uses this routine to control whether the SMF record is written. When FNMU83X1 returns control to the router, the router passes control to the FNMU83X2 routine. NPM uses this routine to intercept the RTM data that is being written to SMF by the NetView program. When FNMU83X2 returns control to the router, the router passes control to the routine YOUREXIT. Finally, when YOUREXIT returns control to the router, the router returns control to SMF.

When SMF calls the IEFU84 exit, the FNMU84 router entry point gets control. The router first passes control to the FNMU83X1 routine. NPM uses this routine to control whether the SMF record is written. When FNMU83X1 returns control to the router, the router passes control to the YOUREXIT routine. When YOUREXIT returns control to the router, the router returns control to SMF.

For additional information about coding a VTAM session manager exit routine, see VTAM Customization. For additional information about coding an SMF exit routine, see MVS/XA System Programming Library: System Management Facilities.

Installing NPM’s VTAM and SMF Exits

NPM provides exit routines to both VTAM and SMF. Although these exits provide different functions, they are presented together because you can install them at the same time. Do not install the functions provided by these exits if you do not need them.

See VTAM Customization for more information about coding a VTAM session management exit routine. See MVS/XA System Programming Library: System Management Facilities for more information about coding an SMF exit routine.

Installing the VTAM and SMF Exits in MVS

You can use any one of four methods to install the VTAM and SMF exits in MVS systems. Your system may or may not already use the ISTEXCAA VTAM exit and the IEFU83 and IEFU84 SMF exits for other programs. You need to know if you have any of these exits installed for programs other than NPM before you can proceed.

If you do not have the exits installed for other programs, you can use one of two methods to install NPM’s exits:

- Use SMP/E with an NPM-supplied USERMOD. See step 1 on page 126 for specific instructions.
- If your exit routine consists of only a single routine, you can rename the routine and copy it into the appropriate library. See step 2 on page 126 for specific instructions.

If you do have existing ISTEXCAA, IEFU83, and IEFU84 exits installed for programs other than NPM, you can use one of two methods to install NPM’s exits:

- Modify the sample router programs supplied with NPM to install both the required NPM routines and your existing exits. See 3 on page 126 for specific instructions.
- Modify your existing exit routines to install the required NPM routines. See step 7 on page 129 for specific instructions.
Installing NPM Exits

Step 1. If you do not have existing ISTEXCAA, IEFU83, or IEFU84 exits installed for products other than NPM, install NPM exits by completing the following steps:

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Follow the steps in 14 on page 134 to install the routines you have selected.

c. IPL MVS with the CLPA option to activate the SMF exits IEFU83 and IEFU84.

d. Start VTAM to activate the VTAM session management exit ISTEXCAA.

Step 2. If you do not have existing ISTEXCAA, IEFU83, or IEFU84 exits installed for products other than NPM and you need only one of NPM’s routines for an exit, install the routine by completing the following steps:

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Copy the required routine from NPM.V2R4M0.SFNMLMD1 to the load module library for VTAM or SMF.

c. Rename the routine to the load module name for the exit. For example, rename FNMU83X1 to IEFU83.

d. IPL MVS with the CLPA option to activate the SMF exits IEFU83 and IEFU84.

e. Restart VTAM to activate the VTAM session management exit ISTEXCAA.

Step 3. If you have existing ISTEXCAA, IEFU83, or IEFU84 exits from programs other than NPM, you can use NPM’s sample router programs to install the required NPM routines and to call your existing exit routines.

For the VTAM session management exit, the sample router program is FNMEXCAA. For the SMF exit, the sample router programs are FNMU83 and FNMU84. The sample router programs are in NPM.V2R4M0.SFNMSRC1. See NPM Concepts and Planning for an explanation of how the router programs work.

To create the ISTEXCAA, IEFU83, and IEFU84 load modules, complete the following steps. Each load module also contains modules that the router program installs. For the new load modules, the entry point is the router program. These steps link-edit the load modules into the correct libraries.

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Copy each sample router program that you plan to use to a user library.

c. Read the comments in the sample router programs.

d. Modify the router program to install the routines you have selected and to call any existing exit routines. The following example shows a modified router program that calls exits FNMU83X1, FNMU83X2, and an existing installation-wide exit (YOREEXIT).
e. Locate the label LIST DS 0A near the end of the sample router program. This is the beginning of the list of routines that the router calls.

f. Comment out the lines that reference routines you do not need.

g. Add definitions for your existing exit routines for products other than NPM.

h. Assemble the router programs. The required MVS macros are in SYS1.MACLIB and SYS1.AMODGEN.

i. Copy the assembled modules to NPM.V2R4M0.AFNMMOD1.

j. Copy the sample job FNMEXITS from NPM.V2R4M0.SFNMJCL1 to a user library.

k. Modify the sample job FNMEXITS to link-edit the modules you have assembled. The following example shows the FNMEXITS job that builds three load modules: IEFU83, IEFU84, and ISTEXCAA.
Installing NPM Exits

```
//FNMEXITS JOB (accounting,information),programmer.name,
// MSGLEVEL=1,MSGCLASS=A,CLASS=A
//FNMEXITS PROC HLQ=NPM,REL=V2R4M/zerodot,OUT='*',UNIT=SYSDA
/** PROCEDURE: FNMEXITS **
/** FUNCTION: INSTALLATION OF NPM EXIT ROUTINES **
/** SYMBOLIC PARMS: **
/** HLQ : DATA SET PREFIX FOR NPM DATA SETS **
/** REL : NPM RELEASE DATA SET QUALIFIER **
/** OUT : SYSOUT CLASS **
/** UNIT : UNITNAME FOR TEMPORARY DATA SETS **
//STEP9D EXEC PGM=IEWL,PARM='RENT,LET,LIST,XREF'
//STEP9D1 EXEC PGM=IEWL,PARM='RENT,LET,LIST,XREF'
//STEP9E EXEC PGM=IEWL,PARM='RENT,LET,LIST,XREF'
//PEND
// EXEC FNMEXITS
//STEP9D.SYSLIN DD * LINKEDIT NPM SMF EXIT ROUTINES
INCLUDE AFNMMOD1(FNMUB3) IEFUB3 EXIT INTERFACE ROUTINE
INCLUDE AFNMMOD1(FNMUB3X1) SMF RECORD CONTROL ROUTINE
INCLUDE AFNMMOD1(FNMUB3X2) POST RTM RECORDS TO NPM
ENTRY FNMUB3
NAME IEFUB3(R)
//STEP9D1.SYSLIN DD * LINKEDIT NPM SMF EXIT RTN.
INCLUDE AFNMMOD1(FNMUB4) IEFUB4 EXIT INTERFACE ROUTINE
INCLUDE AFNMMOD1(FNMUB3X1) SMF RECORD CONTROL ROUTINE
ENTRY FNMUB4
NAME IEFUB4(R)
//STEP9E.SYSLIN DD * LINKEDIT NPM VTAM EXIT ROUTINES
INCLUDE AFNMMOD1(FNMEXCAA) EXIT INTERFACE ROUTINE
INCLUDE AFNMMOD1(FNMCAAX1) ACCOUNTING RECORDS TO SMF
INCLUDE AFNMMOD1(FNMCAAX3) DNC SESSION POST ROUTINE
ENTRY FNMEXCAA
NAME ISTEXCAA(R)
/*
```
I. Modify the JOB statement to meet the requirements of your installation.

m. Delete the link-edit step for any load modules you do not need:
   - STEP9D creates the IEFU83 load module.
   - STEP9D1 creates the IEFU84 load module.
   - STEP9E creates the ISTEXCAA load module.

n. Delete the SYSLIN statements for load modules you do not need:
   - STEP9D.SYSLIN lists the routines to include in the IEFU83 load module.
   - STEP9D1.SYSLIN lists the routines to include in the IEFU84 load module.
   - STEP9E.SYSLIN lists the routines to include in the ISTEXCAA load module.

o. Add INCLUDE statements to add text files for your existing exit routines from products other than NPM.

Step 4. Run the FNMEXITS job to build the load modules and link-edit them into the correct product libraries. The assemble and link-edit should run with return codes of 0. If they do not, correct the errors and rerun the job.

Step 5. IPL MVS with the CLPA option to activate the SMF exits IEFU83 and IEFU84.

Step 6. Start VTAM to activate the VTAM session management exit ISTEXCAA.

Step 7. If you have existing ISTEXCAA, IEFU83, or IEFU84 exits from products other than NPM, you can modify your existing exits to call NPM’s routines by completing the following steps:
   a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.
   b. Read the sample router programs FNMEXCAA, FNMU83, and FNMU84, which are in NPM.V2R4M0.SFNMSRC1.
   c. Update your router programs to call the NPM modules for the VTAM session management and SMF exits, following the steps you used to update the samples.
   d. IPL MVS with the CLPA option to activate the SMF exits IEFU83 and IEFU84.
   e. Start VTAM to activate the VTAM session management exit ISTEXCAA.

Step 8. You can use one of three methods to install the VTAM exit in VM systems. Your system may or may not already use the ISTEXCAA VTAM exit for other programs. You need to know whether you have this exit installed for programs other than NPM before you can proceed.

If you do not have an existing ISTEXCAA exit installed and your exit routine consists of only one routine, you can rename the routine and copy it into the appropriate library. See 9 on page 130 for specific instructions.

If you do have an existing ISTEXCAA exit installed for programs other than NPM, you can use one of two methods to install NPM’s exits:
Installing NPM Exits

- Modify the sample router programs supplied with NPM to install the required NPM routines and to install your existing exits. See 10 on page 130 for specific instructions.
- Modify your existing exit routines to install the required NPM routines. See 11 on page 131 for specific instructions.

**Step 9.** Use the following procedure if you do not have an existing ISTEXCAA exit installed for products other than NPM and you need only one of NPM's routines for this exit:

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Search the NPM MERGE or BASE disk for the routine that you want to install.

c. Rename the routine to the load module name for the exit. For example, rename FNMCAAX2 to ISTEXCAA.

d. Link and access the VTAM RUN disk on which the VTAMUSER LOADLIB is located.

e. Use the LKED VM function to create a load module for your exit. For example, to create the ISTEXCAA LOADLIB use:

```
LKED ISTEXCAA
```

f. Use the LOADLIB copy VM command to put the exit in the VTAMUSER LOADLIB. For example, use:

```
LOADLIB COPY ISTEXCAA LOADLIB A VTAMUSER LOADLIB A
```

**Step 10.** If you have an existing ISTEXCAA exit from programs other than NPM, you can use NPM's sample router program to install the required NPM routines and to call your existing exit routine.

The sample router program is FNMEXCAA FNMSAMP, which is on the NPM RUN disk. See NPM Concepts and Planning for an explanation of how the router program works. NPM also supplies an installation EXEC, FNMUXCAA EXEC, on the RUN disk. FNMUXCAA EXEC assembles and link-edits the session management exit.

Use the following procedure to install the VTAM session management exit for VM:

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Copy FNMEXCAA FNMSAMP on the RUN disk and rename it FNMEXCAA ASSEMBLE.

c. Modify FNMEXCAA ASSEMBLE to install the routines you have selected and to call any existing exit routines.

- Locate the label LIST DS 0A, which is near the end of the sample router program. This is the beginning of the list of routines that the router calls.
- Comment out the lines that reference routines you do not need.
• Add definitions for your existing exit routines for products other than NPM.

d. Update FNMUXCAA LKEDCTRL on the NPM RUN disk to specify the routines that are included when the load module is created. The following example shows a sample of FNMUXCAA LKEDCTRL, which calls exit FNMCAX4, the sample router FNMECAA, and an existing routine (YOUREXIT).

```plaintext
%LIBRARY VTAMUSER
%LEPARMS LIST XREF LET RENT SIZE 256K 32K
  INCLUDE FNMXCAA
  *INCLUDE FNMCAX2
  *INCLUDE FNMCAX3
  INCLUDE FNMCAX4
  *INCLUDE YOUREXIT
ENTRY FNMECAA
NAME ISTEXCAA(R)
```

• Uncomment the INCLUDE statements for the exit routines that you want. Make sure the INCLUDE statements for the exit routines you do not want are commented out.

• Add an INCLUDE statement for your existing exit routines from products other than NPM.

e. Link to the VTAM RUN disk in write mode. This disk contains the VTAMUSER LOADLIB file.

f. Link to the VTAM RUN disk in write mode. This disk contains the VTAMUSER LOADLIB file.

g. Link to the NPM RUN disk in write mode.

h. Link to the NPM BASE disk in read-only mode.

i. Link to the NPM MERGE disk in read-only mode.

j. Link and access the disk that contains the GCS operating system macros in read-only mode.

k. Modify FNMUXCAA EXEC on the NPM RUN disk to add the addresses of the VTAM RUN disk and the NPM RUN disk.

l. Run FNMUXCAA EXEC to assemble and link-edit the module into VTAMUSER LOADLIB. The assemble and link-edit should run with return codes of 0. If they do not, correct the errors listed in FNMEXIT LKEDIT and rerun the EXEC.

m. Start VTAM to activate the VTAM session management exit ISTEXCAA.

**Step 11.** If you have an existing ISTEXCAA exit from a product other than NPM, you can modify your existing exit to call NPM's routines by completing the following steps:

a. Follow the steps shown in Figure 7 on page 122 and Figure 8 on page 123 to select the routines you need to install.

b. Read the sample router program FNMECAA FNMSAMP on the NPM RUN disk.
c. Update your router program to call the NPM modules for theVTAM session management exit following the steps you used to update the sample.

d. Start VTAM to activate the VTAM session management exit ISTEXCAA.

Installing the VSAM Local Shared Resource Option

**Step 12.** You can improve the performance of your session and review data sets by using the VSAM local shared resource (LSR) option. NPM provides the VSAM LSR function as an option for any VSAM file to enable you to manage your own buffers and share VSAM buffers, control blocks, and channel programs with other users. Managing your own buffers is more efficient for direct processing because VSAM searches the buffers for desired records.

You can tell NPM to build a pool of global VSAM buffers for all VSAM data sets. The pool of buffers is defined with the BLDVRP parameter of the BUFFERS initialization statement. If the FILE initialization statement specifies MACRF=LSR, NPM uses this pool of buffers. NPM builds one VSAM buffer pool for all files that specify MACRF=LSR.

**Step 13.** To install the VSAM LSR option for MVS, you can use SMP/E to assemble and link-edit the load module. For information about this method, see 14 on page 134. Use the following procedure if you do not use SMP/E:

a. Use the following procedure to create the FNMBLVRP VSAM buffer module.

1) Copy the source for the FNMBLVRP load module to a user library. NPM supplies a sample source data set to use with the SMP/E USERMOD in NPM.V2R4M0.SFNMSRC1(FNMBLVRP). NPM also supplies sample JCL to define the FNMBLVRP parameter list and to assemble and link-edit the buffer definitions into a user-defined NPM library without using SMP/E. This sample JCL is in NPM.V2R4M0.SFNMJCL1(FNMZVLSR). An example of this JCL is shown below.

```plaintext
//FNMZVLSR JOB MSGLEVEL=1,REGION=2/zerodot/zerodotK
//ASMLK EXEC ASMHCL,
//PARM.L='NCAL,LIST,LET,XREF,REUS,AMODE=24,RMODE=24'
//C.SYSIN DD /c5197
TITLE 'NETVIEW PERFORMANCE MONITOR - BLDVRP PARAMETER LIST'
FNMBLVRP CSECT
BLDVRP KEYLEN=70, REQUIRED C
STRNO=30, REQUIRED C
TYPE=LSR, REQUIRED C
MF=L, REQUIRED C
LOC=ANY, REQUIRED C
SHRPool=0, REQUIRED C
BUFFERS=(24576(2/zerodot))
END
/*
//L.SYSLMOD DD DSN=user.linklib,VOL=SER=volser,UNIT=type,DISP=SHR
/*L.SYSIN DD *
NAME FNMBLVRP(R)
*/
```

**Note:** If you are running MVS/ESA V3 or higher, then this JCL to create a DBLDVRP parameter list must be assembled using Assembler H (EXEC ASMHCL)
The following example shows how this JCL is coded for use with Assembler F.

```
//FNMZVLSR JOB MSGLEVEL=1,REGION=200K
//ASMLK EXEC ASMFCL,
//   PARM.LKED='NCAL,LIST,LET,XREF,REUS,AMODE=24,RMODE=24'
//ASM.SYSIN DD +
//   TITLE 'NETVIEW PERFORMANCE MONITOR - BLDVRP PARAMETER LIST'
FNMBLVRP CSECT
  BLDVRP KEYLEN=70, REQUIRED C
  STRNO=30, REQUIRED C
  TYPE=LSR, REQUIRED C
  MF=L, REQUIRED C
  LOC=ANY, REQUIRED C
  SHRPOOL=/zerodot, REQUIRED C
  BUFFERS=(24576(2/zerodot))
END
```

2) Update FNMBLVRP for use with the USERMOD, or update FNMZVLSR if you do not want to use the USERMOD. Change only the LOC and BUFFERS parameters in the BUILD macro.

3) Update the BUFFERS operand of the BLDVRP macro. Specify a buffer size greater than or equal to the control interval size (CISIZE) of each VSAM data set. You can use buffer sizes of 512, 1024, 2048, 4096, or any multiple of 4096 up to 32768.

When a data set is opened and LSR is defined, VSAM looks for a buffer pool for its index and data components, based on control interval sizes. A buffer pool that is the same size as the control interval size is chosen. Data sets with the same control interval sizes share the same buffer pool. If a buffer pool with the same size has not been defined, then the next higher buffer pool is chosen. If there are no compatible buffer pools defined, the OPEN fails with a VSAM error code of X‘DC’. If you have not defined a resource pool, the OPEN fails with a VSAM error code of X‘E4’.

The buffer sizes used here are examples only. See VSAM Administration: Macro Instruction Reference for your system requirements. The source for FNMBLVRP also contains additional information.

4) Use the SMP/E USERMOD FNMUXA1, or run the FNMZVLSR job to assemble and link-edit the FNMBLVRP load module. See step 14 on page 134 for more information about SMP/E installation.

b. Update the BLDVRP parameter on the BUFFERS initialization statement with the name of the FNMBLVRP module. Code the following on the BUFFERS statement in FNMINIT:

```
BLDVRP=FNMBLVRP
```

c. Update the FILE initialization statements so that the following VSAM data sets specify the MACRF parameter as MACRF=LSR:
Installing NPM Exits

You must specify MACRF=LSR for each VSAM data set that uses the LSR option.

d. Concatenate the user library to the STEPLIB DD statement in the NPM startup JCL. The following example shows how this JCL is coded.

```
//STEPLIB DD DSN=NPM.V2R4M0.SFNMLMD1,DISP=SHR,
//   UNIT=type,VOL=SER=volser
//       DD DSN=user.linklib,DISP=SHR,UNIT=type,VOL=SER=volser
```

Step 14. NPM provides 12 sample user modifications (USERMODs) that assemble and link-edit modules. It also provides a sample job (FNMUMODS) that receives and applies the USERMODs in batch mode. FNMUMODS is shown in Figure 10 on page 135. If you are familiar with SMP/E, you can also receive and apply the USERMODs in interactive mode.

Table 14 lists the modules that are installed by each USERMOD.

<table>
<thead>
<tr>
<th>USERMOD</th>
<th>Function</th>
<th>Modules Installed</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNMUXA1</td>
<td>VSAM LSR</td>
<td>FNMBLVRP</td>
</tr>
<tr>
<td>FNMUXB1</td>
<td>SMF exit</td>
<td>IEFU83 + FNMU83X1</td>
</tr>
<tr>
<td>FNMUXB2</td>
<td>SMF exit</td>
<td>IEFU83 + FNMU83X2</td>
</tr>
<tr>
<td>FNMUXB3</td>
<td>SMF exit</td>
<td>IEFU83 + FNMU83X1 + FNMU83X2</td>
</tr>
<tr>
<td>FNMUXC1</td>
<td>SMF exit</td>
<td>IEFU84 + FNMU83X1</td>
</tr>
<tr>
<td>FNMUXD1</td>
<td>VTAM session manager</td>
<td>FNMXCAA + FNMXCAAX1</td>
</tr>
<tr>
<td>FNMUXD2</td>
<td>VTAM session manager</td>
<td>FNMXCAA + FNMXCAAX2</td>
</tr>
<tr>
<td>FNMUXD3</td>
<td>VTAM session manager</td>
<td>FNMXCAA + FNMXCAAX1 + FNMXCAAX3</td>
</tr>
<tr>
<td>FNMUXD4</td>
<td>VTAM session manager</td>
<td>FNMXCAA + FNMXCAAX4</td>
</tr>
<tr>
<td>FNMUXD5</td>
<td>VTAM session manager</td>
<td>FNMXCAA + FNMXCAAX3</td>
</tr>
<tr>
<td>FNMUXE1</td>
<td>Message table</td>
<td>FNMMGS00</td>
</tr>
<tr>
<td>FNMUXF1</td>
<td>NPM installation-wide exit</td>
<td>FNMUEXIT</td>
</tr>
</tbody>
</table>

Use the following procedure to use these USERMODs to install modules and exits:

a. Read the installation instructions for the exit or module you are installing.

b. Copy the source for the module or exit from the library NPM.V2R4M0.AFNMSRC1 to your user library.

c. Modify the source of FNMBLVRP, FNMMGS00, or FNMUEXIT for your environment if you are using USERMOD FNMUXA1, FNMUXE1, or FNMUXF1.
You do not need to modify the source for the SMF or VTAM session manager exits.

d. Modify the job FNMUMODS by changing FNMUXZZ to the name of the USERMOD. In addition, change the target zone to conform to the NPM zones for NPM Version 2.

e. Run the job FNMUMODS to receive and apply the USERMOD. The USERMOD assembles and link-edits the module or exit.

**Note:** Do not accept the USERMOD. Accepting the USERMOD replaces the source in NPM.V2R4M0.AFNMSRC1.

f. Complete the other installation steps for the exit or module. You must recycle NPM and other products as described in the installation instructions.

```
//FNMUMODS JOB (ACCOUNTING,INFORMATION),PROGRAMMER.NAME,
// MSGLEVEL=1,MSGCLASS=A,CLASS=A
//******************************************************************************
//** Function: This job is used to install NPM usermods for exit routines and tables. *
//** Update the npmtzn parameter to conform to the NPM target zones for NPM Version 2. *
//** Replace the fnmuxzz with the name of the NPM usermod that is to be installed. *
//** The USER.SOURCE data set contains the updated source for the exit or table that is to be installed. *
//******************************************************************************
//SMP EXEC FNMSMPE
//USOURCE DD DSN=USER.SOURCE,DISP=SHR
//SYSIN DD *
SET BDY(npmtzn) . /*
 ** Move the closing comment delimiter from the following RESTORE **
 ** statement to the line before it if you want to APPLY a usermod **
 ** which contains a module previously applied by another usermod. **
 ** Specify the name of the usermod to be removed on the RESTORE **
 ** statement. **
RESTORE S(fnmuxyy) . */
SET BDY(GLOBAL) .
RECEIVE S(fnmuxzz) SYMSMDS LIST.
SET BDY(npmtzn) .
APPLY S(fnmuxzz) REDO . /*
//SMPPTFIN DD DSN=NPM.V2R4M0.SFNJCL1(fnmuxzz),DISP=SHR
```

Figure 10. FNMMODS Job to Receive and Apply USERMODs

**Step 15.** Once you have applied an exit or module using SMP/E, SMP/E keeps track of that exit or module for you. If you want to reinstall that exit or module using a different USERMOD, use the following procedure to restore the original USERMOD using SMP/E:

a. Move the closing comment delimiter from the RESTORE statement to the line before the RESTORE statement. The RESTORE function is contained in the FNMUMODS job, shown in Figure 10.

b. Specify the USERMOD to be removed by changing FNMUXYY to the name of the USERMOD.

c. Run FNMUMODS to remove the existing exit or module.
Installing NPM Exits

After you have completed these steps, you can reinstall the exit or module, using a different USERMOD. Installing Maintenance to Exits, Using SMP/E

Step 16. Use the following procedure to apply maintenance to the exits, message table, and VSAM definitions that were installed using SMP/E:

a. Save any changes you have previously made to the samples.

b. Apply the program temporary fix (PTF) using the bypass option on the APPLY command:

   ```
   APPLY ... BYPASS(ID).
   ```

   This option specifies that SMP/E should ignore any errors it detects when checking the SYSMOD’s RMID and UMIDs. See SMP/E User’s Guide for additional information about PTF installation.

c. Modify the new sample source with your changes.

d. Modify the job FNMUMODS (Figure 10 on page 135) to specify the USERMOD you will use by changing FNMUXZZ to the name of the USERMOD. In addition, change the target zone to conform to the NPM zones for NPM Version 2.

e. Run the job FNMUMODS to receive and reapply the USERMOD. The USERMOD assembles and link-edits the module for the exit.

   **Note:** Do not accept the USERMOD. Accepting the USERMOD replaces the source in NPM.V2R4M0.AFNMSRC1.

f. Reinstall the affected exits as described in the installation instructions.

If you do not follow these steps when applying maintenance, SMP/E issues the following warning messages and stops processing to prevent possible regression of an exit that you might have previously modified:

- **GIM3820E** There is a MODID error for SRC entry source in SYSMOD PTF.
- **GIM31901I** SYSMOD PTF does not specify USERMOD on the PRE or SUP operand. USERMOD is a RMID for an element that is installed.

The following list describes the terms used in these messages:

- **SOURCE** The name of the source contained in the PTF.
- **PTF** Any PTF that ships the source as SRC.
- **USERMOD** The USERMOD that was already installed.
Part 2. Customizing NPM

Chapter 10. NPM Data Collection ............................................. 141
Issuing NPM Commands at Startup (FNMSRT) ............................ 141
Threshold Monitoring .................................................................. 142
Using the CLOCK Parameter ..................................................... 142
Using the OPTION Parameter on the LBRGCOLL, LSEGCOLL, LWGCOLL, NETCOLL, and VxxxxCOLL Commands ........................................... 143
  OPTION=START: .................................................................. 143
  OPTION=STOP: .................................................................... 144
  OPTION=STOPALL: .............................................................. 145
Altering Options with the NPM Desk/2 Interface .......................... 145
Altering Options with NPM Panels .............................................. 145
Using the INTERVAL Parameter .................................................. 145
NetWare Resources, Network, and VTAM Intervals .................... 146
Example FNMSRT Commands ..................................................... 149
  SESSCOLL ....................................................................... 151
  NETCOLL ........................................................................ 152
  EXECUTE ......................................................................... 152
  LANMGR ...................................................................... 152
  LBRGCOLL ...................................................................... 153
  LWGRES ......................................................................... 153
  LWGCOLL ...................................................................... 154
  VADRCOLL ..................................................................... 155
  VMNPCOLL ..................................................................... 155
  VRTPCOLL ..................................................................... 156
  VAPNCOLL ..................................................................... 156
  VCSMCOLL ..................................................................... 156
  VAPLCCOLL .................................................................... 157
  VBPLCCOLL ..................................................................... 157
  VDEVCOLL ..................................................................... 157
  VGBLCCOLL ..................................................................... 158
  VVRCCOLL ..................................................................... 158
Specifying NPM Data Destinations .............................................. 159
Processing NPM EXEC Members (FNMSCMDS) .............................. 160
Running NPM EXECs .................................................................. 162
Examples of NPM EXECs ............................................................ 162
  Example 1 ....................................................................... 162
  Example 2 ....................................................................... 163
Using the System Console .......................................................... 164
  WTOR Interface .................................................................. 164
  MVS MODIFY Command ..................................................... 165
Disabling System Console Support in MVS ................................. 165
Running an NPM EXEC from the System Console ....................... 165
Working with Groups of Logical Units ........................................ 166
  Contents of LU Groups ....................................................... 166
  Processing an LU Group ...................................................... 166
  Example of an LU Group List ................................................. 170
Enhanced LU Groups .................................................................. 171
Defining NPM Desk/2 Resource Groups ...................................... 173

Chapter 11. Starting and Stopping NPM and Verifying Installation ... 175

© Copyright IBM Corp. 1984, 1998
### Chapter 12. Tuning NPM

- Inactivating Session Collection ............................................. 209
- Cutting CSA Usage during Session Collection .............................. 209
- Collecting Session Data by Application .................................. 210
- Tuning VSAM File Processing .................................................. 210
- Optimizing Data Set Organization .......................................... 211
- Using the LSR Option ............................................................ 211
- Using the Performance Measurement Table ................................ 211
- Tuning Initialization Statements ............................................ 211
  - BUFFERS Statement .......................................................... 212
  - CDRM Statement .............................................................. 214
  - CONTROL Statement ......................................................... 214
  - FILE Statement ............................................................... 215
  - HOST Statement ............................................................... 215
  - NPM Statement ............................................................... 215
  - SAMP Statement .............................................................. 217
  - DEFAULTS Statement .......................................................... 217
  - NGRAPHS Statement .......................................................... 217
  - SYS Statement ................................................................. 218
  - TASK Statement ............................................................... 218
- Tuning Performance Data Collection Commands ............................ 218
  - Tuning Network Collection ................................................. 218
    - NETCOLL Command ......................................................... 218
    - Network Monitor Thresholds ............................................ 218
    - Network Collection Intervals .......................................... 219
  - Tuning Session Collection .................................................. 219
  - Tuning VTAM Collection ..................................................... 219
    - VxxxCOLL Commands ....................................................... 219
    - VTAM Monitor Thresholds ............................................... 219
    - VTAM Collection Intervals .............................................. 219
  - Tuning LAN Collection ....................................................... 220
  - Tuning NetWare Resources Data Collection .............................. 220
    - LWGCOLL Command ......................................................... 220
    - NetWare Resources Monitor Thresholds ................................ 220
    - NetWare Resources Collection Intervals .............................. 220
  - Tuning Accounting Collection ............................................. 221
Part 2. Customizing NPM

Chapter 13. Customizing NPM Messages

- Updating the NPM Message Table
  - Defining an NPM Message (FNMMMSG)
  - Message Inserts
  - Defining an NPM Message Replacement String (FNMMRS)
  - Installing the Modified Message Module

- Writing Messages to the System Console
- Writing Messages to the SYSPRINT Data Set
- Sending Messages to an NPM Operator Logged on to NPM
- Sending Messages to an NPM Operator with Global Authority
- Sending Messages to an NPM Operator Not Logged on to NPM
- Sending Alerts to the NetView Program
- Message Table Changes for NPM V2R4

Chapter 14. Customizing NPM Processing with Installation-Wide Exits

- The NPM Installation-Wide Exit
  - Register Contents
  - NPM Installation-Wide Exit Functions
  - Begin Function (Function Code X'FE')
  - End Function (Function Code X'FF')
  - Operator Connect Function (Function Code X'01')
  - Operator Disconnect Function (Function Code X'02')
  - NPALU Connect Function (Function Code X'03')
  - NPALU Disconnect Function (Function Code X'04')
  - NPM Connect Function (Function Code X'05')
  - NPM Disconnect Function (Function Code X'06')
  - DNC Authorization Function (Function Code X'07')
  - File Switch Function (Function Code X'09')
  - Message Function (Function Code X'0A')
  - NPMLOG Record Function (Function Code X'10')
  - VTAMLOG Record Function (Function Code X'11')
  - Session Record Function (Function Code X'12')
  - Review Record Function (Function Code X'13')
  - SMF Record Function (Function Code X'14')
  - Performance Alert Function (Function Code X'15')
  - NetWare Record Function (Function Code X'16')
  - Session Interval Analysis (Function Code X'18')

- Parameter Descriptions
  - Environment Vector List
  - Exit Routine Function Codes
  - User Data Field
  - Exit Options Field
  - Operator User Data Field
  - File User Data Field
  - Session User Data Field
  - Installation-Wide Exit Parameter
  - User Security Area
  - Data Definition Name (DD Name)
  - Message Parameter List
  - Data Set Name
  - WTO Descriptor and Routing Codes (MVS Only)
Volume Serial Field ................................................. 257
SMF Data Areas ....................................................... 258
Installing the NPM Installation-Wide Exit .......................... 258
Installing the NPM Installation-Wide Exit for MVS .............. 259
The PIU Analysis Exit ............................................... 259
Register Contents .................................................... 259
Parameter Descriptions ............................................. 260
   VTAM Transmission Header and Request Header .............. 260
   PIU Data Area .................................................... 260
   Exit Routine Secondary Function Codes ...................... 260
   PIU Analysis Parameter List .................................. 261
Installing a PIU Analysis Exit ..................................... 262
Testing a PIU Analysis Exit ........................................ 262
Chapter 10. NPM Data Collection

This chapter contains information about using NPM commands, and also specifies where to find information about defining LU groups by coding an LU group list.

You can issue NPM commands directly from the system console, or code them in the FNMSTRT sample initialization member or in the NPM EXECs.

- FNMSTRT contains a list of commands that can be executed at NPM startup.
- NPM EXECs contain a list of commands which you can write to perform a variety of tasks. You can execute an NPM EXEC at any time.

The following sections explain how to use FNMSTRT, how to write and process NPM EXECs, and how to issue commands from the system console.

See the NPM User’s Guide for more information about NPM commands.

Issuing NPM Commands at Startup (FNMSTRT)

FNMSTRT is the FNMSTRT member of the FNMPARM partitioned data set.

FNMSTRT contains commands that are read and executed only during NPM initialization. These commands can be grouped in the following way:

**Definition commands**

Define or redefine resources to NPM, or modify parameters of resources already defined.

These commands include LANMGR, GENERIC, NCP, LINE, PU, LU, LUGROUP, NSAMDFY, NGAMDFY, and LWGRES.

**Collection commands**

Start and stop data collection or monitoring, or both.

These commands include LBRGCOLL, LSEGCOLL, LWGCOLL, NGACNTL, NSACNTL, NSASOLCT, NETCOLL, SESSCOLL, VADRCOLL, VAPLCOLL, VAPNCOLL, VBPLCOLL, VCSMCOLL, VDEVCOLL, VGBLCOLL, VMNPCOLL, VRTPCOLL, and VVRCCOLL.

**Other commands**

These commands include EXECUTE and SMFCNTL.

See the NPM User’s Guide for the syntax and descriptions of these commands.

**Note:** If any command in FNMSTRT is incorrect, no command is executed. The NPM EXECs are preprocessed to identify syntax errors. If any of the commands in the EXEC fails the syntax check, the entire EXEC is ignored until the syntax error is corrected. NPM issues a message to alert the user to the syntax error.
Threshold Monitoring

On some NPM commands, you can use keywords that enable you to monitor lower and upper thresholds for specific data. These keywords are not generic and take the name of the data that you want to monitor. The following is an example of the keyword used to specify these thresholds on the VBPLCOLL command:

VBUFTHR=(20,200)

where:

VBUFTHR  Is the keyword used to monitor the degree of VTAM buffer pool thrasing.
(20,200)  Specifies the lower (20) and upper (200) VTAM buffer pool thrashing monitor thresholds.

When you specify VBUFTHR=(20,200), NPM notifies you when the degree of VTAM buffer pool thrashing is below 20 or above 200. If you specify an upper threshold of 0, NPM does not provide monitoring for that threshold. For example, if you code VBUFTHR=(20,0), NPM notifies you when VTAM buffer pool thrashing falls below 20, but does not notify you when VTAM buffer pool thrashing exceeds 20.

There are cases when the minimum value the data can assume is different from 0. The situation does not change for these threshold monitors. Any value specified between 0 and the minimum value (non inclusive) is rejected. A value of 0 is accepted because it means that there should be no monitoring.

To measure specific resources at specific times during a period of several days, you may want to issue commands as part of the initialization process. If NPM is brought up each day or if DAILY=YES is coded as part of the startup commands, then commands with appropriate CLOCK parameters provide daily measurements of specified resources. Issuing commands as part of the initialization process saves the operator time at the terminal.

Operators do not need to be logged on for these commands to be executed. Once the operations are in progress, they can be stopped only by an operator whose security profile specifies CONTROL=YES on the PROFILE statement. See NPM Reference for more information about security profiles and coding the PROFILE statement.

Using the CLOCK Parameter

Most NPM commands that you can issue at startup have an optional CLOCK parameter. This parameter determines when data collection starts and stops. Some commands have a single CLOCK value that determines when the command is issued. Other commands have two CLOCK values: a start time and a stop time. These CLOCK values are the same as the START TIME and STOP TIME fields on the NPM online panels. The CLOCK parameter is optional unless you also specify DAILY=YES for the command.

Note: If you code DAILY=YES, you must code both the start and stop times on the CLOCK parameter, and you must code them with a value greater than 0.
Commands included in the initialization data set are initiated just once each time NPM is activated. Therefore, if a startup command requests data collection from 10:00 a.m. to 12:00 p.m. with DAILY=NO, and NPM is brought up on Monday and remains active for five days, this collection occurs only on Monday. If DAILY=YES, the collection occurs each day during the specified time period.

Times are specified in a **hh.mm.ss** format based on a 24-hour clock.

- **hh** Hours range from 00–24. The time 24.00.00 is converted to 23.59.59 by NPM. The default is 00.
- **mm** Minutes range from 00–59. Minutes are optional. The default is 00.
- **ss** Seconds range from 00–59. Seconds are optional. The default is 00.

The stop time must be at least one base interval greater than the start time, except when both are coded as 00.00.00. The times cannot span midnight. For example, you cannot specify a start time of 23.00.00 and an end time of 01.00.00. If you specify DAILY=YES, it is not possible to code the start time equal to 00.00.00. If you code the start time as 00.00.00, message FNM021E is generated.

If you code a start time of 00.00.00, or do not code a time at all, NPM begins collection immediately when the command is issued. You can also specify that collection occur continuously by coding both the start and stop times as 00.00.00, or by omitting the CLOCK parameter.

If the start time you code is later than the current system time, NPM waits until the start time to process the command. For example, if the current time is 10.15.00 and the start time is 13.30.00, NPM waits until 13.30.00 (1:30 p.m.) to start collecting data. If the start time has already passed, NPM processes the command immediately. For example, if the current time is 10.15.00 and the start time is 08.00.00, NPM starts collection right away, at 10:15 a.m.

Similarly, if the stop time is later than the current system time, NPM waits until the stop time to stop the collection. If the stop time has already passed, NPM stops collection immediately. For network and VTAM collections, NPM logs the data for the completed portion of the current interval when the collection is stopped. For NetWare resources collections, interval and summary records are not written, but monitor resolution notifications are sent if they have been requested.

### Using the OPTION Parameter on the LBRGCOLL, LSEGCOLL, LWGCOLL, NETCOLL, and VxxxCOLL Commands

This section describes the processing associated with the OPTION parameter on the LBRGCOLL, LSEGCOLL, LWGCOLL, NETCOLL, and VxxxCOLL commands, which is used to start and stop the collection of data.

**OPTION=START:**

- When you code **OPTION=START** without a corresponding start or stop time, the command is executed immediately and the command parameters are saved.
• When you code OPTION=START with a corresponding start and stop time, if the start time has passed, the command is executed immediately. The stop is deferred to the entered stop time. If both the start and stop times have passed, the DAILY parameter is checked. If DAILY=YES is coded, the command is deferred until the following day. If DAILY=NO is coded, the command fails. DAILY is only valid if both the start and stop times are specified. If the start time has not passed, the entire command is deferred.

• When you code OPTION=START with only a corresponding start time, the command is deferred until the requested start time. If the start time has passed, the command is executed immediately. When a command is entered with only a stop time, START is executed immediately, and STOP is deferred. If the stop time has passed, the command is rejected.

OPTION=STOP:

• When you code OPTION=STOP without a corresponding start or stop time, the command stops the current collection for the given interval immediately. If there is a deferred stop associated with the active collection, the command also deletes the associated stop (not valid for generic names). If no collection is active, an error message is issued.

• When you code OPTION=STOP with both a corresponding start and stop time, the deferred queue is searched for a corresponding command. If a matching command is found with the same resource name, interval, start, and stop time, it is removed. The command could also be the current active collection, in which case the current collection is stopped.

• When you code OPTION=STOP with only a corresponding start time, and the start time (x) has not passed, the deferred queue is searched for a corresponding command, and the command is deleted (if found). If the start time has passed and there is a current active collection, the current collection is stopped.

• When you code OPTION=STOP with only a corresponding stop time, the command is executed immediately if there is an active collection and the stop time (y) has already passed. If the stop time (y) has not passed, and the RESOURCE is not a GENERIC BUCKET (for example, ALLLINES, PU, ALLVR), the processing logic is as follows:

  IF no collection active, error message;
  ELSE (collection active):
    IF active collection is (0,y) THEN
      Stop the active collection and delete the associated STOP;
    ELSE
      IF active collection is (0,0) or (x,0) THEN
        Stop at (0,y)
      ELSE /* active is (x,y) */
        error
If the stop time \( (y) \) has not passed, and the RESOURCE is a GENERIC BUCKET (for example, ALLLINES, PU, ALLVR), a STOP command is scheduled at time \( (y) \).

**OPTION=STOPALL:**
This command affects only active collections. After you have issued OPTION=STOPALL, all of the active collections in all of the intervals for the chosen resource are stopped, including all collections for the REFRESH interval. This option does not affect deferred collections.

**Note:** OPTION=STOPALL cannot be used with the LBRGCOLL and LSEGCOLL commands.

### Altering Options with the NPM Desk/2 Interface

You can use the NPM Desk/2 interface to modify the options you chose when you coded the LBRGCOLL, LSEGCOLL, LWGCOLL, NETCOLL, and VxxxxCOLL commands. After you select a resource, use the analysis option to get to the Resource details panel. Specify the changes you want to make on this panel and press the Modify push button. See the *NPM Desk/2 User's Guide* for more information on how to use NPM Desk/2 to modify options.

### Altering Options with NPM Panels

You can also use NPM panels to modify the options you chose when you coded the LBRGCOLL, LSEGCOLL, NETCOLL, and VxxxxCOLL commands. To modify the options you chose when you coded the LWGCOLL command, see “Altering Options with the NPM Desk/2 Interface.”

- To alter the options you coded on the VxxxxCOLL and NETCOLL commands, choose the Alter a Collection option on NPM panel FNMO2CMD.
- To alter the options you coded on the LBRGCOLL command, choose the Alter a Collection option on NPM panel FNMO2LBC.
- To alter the options you coded on the LSEGCOLL command, choose option A, Modify a Collection option, on NPM panel FNMO2LSC.

**Note:** You cannot modify options through NPM EXECs or from the system console.

### Using the INTERVAL Parameter

NPM collects data when a specified interval ends. NPM calculates the length of the interval differently for different types of collections.

For LAN segment and LAN bridge collections, NPM calculates the length of the interval, based on the value in minutes you specify on the LBRGINT and LSEGINT parameters on the NPM initialization statement in FNMINIT

where:

- **LBRGINT** Specifies the interval in minutes used for LAN bridge collection.
- **LSEGINT** Specifies the interval in minutes used for LAN segment collection.
For NetWare resources, network, session, and VTAM collections, NPM calculates the length of the interval, based on two factors: the length of the base interval and the interval number.

- You specify the length of the base interval in seconds for NetWare resources, network, session, and VTAM collections, with the INTERVAL parameter on the NPM initialization statement.
- You specify the interval number for NetWare resources and VTAM collections, with the INTERVAL parameter on the LWGCOLL or VxxxCOLL command in FNMSTRT.
- You specify the interval number for session collections, with the SESSINT parameter of the NPM initialization statement in FNMINIT

where:

**INTERVAL (on the NPM statement)**
Specifies the base-interval length, in seconds, used for NetWare resources, network, session, and VTAM collections. The default is 225 seconds.

**INTERVAL (on the LWGCOLL and VxxxCOLL commands)**
Specifies the interval number used for NetWare resources and VTAM collections. The value of the interval number can be any number from 1 to 7. The default value is 1.

**SESSINT**
Specifies the interval number used for session collections.

### NetWare Resources, Network, and VTAM Intervals
NPM uses seven collection intervals to determine when to collect NetWare resources, network, and VTAM data. NPM calculates these intervals, based on the base-interval length, in seconds, and the interval number.

Table 15 shows the interval times resulting from selected base interval lengths and interval numbers. The value for each interval higher than Interval 1 is twice the value of the previous interval.

For example, if you specify a base-interval length of 30 seconds and an interval number of 5, each interval will be 8-minutes long.

For NetWare resources and VTAM collections, you can change the default interval number by coding the LWGIN T and VTAMINT parameters on the DEFAULTS statement in FNMINIT. These parameters are optional.

where:

**LWGINT**
Specifies the default interval number used for NetWare resources collections. The interval number can be any number from 1 to 7. The default value is 1.
VTAMINT

Specifies the default interval number used for VTAM collections. The interval number can be any number from 1 to 7. The default value is 1.

Table 15. Collection Interval Example

<table>
<thead>
<tr>
<th>Base interval</th>
<th>Interval 2</th>
<th>Interval 3</th>
<th>Interval 4</th>
<th>Interval 5</th>
<th>Interval 6</th>
<th>Interval 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 sec</td>
<td>1 min</td>
<td>2 min</td>
<td>4 min</td>
<td>8 min</td>
<td>16 min</td>
<td>32 min</td>
</tr>
<tr>
<td>45 sec</td>
<td>90 sec</td>
<td>3 min</td>
<td>6 min</td>
<td>12 min</td>
<td>24 min</td>
<td>48 min</td>
</tr>
<tr>
<td>1 min</td>
<td>2 min</td>
<td>4 min</td>
<td>8 min</td>
<td>16 min</td>
<td>32 min</td>
<td>64 min</td>
</tr>
<tr>
<td>225 sec</td>
<td>450 sec</td>
<td>15 min</td>
<td>30 min</td>
<td>1 hr</td>
<td>2 hr</td>
<td>4 hr</td>
</tr>
<tr>
<td>5 min</td>
<td>10 min</td>
<td>20 min</td>
<td>40 min</td>
<td>80 min</td>
<td>160 min</td>
<td>320 min</td>
</tr>
<tr>
<td>15 min</td>
<td>30 min</td>
<td>1 hr</td>
<td>2 hr</td>
<td>4 hr</td>
<td>8 hr</td>
<td>16 hr</td>
</tr>
</tbody>
</table>

You can select base interval lengths other than those shown in Table 15. You select one of the seven intervals for NPM to use on the command or panel you use to start network collection.

Session Intervals: Session data collection uses the same base interval as in NetWare resources, network, and VTAM data collections, but multiplies it by a single session interval value to determine the length of the collection interval. For example, if you specify a base interval of 1 minute and a session interval number of 10, session data is collected every 10 minutes. Specify the session interval with the SESSINT parameter of the NPM initialization statement.

LAN Bridge Intervals: LAN bridge data collection uses its own base interval. You specify the interval length in minutes on either the LBRGINT parameter of the NPM initialization statement or on the LAN Bridge Control panel. Collection can occur every 5, 10, 15, 20, 30, or 60 minutes.

LAN Segment Intervals: LAN segment data collection uses its own base interval. You specify the interval length in minutes on either the LSEGINT parameter of the NPM initialization statement or on the LAN Segment Control panel. Collection can occur every 5, 10, 15, 20, 30, or 60 minutes.

Synchronizing Intervals: For network, VTAM, and session collection, NPM starts its intervals when you start NPM. For example, if you start NPM at 7:45 a.m. and specify an interval of 20 minutes, the first interval expires at 8:05 a.m. and the following intervals expire 20 minutes apart. If you want NPM to collect network, VTAM, or session data at specific times of day, specify the SYNCH parameter on the NPM statement.

Note: For VTAM statistics data collection, NPM uses the SYNCH parameter to synchronize VTAM statistics collection interval times, but this parameter is not used to synchronize VTAM statistics sampling intervals.
The format of the SYNCH parameter is as follows:

\[ \text{SYNCH} = (i, m) \]

where:

- \( i \) is the interval number you want to synchronize.
- \( m \) is the number of minutes after the hour.

For example, if you want to collect network data every 30 minutes on the hour and the half hour, use a base interval of 225 seconds. Interval 4 is then 30 minutes. Specifying \( \text{SYNCH} = (4, 30) \) causes interval 4 to occur on the hour and half hour. If you specify interval 4 on your network collection commands, data is collected on the hour and half hour.

For LAN bridge and LAN segment collection, NPM starts intervals on the hour. For example, if you start NPM at 7:45 a.m. and specify an interval of 20 minutes, the first interval starts at 8:00 a.m. and expires at 8:20 a.m.

If you want LAN bridge and LAN segment intervals to start after the hour, specify the interval synchronization on the LBRGINT and LSEGINT parameters of the NPM statement. The interval synchronization value on the LBRGINT and LSEGINT parameters specifies the number of minutes after the hour at which NPM starts the interval. For example, if you specify the interval length as 15 minutes and the synchronization as 5 minutes, then data is collected every 15 minutes, beginning five minutes after the hour, at 11:05, 11:20, 11:35, etc.

Because of system overhead, the time between two collection intervals does not exactly match the interval you specified. NPM makes adjustments to keep collections in synchronization.

**Examples:** The following examples show the effects of the INTERVAL, SYNCH, SESSINT, LBRGINT, and LSEGINT parameters of the NPM initialization statement on the collection of network, VTAM, NetWare resources, session, LAN bridge, and LAN segment data. Each example assumes NPM starts at 7:45 a.m.

- **NPM INTERVAL=(15, MIN), SESSINT=4, SYNCH=(3, 0), LBRGINT=(60, 0)**
  
  NPM network, NetWare resources, and VTAM interval 3 expires on the hour (8:00, 9:00, and so on). The session interval (4x15=1 hour) and LAN bridge interval expire at the same time.

- **NPM INTERVAL=(15, MIN), SESSINT=3, SYNCH=(2, 30), LSEGINT=(15, 0)**
  
  NPM network, NetWare resources, VTAM interval 2 expires on the half hour and the hour (8:00, 8:30, 9:00, and so on). The session interval (3x15=45 minutes) expires every 45 minutes (8:00, 8:45, 9:30, and so on). The LAN segment interval expires every 15 minutes (8:00, 8:15, 8:30, and so on).
NPM network, NetWare resources, VTAM interval 1, the session interval, and the LAN bridge interval expire every 20 minutes starting at 8:09.

NPM network, NetWare resources, VTAM interval 1 (20 minutes), and the session interval (20 minutes) are initially set to expire on the hour. The LAN bridge interval expires every 60 minutes (8:00, 9:00, 10:00, and so on), and the LAN segment interval expires every 20 minutes (8:00, 8:20, 8:40, and so on).

NPM network, NetWare resources, VTAM interval 5 (1 hour), and the LAN bridge intervals expire on the hour (8:00, 9:00, and so on). The session interval would also expire once every hour, but not necessarily on the hour. You must use SESSINT=1 to ensure that the session interval is synchronized to a particular time.

NPM network, NetWare resources, VTAM interval 1, and the session interval expire every 30 seconds, on the minute and minute-and-a-half boundaries. The LAN segment interval expires every 5 minutes (8:00, 8:05, 8:10, and so on).

Example FNMSRT Commands

Figure 11 shows a sample of the start commands and initialization statements that you can code in the FNMSRT data set. This data set is the FNMSRT member of the FNMPARM partitioned data set.

SESSCOLL APPL=NPM,
SESSH=Y,
OPERATOR=(0.0.0.5),
VLOG=1,
HOST=(0.0.0.2),
NET=(0.0.0.2),
BVAL=(0.1.0.3.0.5,1.0)

NETCOLL RESOURCE=(ALLLINES,ALLLUS),
NCP=A3NV4,
OPTION=START,
CLOCK=(13.00.00,16.00.00),
DAILY=YES

EXECUTE MEMBER=FNMEXEC,
PRINTMBR=YES

LANMGR OPTION=ADD,
ADDRESS=1895376548/0.0,
SP=LANNPM9

Figure 11 (Part 1 of 3). Example FNMSRT
LBRGCOLL OPTION=START,
LANMGRNM=LANNPM9,
BC=(5,5,5,5),
NB=(4,4,5,5),
LF=(1,10,1,5),
CLOCK=(8.00.00,16.00.00),
DAILY=YES

LWGRES OPTION=ADD,
SP=NETWARE1,
NETWORK=3BCE5A,
NODEADDR=FFE22A,
AGENT=SERVER0,
RESOURCE=NWAGE

LWGCOLL OPTION=START,
SP=NETWARE1,
AGENT=SERVER0,
CLOCK=(08.00.00,16.00.00),
DAILY=YES,
GROUP=ALL,
DYNAMIC=YES,
SRVUTIL=(20,90),
CONNUTIL=(20,90),
VOLUTIL=(20,90)

VADRCOLL RESOURCE=NETVIEW,
CLOCK=(08.00.00,18.00.00),
DAILY=YES,
DYNAMIC=NO

VAPLCOLL RESOURCE=ALLAPL,
CLOCK=(08.00.00,18.00.00),
DAILY=YES,
DYNAMIC=NO

VBPICOLL RESOURCE=CRPL,
VBUFTHR=(20,200),
CLOCK=(11.00.00,14.00.00),
VXPNRATE=(30,90)

VDEVCOLL RESOURCE=ALLNCP,
CLOCK=(08.00.00,18.00.00),
DAILY=YES,
VDVSLWDN=(10,30),
DYNAMIC=YES

Figure 11 (Part 2 of 3). Example FNMSTRT
The following section explains the commands and statement parameters coded in Figure 11. See the NPM User's Guide for more information about these commands.

**SESSCOLL**

This command starts session collection for any logical unit in session with the application NPM immediately. It stops collection when a stop collect command is issued or NPM ends. It also writes inbound and outbound PIUs to NPM's VTAM log.

The SESSCOLL command contains the following parameters (see Figure 11 on page 149):

- **APPL=NPM** specifies that NPM is the session collection application name.
- **SESSH=Y** specifies that data is compiled and written to the session file.
- **OPERATOR=(0.0,0.5)** specifies that 0.0 is the lower threshold value and 0.5 is the upper threshold value for total or operator transit time data. Threshold values are given in seconds and can range from 0–4620.
- **VLOG=1** specifies to write inbound and outbound PIUs to the VTAM log.
- **HOST=(0.0,0.2)** specifies that 0.0 is the lower threshold value and 0.2 as the upper threshold value for application or host transit time data. Threshold values are given in seconds and can range from 0–4620.
FNMSTRT

- NET=(0.0,0.2) specifies that 0.0 is the lower threshold value and 0.2 is the upper threshold value for network transit time data. Threshold values are given in seconds and can range from 0–4620.
- BVAL=(0.1,0.3,0.5,1.0) specifies that 0.1, 0.3, 0.5, 1.0 are the upper boundary values used to define the distribution groups in the Logical Unit Distributions panel. BVAL can range from 0.01–9999.99 in hundredths of seconds.

NETCOLL
This command starts collecting network performance data for all lines and all LUs defined under NCP A03NV4 each day from 1 p.m. to 4 p.m. The NETCOLL command contains the following parameters (see Figure 11 on page 149):

- RESOURCE=(ALLLINES,ALLLUS) specifies the name of the resource for which you want to start the network collection. The following parameters have been coded for this example:
  - ALLLINES specifies that data is collected for all lines defined to the NCP named in the corresponding NCP=ncp_name parameter.
  - ALLLUS specifies that data is collected for all LUs defined to the NCP named in the corresponding NCP=ncp_name parameter. See the NPM User’s Guide for information about the NETCOLL command.
- NCP=A03NV4 specifies that this NCP resource resides under the name A03NV4.
- OPTION=START specifies to start collection or monitor activity for the named resources.
- CLOCK=(13.00.00,16.00.00) specifies a collection start time of 1:00 p.m. and a collection stop time of 4:00 p.m. These times are specified in the form hh.mm.ss.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

EXECUTE
This command reads and processes a member named FNMEXEC from the FNMSGCMDS data set. The EXECUTE command contains the following parameters (see Figure 11 on page 149):

- MEMBER=FNMEXEC specifies FNMEXEC as the NPM EXEC member to read and execute. The maximum length of this parameter is 8 characters.
- PRINTMBR=YES specifies to print the contents of the FNMEXEC NPM EXEC in the SYSPRINT log.

LANMGR
This parameter defines a LAN Manager to NPM. The LANMGR command contains the following parameters (see Figure 11 on page 149):

- OPTION=ADD specifies to add a LAN Manager.
- ADDRESS=189537654080 specifies the address of the LAN Manager.
- SP=LANNPM9 specifies that LANNPM9 is the PU name of the LAN Manager service point, or the PU or LU name of the LAN Network Manager service point, as defined to VTAM.
**LBRGCOLL**

This command starts and stops collection of LAN bridge data for LAN bridges in your network. Data such as number of broadcast, non-broadcast, and lost frames can be collected. The LBRGCOLL command contains the following parameters (see Figure 11 on page 149):

- **OPTION=START** specifies to start LAN bridge collection on any bridges known to LAN Manager LANNPM9.

- **LANMGRNM=LANNPM9** specifies that the LAN Manager name is LANNPM9. The LAN Manager name should match the name defined by the SP parameter in the LANMGR command.

- **BC=(5,50,5,50)** specifies that 5 is the low threshold value and 50 is the high threshold value for broadcast frames forwarded per minute from both segment 1 and segment 2.

- **NB=(4,40,5,50)** specifies that 4 is the low threshold value and 40 is the high threshold value for non-broadcast frames forwarded per minute from segment 1, and 5 is the low threshold value and 50 is the high threshold value for non-broadcast frames forwarded per minute from segment 2.

- **LF=(1,10,1,5)** specifies that 1 is the low threshold value and 10 is the high threshold value for lost frames forwarded per minute from segment 1, and 1 is the low threshold value and 5 is the high threshold value for lost frames forwarded per minute from segment 2.

- **CLOCK=(8.00.00,16.00.00)** specifies that LAN bridge data collection starts each day at 8 a.m. and stops each day at 4 p.m. These times are specified in the form hh.mm.ss.

- **DAILY=YES** specifies that this collection is started each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

**LWGRES**

This command defines the NetWare resources on which you want to collect data. The LWGRES command contains the following parameters (see Figure 11 on page 149):

- **OPTION=ADD** tells NPM to add the NetWare resources that you specify.

- **SP=NETWARE1** specifies the name of the service point (SP) physical unit (PU) where the selected NetWare resource is defined. In this example, the name is NETWARE1.

- **NETWORK=3BCE5A** specifies the 4-byte address of the Internetwork Packet Exchange (IPX) network. In this example, the address is 3BCE5A.

- **NODEADDR=FFE22A** specifies the 6-byte IPX address of the resource on which you want to collect data.

**Note:** The SP, NETWORK, and NODEADDR combination that you use must be unique for the specified SP for each LWGRES command that you code. That is, you can specify the same NETWORK on several LWGRES commands, but the NODEADDR must be different on each of them.

- **AGENT=SERVER0** specifies the NetWare server that contains the NPM NetWare Agent code for collecting the NetWare resources data.
FNMSTRT

• RESOURCE=NWAGE specifies the type of device to define to NPM. In this example, the device is a NetWare server that does not have the NPM NetWare Agent code installed.

LWGCOLL
This command starts and stops collection of NetWare resources data. The following types of NetWare resources data can be collected from the NPM NetWare Agent:

• Server global
• Server volume
• Router global
• LAN board
• Communications

If the LWGCOLL command fails, NPM will retry the command 10 times, at 30-second intervals, in certain cases where temporary problems cause a failure. For example, NPM will retry the command if it failed because of a temporary shortage of memory in the NPM NetWare Agent.

The LWGCOLL command contains the following parameters (see Figure 11 on page 149):

• OPTION=START tells NPM to start collection on the NetWare resource you have specified. The effect of this command depends on the current time of day and how the CLOCK parameter is coded.

• SP=NETWARE1 specifies the name of the service point (SP) physical unit (PU) where the selected NetWare resource is defined. In this example, the name is NETWARE1.

• AGENT=SERVER0 specifies the NetWare agent on which you want to collect data. You must code a LWGRES command to define this resource to NPM.

• CLOCK=(08.00.00,16.00.00) specifies that NetWare resources data collection starts each day at 8 a.m. and stops each day at 4 p.m. These times are specified in the form hh.mm.ss. You must specify the start and stop times on the clock parameter if you code the DAILY parameter as DAILY=YES.

• DAILY=YES specifies that NetWare resources data collection is started each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

• GROUP=ALL specifies that collection on all the counter groups valid for the NetWare resource you have specified will be started.

• DYNAMIC=YES specifies that collection will be started for a currently defined resource and that a request will be held to start collection for a dynamically reconfigured resource of the specified type.

• SRVUTIL=(20,90) specifies that you will receive monitor exception notification if server utilization falls below 20 % or exceeds 90 %.

• CONNUTIL=(20,90) specifies that you will receive monitor exception notification if connection utilization falls below 20 % or exceeds 90 %.

• VOLUTIL=(20,90) specifies that you will receive monitor exception notification if volume space utilization falls below 20 % or exceeds 90 %.
VADRCOLL
This command starts and stops the collection of MVS address spaces that have active VTAM applications. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM address space data. See the NPM User’s Guide for more information about these VTAM commands.

The VADRCOLL command contains the following parameters (see Figure 11 on page 149):

- RESOURCE=NETVIEW specifies that data is collected for all MVS address spaces named NetView.
- CLOCK=(08.00.00,18.00.00) specifies a collection start time of 8:00 a.m. and a collection stop time of 6:00 p.m. These times are specified in the form hh:mm:ss.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.
- DYNAMIC=NO specifies that collection is not started on VTAM resources that become active in the future.

VMNPCOLL
This command starts and stops the collection of VTAM MNPS application data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM MNPS application data. See the NPM User’s Guide for more information about these VTAM commands.

The VMNPCOLL command contains the following parameters (see Figure 11 on page 149):

- RESOURCE=ALLMNPS specifies that data is collected for all MNPS application data.
- CLOCK=(08.00.00,18.00.00) specifies a collection start time of 8:00 a.m. and a collection stop time of 6:00 p.m. These times are specified in the form hh:mm:ss.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.
- DYNAMIC=NO specifies that collection is not started on VTAM resources that become active in the future.
**VRTPCOLL**

This command starts and stops the collection of VTAM RTP data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM RTP data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VRTPCOLL command contains the following parameters (see Figure 11 on page 149):

- RESOURCE=ALLRTP specifies that data is collected for all active VTAM RTP resources.
- CLOCK=(08.00.00,21.00.00) specifies a collection start time of 8:00 a.m. and a collection stop time of 9:00 p.m. These times are specified in the form *hh.mm.ss*.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.
- DYNAMIC=YES specifies that collection is started on VTAM RTP resources that become active in the future.

**VAPNCOLL**

This command starts and stops the collection of APPN data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting APPN data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VAPNCOLL command contains the following parameters (see Figure 11 on page 149):

- GROUP=TOPOLOGY specifies that data is collected on APPN topology.
- CLOCK=(00.00.00,15.12.00) specifies a collection start time of 12:00 a.m. and a collection stop time of 3:12 p.m. These times are specified in the form *hh.mm.ss*.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

**VCSMCOLL**

This command starts and stops the collection of CSM storage data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting CSM storage data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VCSMCOLL command contains the following parameters (see Figure 11 on page 149):

- GROUP=STORAGE specifies that data is collected on CSM storage usage.
- CLOCK=(00.00.00,15.12.00) specifies a collection start time of 12:00 a.m. and a collection stop time of 3:12 p.m. These times are specified in the form *hh.mm.ss*.
- DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.
VAPLCOLL
This command starts and stops the collection of VTAM application data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM application data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VAPLCOLL command contains the following parameters (see Figure 11 on page 149):

- **RESOURCE=**ALLAPPL specifies that data is collected for all active VTAM applications.
- **CLOCK=(08.00.00,18.00.00) specifies a collection start time of 8:00 a.m. and a collection stop time of 6:00 p.m. These times are specified in the form *hh.mm.ss*.
- **DAILY=YES specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.**
- **DYNAMIC=NO specifies that collection is not started on VTAM applications that become active in the future.**

VBPLCOLL
This command starts and stops the collection and monitoring of VTAM buffer pool data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM buffer pool data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VBPLCOLL command contains the following parameters (see Figure 11 on page 149):

- **RESOURCE=**CRPL specifies the name of the VTAM buffer pool ID on which the expansion rate and degree of thrashing for VTAM buffer pool data is to be monitored.

  **Note:** VTAM buffer pool names are VTAM release dependent.
- **VBUFTHR=(20,200) specifies that the lower and upper threshold values for the degree of VTAM buffer pool thrashing are 20 and 200, respectively.**
- **CLOCK=(11.00.00,14.00.00) specifies a collection start time of 11:00 a.m. and a collection stop time of 2:00 p.m. These times are specified in the form *hh.mm.ss*.**
- **VXPNRATE=(30,90) specifies that the lower and upper threshold values for the VTAM buffer pool expansion rate are 30 and 90, respectively.**

VDEVCOLL
This command starts and stops the collection and monitoring of channel activity data for channel-attached VTAM resources. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting activity data for channel-attached VTAM resources. See the *NPM User’s Guide* for more information about these VTAM commands.

The VDEVCOLL command contains the following parameters (see Figure 11 on page 149):
• **RESOURCE=ALLNCP** specifies that data is collected for all PU type 4 resources.

• **CLOCK=(08.00.00,18.00.00)** specifies a collection start time of 8:00 a.m. and a collection stop time of 6:00 p.m. These times are specified in the form `hh.mm.ss`.

• **DAILY=YES** specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

• **VDVSLWDN=(0,30)** specifies that the lower and upper threshold values for the percentage of time a particular channel is in slowdown during a specified interval are 0 and 30, respectively.

• **DYNAMIC=YES** specifies to start collection on VTAM resources that become active in the future.

### VGBLCOLL
This command starts and stops the collection and monitoring of VTAM global data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM global data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VGBLCOLL command contains the following parameters (see Figure 11 on page 149):

- **RESOURCE=VTAM** specifies that global data is collected for VTAM.

- **VCSAUS=(0,90)** specifies that the lower and upper threshold values for total VTAM CSA usage are 0 and 90, respectively. These values represent the total CSA above and below the line used by VTAM.

- **CLOCK=(11.00.00,14.00.00)** specifies a collection start time of 11:00 a.m. and a collection stop time of 2:00 p.m. These times are specified in the form `hh.mm.ss`.

- **DAILY=YES** specifies to start this collection each day. You must specify the start and stop times on the CLOCK parameter if you code DAILY=YES.

- **VCSA24US=(30,90)** specifies that the lower and upper threshold values for VTAM CSA24 usage are 30 and 90, respectively. CSA24 usage monitors only the explicitly requested 24-bit addressable storage (GETMAIN LOC=BETWEEN) by VTAM. If VTAM does not explicitly request the desired storage location (GETMAIN LOC=ANY), and MVS returns 24-bit addressable storage, the CSA24 usage monitor does not apply.

### VVRCOLL
This command starts and stops the collection and monitoring of VTAM virtual route data. You must code VTAMCOLL=YES on the NPM initialization statement in FNMINIT before you can use this command to start collecting VTAM virtual route data. See the *NPM User’s Guide* for more information about these VTAM commands.

The VVRCOLL command contains the following parameters (see Figure 11 on page 149):

- **DESTSA=4** specifies that VTAM virtual route data is collected for destination subarea 4.
Specifying NPM Data Destinations

You can have NPM write records to a variety of destinations. NPM data that is logged for online analysis should be written to one of the following NPM VSAM files:

- FNMREVx for network, LAN, and VTAM data
- FNMSEx for session data
- FNMLWGx for NetWare resources data

NPM data that is logged for reporting, using NPM’s batch reporting program or any similar tool, should be written to NPM’s sequential log file, FNMLOGx, or to SMF.

**Note:** For SMF records to be written, the SMFPRMxx member of SYS1.PARMLIB must allow type 28 records.

You can specify data destinations on individual collection commands, and you can choose one or more destinations. For example, the following command,

```
VDEVCOLL RESOURCE=ALLCTC,DEST=NPMLOG
```

starts a collection on VTAM channel activity, on all channel-attached hosts, and writes collection records to NPM’s sequential log only.

Similarly, the following command

```
NETCOLL RESOURCE=NCP01,CCUUTIL=(0,70),DEST=(VSAM,SMF),DESTX=SMF
```

starts network collection on an NCP with the name NCP01, and writes records to the VSAM file FNMREVx and to SMF. Exception records are written to SMF only, when the CCU utilization exceeds 70 %.

If you do not specify a destination on the collection command, then NPM uses the active defaults for that command type. Default destinations are defined in the NPM initialization statement of FNMINIT.

**Note:** It is not necessary to define a default data destination for SMF records.

Figure 12 on page 160 shows an example of how to define default data destinations for the various collection types, in FNMINIT:
Processing NPM EXEC Members (FNMSCMDS)

NPM EXECs are members in the FNMSCMDS data set (MVS). You can write EXECs to perform a variety of tasks, including dynamically defining NCPs and LAN Managers.

The FNMSCMDS data set contains members that describe LU groups and members that control SMF record types stored in the SMF log. FNMSCMDS contains the new NPM EXEC members. These members are read and processed when an EXECUTE command is issued either from FNMSTRT, the system console, or online panels. An NPM EXEC can also call another NPM EXEC.

In MVS, to avoid the possibility of members being written in extents that are not known to NPM, you should allocate the FNMSCMDS data set without secondary extents. (NPM issues message FNM261S when members are written in extents that are not known to NPM.)

If the FNMSCMDS data set is allocated with secondary extents and the message FNM261S is issued, the example of JCL shown in Figure 14 on page 161 can be used to compress the data set. When you use this JCL, it is not necessary to recycle NPM, even when NPM is up and running.
The commands in the NPM EXEC members are processed serially, one at a time. For this reason, the order of the statements in FNMSCMDS is important. For example, if a data collection command is processed before the NPM NCP command for the NCP specified on the collection command, the collection command fails. Any error messages resulting from the execution of a statement in an NPM EXEC are written to the SYSPRINT data set along with the statement number of the statement in error.

You can use any of the following commands in an NPM EXEC:

**Definition commands**
Define or redefine resources to NPM, or modify parameters of resources already defined.

These commands include LANMGR, GENERIC, NCP, LINE, PU, LU, LUGROUP, NSAMDFY, NGMDFY, and LWGRES.

**Collection commands**
Start and stop data collection or monitoring, or both.

These commands include LBRGCOLL, LSEGCOLL, LWGCOLL, NGACNTL, NSACNTL, NSASOLCT, NETCOLL, SESSCOLL, VADRCOLL, VAPLCOLL, VAPNCOLL, VBPLCOLL, VDEVCOLL, VCSMCOLL, VGBLCOLL, VMNPCOLL, VRTPCOLL, and VVRCCOLL.

**Other commands**
These commands include EXECUTE, INTERVAL and SMFCNTL.

See the *NPM User’s Guide* for the syntax and descriptions of these commands.
Examples of NPM EXECs

Running NPM EXECs

You can run an NPM EXEC in any of the following ways:

- At initialization
  
  To run an NPM EXEC at initialization, code an EXECUTE command in your FNMSTRT member or file. See the *NPM User’s Guide* for more information.

- Online
  
  To run an NPM EXEC online you need to have CONTROL=YES in your security profile. Select the EXECUTE Function (Option 8) from the NPM Control Functions panel. See the *NPM User’s Guide* for more information about running NPM EXECs from the online panels.

- From the system console

  See *NPM Reference* for more information.

Note: The NPM EXECs are preprocessed to identify syntax errors. If any of the commands in an EXEC fails the syntax check, the entire EXEC is ignored until the syntax error is corrected. NPM issues a message to alert the user to the syntax error.

Examples of NPM EXECs

The following are examples of some simple NPM EXECs.

Example 1

The following EXEC contains two commands, one for defining an NCP to NPM and the other for starting a monitor on all lines defined to the NCP.

```
***----------------------------------------------------------------------------***
*                                                                             *
* MEMBER FNMEXEC                                                              *
*                                                                             *
* FUNCTION: Define NCP01 to NPM                                               *
*    and start monitoring line utilization                                    *
*    on lines defined to the NCP.                                             *
*                                                                             *
***----------------------------------------------------------------------------***
*                                                                             *
NCP NAME=NCP01,ISTATUS=ACTIVE                                                *
*                                                                             *
NETCOLL,NCP=NCP01,INTERVAL=1,LINEUTL=(5,75),RESOURCE=ALLLINES
```
Example 2
The following example contains a central EXEC to call other EXECs that perform different functions.

******************************************************************************************************************************************
* *
* MEMBER XMPEXEC *
*
* FUNCTION: Invokes EXECs NCP1, NCP2, and NCP3. *
*
******************************************************************************************************************************************
*
EXECUTE MEMBER=NCP1
EXECUTE MEMBER=NCP2
EXECUTE MEMBER=NCP3

In this example, NCP1, NCP2, and NCP3 are modular EXECs. NCP1 could contain all the commands related to NCP1, as shown in the following example:

******************************************************************************************************************************************
* *
* MEMBER NCP1 *
*
* Function: Define NCP1 to NPM *
*     Make LINE L02LN010 invalid for network collection *
*     Exclude LU LU02003 from session collection *
*     Enable NSA accounting *
*     Start collection on all lines *
*
******************************************************************************************************************************************
*
NCP NAME=NCP1,ISTATUS=ACTIVE
*
LINE NAME=L02LN010,NPACOLL=NO
*
LU NAME=LU02003,SESSH=X
*
NSACNTL ENABLE,NCP1
*
NETCOLL,NCP=NCP1,INTERVAL=1,RESOURCE=ALLLINES

NCP2 and NCP3 would be similar EXECs. You could invoke XMPEXEC from FNMSTRT to define the three NCPs at initialization. Later, if one of the NCPs (for example, NCP2) is reloading, you could run the NCP2 EXEC from online panels or from the system console to redefine NCP2 to NPM.
Using the System Console

You can also issue the following commands to NPM from the system console, if you have the proper authorization.

**Definition commands**
Define or redefine resources to NPM, or modify parameters of resources already defined.

These commands include LANMGR, NCP, GENERIC, LINE, LU, LUGROUP, LWGRES, PU, NSAMDFY, and NGAMDFY.

**Collection commands**
Start and stop data collection or monitoring, or both.

These commands include LBRGCOLL, LSEGCOLL, LWGCOLL, NGACNTL, NSACNTL, NSASOLCT, NETCOLL, SESSCOLL, VADRCOLL, VAPLCOLL, VAPNCOLL, VBPLCOLL, VDEVCOLL, VCSMCOLL, VGLCOLL, VMNPCOLL, VRPCOLL, and VVRCOLL.

**Console control commands**
Send messages to other NPM users, display information about resources, and manage data files.

These commands include ABEND, DISPLAY, NOTRACE, SEND, SNAP, STOP, TIME, TRACE, and VARY.

**Other commands**
These commands include EXECUTE, INTERVAL, and SMFCNTL.

See the *NPM User's Guide* for the syntax and descriptions of these commands.

There are two ways of using the system console to enter commands. Your system is set up for only one of them:

- WTOR interface
- MODIFY command

**WTOR Interface**

You can use the WTOR interface to pass commands to NPM. This interface is enabled if PROMPT=YES is specified on the CONSOLE initialization statement. The interface issues the message FNM939A to which you can respond with the REPLY command.

```
nn
```

FNM939A hostname REPLY WITH VALID NPM SYSTEM OPERATOR COMMAND

```
├──Reply nn command_text
```

where:

- **nn** Specifies the reply number assigned by the operating system.
- **hostname** Specifies the name of the host.
- **command_text** Specifies the command to issue to NPM.
MVS MODIFY Command

You can use the MVS MODIFY command to pass commands to NPM. This command is enabled if COMMAND=YES is specified on the CONSOLE initialization statement.

```
modify procname,command_text
```

where:

- **procname** Specifies the name of the procedure used to start NPM.
- **command_text** Specifies the command to issue to NPM.

Disabling System Console Support in MVS

You can turn off STOP and MODIFY command processing in MVS by issuing the following command:

```
CONSOLE COMMAND=NO
```

If you issue this command, you must use fast path=9.99 on the online panels to end NPM.

Running an NPM EXEC from the System Console

Use one of the following methods to run an NPM EXEC from the system console, depending on your system configuration:

- **WTOR Interface**
  
  Reply `nn EXECUTE MEMBER=npm_exec_name`

- **MVS MODIFY COMMAND**
  
  `MODIFY fnmsnpn,EXECUTE MEMBER=npm_exec_name`

See the *NPM User's Guide* for more information about running NPM EXECs from the system console or online panels.
LU Groups

Working with Groups of Logical Units

LU groups enable operators to collect and analyze data for a set of secondary LUs. Each LU group contains a user-defined list of LUs in the start session collection (SESSCOLL) command format. The LUs in the group can be unrelated or have any assigned relationship. For example:

- LUs using a particular application
- LUs dedicated to a functional area in the network
- Data entry terminals

In MVS, an LU group is defined as a member of the FNMSCMDS partitioned dataset. The member name becomes the LU group name. In VM, an LU group is defined as a CMS file. The file name is the group name, the file type is FNMSCMDS, and the file mode is an asterisk (*).

LU group names must be unique. Do not name an LU group with the name of any resource defined to NPM. If you change your LU group, you must stop and restart collection on the LU group before your changes are recognized by NPM, unless you are using the enhanced LU grouping function. See “Enhanced LU Groups” on page 171 for details about the benefits of this new support. The sample FNMLUGRP, provided with NPM, contains examples of how to code LU groups (see “Example of an LU Group List” on page 170).

Contents of LU Groups

Each LU list statement in an LU group requires an LU name and can include session data collection parameters.

Note: If the LU group is used for summarizing session data, only the LU name is necessary.

When session collection is started for an LU group, the defaults for the parameters come from the SESSCOLL command. Defaults for the SESSCOLL command are provided by the DEFAULTS and THRESHOLD initialization statements. To override the LU group defaults, specify the default values on an LU list statement with an LU name of $DEFAULT. Any values that you specify for parameters in this way cannot be changed by an operator using NPM’s online panels.

Processing an LU Group

If you want to process individual LUs contained in an LU group and collection has already started on these LUs, you do not need to start session collection for the LU group. If the session collection is in progress for an LU when you enter a SESSCOLL command for the LU group, the collection options listed for the LU in the LU list statement override those for the earlier command.

When you specify an LU group using NPM-to-NPM facilities, the LU group must exist at the remote NPM. The LU names of the group are not transmitted, only the group name is sent.
LU list statements follow the same syntax rules as initialization statements. Processing of the LU group stops if NPM encounters a syntax error. LUs already processed remain in effect. After correcting the syntax, restart the group.

**Note:** The following syntax diagram is only applicable for host NPM LU group definitions, and does not apply to NPM Desk/2 group definitions or processing. See “Defining NPM Desk/2 Resource Groups” on page 173 for an example of how code group definitions for NPM Desk/2.

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

```
LU=lu_name
├── BVAL=(value1,value2,value3,value4)
├── CDRM=nodename
├── GTF=NO
└── HOST=(low_threshold,high_threshold)
```

The following list describes the parameters you can code on the LU list statement:

```
LU=lu_name
```

Specifies the name of the LU included in the LU group. The LU name is required and can contain from 1–8 alphanumeric characters.

If `lu_name` is set to `$DEFAULT`, the values listed for the other operands on this statement become the defaults for the rest of the LU statements in the group.

```
BVAL=(value1,value2,value3,value4)
```

Specifies the upper boundary values that define the distribution groups in the Logical Unit Transit Distributions panel. When an operator transit time is measured, the count is incremented for the distribution group in which it falls.

Specify four boundary values in ascending order and none can equal zero. The fifth boundary value is infinity, for all transit times that are above the fourth distribution boundary. BVAL can range from 0.01–9999.99 in hundredths of seconds.

The defaults are specified by the BVAL parameter on the THRESHOLD initialization statement or on the `LU=$DEFAULT` statement.
Specifies the name of the node to which the LU is attached. The name can contain from 1–8 alphanumeric characters. You must specify the CDRM parameter if the LU is unknown to NPM or if it is not owned by VTAM in the NPM host.

If you specify the CDRM parameter, NPM verifies the LU is attached to the node. If you do not specify the CDRM parameter, but the LU name is currently known to NPM, the node name is the same as the name of the CDRM to which the LU is attached. In either case, the node name is required to start session data collection.

For switched major node resources and locally attached LUs, the NODENAME keyword must be the host PU name. This is because the switched major node resources and the locally attached LUs are owned by VTAM in the NPM host.

Specifies whether the PIUs for this LU should be passed to the general trace facility (GTF).

To use GTF to inspect these PIUs, you must enable this option and GTF must be active.

The default is defined by the GTF parameter on the DEFAULTS initialization statement or on the LU=$DEFAULT statement.

Specify the low and high threshold values for application or host transit time data. This parameter is the same as the host transit threshold parameter specified on the Start Data Collection menu. Data is monitored against these thresholds. If host transit time falls outside either of these values, the statistic is highlighted on the session analysis panels.

Threshold values are given in seconds and can range from 0–9999. If the values are zero, no monitoring occurs. The lower boundary is specified first.

The threshold you specify should not be greater than the value of the MAXHOST parameter on the THRESHOLD initialization statement.

Specifies whether to write abbreviated PIUs to the VTAM log. If MIN=NO, entire PIUs are written.

The default is specified by the MIN parameter in the DEFAULTS initialization statement or in the LU=$DEFAULT statement.
Specifies the low and high threshold values for network transit time data. This parameter is the same as the network transit thresholds parameter specified on the Start Data Collection menu. Data is monitored against these thresholds. If network transit time falls outside either of these values, the statistic is highlighted on the session analysis panels.

Threshold values are given in seconds and can range from 0–9999. If the values are zero, no monitoring occurs. The lower boundary is specified first. The threshold you specify should not be greater than the value of the MAXNET parameter on the THRESHOLD initialization statement.

Specifies the low and high threshold values for total or operator transit data. This parameter is the same as the operator transit threshold parameter specified on the Start Data Collection menu. Data is monitored against the thresholds. If the operator transit time falls outside either of these values, the statistic is highlighted on the session analysis panels.

Threshold values are given in seconds and can range from 0–9999. If the values are zero, no monitoring occurs. The lower boundary is specified first. The specified threshold should not be greater than the value of the MAXOPER parameter of the THRESHOLD initialization statement.

Specifies whether the LU statements in the member are to be listed in the SYSPRINT log data set.

The default for this parameter is the value specified for PRINTMBR on the DEBUG initialization statement. See NPM Diagnosis for more information.

Specifies how session data is written to the primary session file:

- **N** Does not collect session data or write it to the session file.
- **R** Turns on definite response and writes data to the session file.
- **V** Collects only volume data for this LU and writes it to the session file.
- **X** Excludes this LU from session collection if it is in session with an application for which data is being gathered.
- **Y** Collects session data and writes it to the session file.

The default is defined by the SESSH parameter of the DEFAULTS initialization statement or on the LU=$DEFAULT statement.
Specifies whether PIUs should be written to the VTAM log and, if so, what type of logging should occur:

0  No logging.
1  Write inbound and outbound PIUs to the VTAM log.
2  Write only inbound PIUs to the VTAM log.
3  Write inbound and outbound PIUs plus internal trace records to the VTAM log. If internal trace records are to be collected and written to the VTAM log for analysis, the VTAM API trace must be activated before NPM is started. If INTTRACE=YES in the NPM initialization statement, the VTAM internal trace is started. If INTTRACE=NO and VLOG=3, VLOG is reset to 1.

The default is defined on the VLOG parameter on the DEFAULTS initialization statement or on the LU=$DEFAULT statement.

Example of an LU Group List

Figure 15 is an example of an LU group definition. This example uses the default parameter and defines local terminals on the host for data collection. NPM supplies a sample LU group definition in NPM.V2R3M0.SFNMJCL1(FNMLUGRP) for MVS and in the file FNMLUGRP FNMSAMP for VM. The first LU command defines the default values for the LUs in the group.

*  FNMLUGRP
  *  LU=$DEFAULT, SET DEFAULTS FOR REST OF LU STMTS
    SESSH=Y, COLLECT SESSION STATISTICS
    CDRM=NCPSA6, LU ATTACHED TO NCPSA6
    GTF=NO, DO NOT PASS PIUS TO GTF
    VLOG=0, NO LOGGING
    HOST=(0,2), SET_THRESHOLDS FOR APPL OR HOST DATA
    OPER=(0,3), SET_THRESHOLDS FOR TOTAL OR OPER DATA
    NETWORK=(0,3) SET_THRESHOLDS FOR NETWORK TRANS DATA
  *  LU=U0320101 SPECIFY LU U0320101
  *  LU=U0320102 SPECIFY LU U0320102
  *  LU=U0320103 SPECIFY LU U0320103
  *  LU=U0320104 SPECIFY LU U0320104
  *  LU=U0320105 SPECIFY LU U0320105
  *  LU=U0320106, SPECIFY LU U0320106
    HOST=(0,1.5) OVERRIDE UPPER THRESHOLD
  *  LU=LUNPA6, SPECIFY LU LUNPA6
    SESSH=X EXCLUDE STATS FOR THIS LU

Figure 15. Example of an LU Group
Enhanced LU Groups

NPM enables you to define LU groups for which summary records are created. These records contain data about the overall performance of the group, such as the average transit times for all LUs or the total bytes sent by all LUs. You can view these records online from NPM's workstation interfaces (PM of SNA or NPM Desk/2) or in batch using the batch reporting program.

With this enhanced LU groups function you can use a few simple graphs to perform a side-by-side comparison of the service experienced by users at different geographical locations, users in different departments, or users grouped by any other criteria.

These groups must be defined to NPM using the LUGROUP command, which is described under the heading “LUGROUP” in the “NPM Commands” chapter of the NPM User’s Guide.

When a group is defined in this way, summary records are created if there is a session collection started that involves one or more of the LUs in the group. Session collection can be started on the group itself, but the summary records will be created even if collection is started by application or by individual LU.

The enhanced LU grouping function uses the same syntax for group definition, allowing existing LU Group members to be used without change. Some additional parameters have been added, however, to make defining an LU group faster and easier. A masking function enables you to specify a ‘mask’ from which the real LU names will be built. This function reduces the number of definitions required.

For example, the following definition generates 200 LU names, from SM0001 to SM0200, eliminating the necessity to define each one manually:

\[\text{LU}=$\text{MASK}, \
\text{LUMASK}=\text{SM/zerodot/zerodot/zerodot1}, \
\text{LUCOUNT}=200\]

When LU=$MASK is defined, NPM knows that LUMASK and LUCOUNT parameters will follow. LUCOUNT defines the number of LU names to generate, while LUMASK defines the naming scheme to use in the name generation.

You can use the additional keywords on the LUMASK parameter to provide greater flexibility in the name generation. The complete syntax is:

\[\text{LUMASK}=(\text{mask}, \text{radix}, \text{position}, \text{length})\]

where

- **mask** is the first of the LUs in the sequence.
- **radix** defines the name generation option. It can be one of the following:
  - **DEC** The characters that you use to generate the names can range from 0–9.
  - **HEX** The characters that you use to generate the names can range from 0–9 and from A–F.
  - **ALPHA** The characters that you use to generate the names can range from 0–9 and from A–Z.
LU Groups

NAT The characters that you use to generate the names can range from 0–9, from A–Z, plus the characters @, #, and $.

position Is the position in the LUMASK string where the replacement character string is to be generated. Valid values are from 2–7.

length Is the length of the replacement character string. Valid values are from 2–5.

Note that the sum of the values for the position and length keywords cannot exceed nine.

The following is an example of the use of these keywords:

| LU=$MASK, LUMASK=(ROMLU00,ALPHA,6,2), LUCOUNT=72

This statement generates the following LU names:

ROMLU00, ROMLU01, ..., ROMLU09, ROMLU0A, ..., ROMLU0Z,
ROMLU10, ROMLU11, ..., ROMLU19, ROMLU1A, ..., ROMLU1Z

By default, a decimal incrementation is used, up to the number of LU names specified on the LUCOUNT parameter.

If an LU group definition is changed after it has been defined to NPM, you can update the definition by issuing the command again from an EXEC or from the console:

LUGROUP NAME=LONDON

You can use the OPTION parameter on the LUGROUP command to remove an LU group definition that is no longer needed:

LUGROUP NAME=LONDON, OPTION=DELETE

Note that the coded member in the data set is not changed by this command, and the LU group can be redefined at any time by issuing the LUGROUP command.

An LU can be part of only one group at any one time. If NPM detects that a group definition includes an LU that is already included in an active LU group, this LU will not be included in the second group, and a message will be written to the SYSPRINT data set.

You should code a suitable value for the LUGRPNUM parameter on the NPM statement in FNMINIT to optimize the performance of this support. See NPM Reference for a complete explanation of the LUGRPNUM parameter. Do not let this parameter default if you want to use the enhanced LU group support function, as the default is that storage will not be allocated for enhanced LU groups.
Defining NPM Desk/2 Resource Groups

The NPM operator can use NPM Desk/2 resource groups to define a logical group of NPM Desk/2 resources that the NPM Desk/2 operator can then load in a configuration window. See the NPM User’s Guide for more information about NPM Desk/2 group processing. In MVS, a NPM Desk/2 group is defined as a member in the FNMSCMDS partitioned data set. In VM, a NPM Desk/2 group is defined as a CMS file with a file type of FNMSCMDS.

Define a separate statement for each type of resource. Each statement references the resource name. Resource definition statements follow the same rules as initialization statements.

The processing of an NPM Desk/2 resource group stops if NPM encounters a syntax error. Figure 16 shows the syntax for defining NPM Desk/2 resource groups.

```
NCP—NAME=NCP_name
  COMM—NAME=COMM_name
  APPL—NAME=application_name
  APPLYSYN—NAME=application_synonym_name
  BSC—NAME=BS_LINE_name
  SDLC—NAME=SDLC_line_name
  SS—NAME=start/stop_line_name
  NTRPL—NAME=NT physical_link_name
  NTRILL—NAME=NT logical_link_name
  NPSIL—NAME=NPSI_link_name
  NEOL—NAME=NEO_line_name
  XIL—NAME=XI_link_name
  FRAMEPL—NAME=Frame_Relay_physical_link_name
  FRAMELL—NAME=Frame_Relay_logical_link_name
  FRAMESP—NAME=Frame_Relay_station_name
  FRAMELMI—NAME=Frame_LMI_PU_name
  ETHERNET—NAME=Ethernet_physical_link_name
  ODLCPL—NAME=ODLC_LAN_physical_link
  ODLCPU—NAME=ODLC_LAN_station_PU
  CLUSTER—NAME=cluster_name
  PU—NAME=PU_name
  PU4—NAME=PU_type_4_name
  NPSIPU—NAME=NPSI_PU_name
  XIPU—NAME=XI_PU_name
  NPSIVC—NAME=NPSI_virtual_circuit_name
  NEOPU—NAME=NEO_PU
  TERMINAL—NAME=terminal_name
  COMP—NAME=component_name
  LU—NAME=LU_name
  LANBRNM—NAME=LAN_bridge_name
  LANSEG—NAME=LAN_segment_name
  MPCGROUP—NAME=MPC_group_name
  MPCWRITE—NAME=MPC_write_device_name
  MPCREAD—NAME=MPC_read_device_name
  VTAMDS—NAME=VTAM_destination_subarea_name
  VTAMVR—NAME=VTAM_virtual_route_name
  VTAMP—NAME=VTAM_application_name
  VTAMAS—NAME=VTAM_address_space_name
  SUBPUS—NAME=VTAM_subchannel_PUS_name
  SUBPU4—NAME=VTAM_subchannel_PU4_name
  SUBPU2—NAME=VTAM_subchannel_PU2_name
  NNAME—NAME=NAMEware_agent_name
  NWSRV—NAME=NAMEware_server_name
  NWARTR—NETADDR=ROUTER_network_address, NODEADDR=ROUTER_node_address
```

Figure 16. Example of an NPM Desk/2 Group
Defining NPM Desk/2 Groups

Figure 17 shows an example of NPM Desk/2 Definitions

**********************************************************************************************************************
* * Network/Session Resources *
* *
NCP NAME=NCP01
CDRM NAME=CS01
APPL NAME=TSO
SDLC NAME=SDLC01
NEOL NAME=NEOL01
PU NAME=PU01
NPSIPU NAME=NPSIPU01
LU NAME=LU01
*
* LAN Resources *
* *
LANBRNM NAME=BRIDGE01
LANSEGM NAME=SEGM01
*
* VTAM Resources *
* *
VTAMAP NAME=VTAMAP01
**********************************************************************************************************************
* Destination Subarea Name = ddddd + 'DSA'
* where ddddd = Destination Subarea number (dec)
**********************************************************************************************************************
VTAMDS NAME=VTAMDS01
**********************************************************************************************************************
* Virtual Route Name = ddddd + v + t + 'V'
* where ddddd = Destination Subarea number (dec)
* where v  = Virtual Route number (0,7)
* where t  = TP number (0,2)
**********************************************************************************************************************
VTAMVR NAME=VTAMVR01
**********************************************************************************************************************
* Subchannel PU type 2,4,5 and MPC device name = xxxx 'OSC'
* where xxxx = Subchannel address (hex)
**********************************************************************************************************************
SUBPU2 NAME=SUBPU201
MPCREAD NAME=MPCREAD01
MPGROUP NAME=MPGROUP01
*
* NetWare Resources *
* *
NWAGE NAME=NWAGE01
NRTR NETADDR=79F8B5,NODEADDR=37982A
**********************************************************************************************************************
* End of group definition
**********************************************************************************************************************

Figure 17. Example of a Definition of an NPM Desk/2 Resource Group
Chapter 11. Starting and Stopping NPM and Verifying Installation

This chapter describes what you should do before starting NPM, how to start NPM and verify that it is installed correctly, and how to stop NPM and clean up NPM's CSA control blocks when needed. It also describes the contents of the configuration report, which you can use to confirm your installation. See NPM User’s Guide for more information about running NPM as an operator.

Before Starting NPM

Before starting NPM, verify that LIST=YES and CONFIG=YES are coded on the NPM initialization statement in FNMINIT to ensure that a complete FNMILOG is created. Verify that you have coded PRINTMBR=YES (the default) on any EXEC commands in FNMSTRT. Verify the NPM startup JCL to ensure that FNMILOG and SYSPRINT DD cards exist, and to find where this information will be sent. FNMILOG has the word FNMIN000 on the left side of the header of every page and SYSPRINT has the word FNMONL00.

FNMILOG

FNMILOG contains a listing of each NPM initialization deck read at NPM startup (for example, FNMINIT, FNMOPER, and FNMPROF). Each listing is delimited by message FNM284I, each deck is printed in FNMILOG exactly as it is coded, and each statement is numbered. Following the NPM initialization statement listing is an error message listing, with each error prefixed by the NPM statement number identifying which statement caused the error. Following the NPM statement error section is another listing of all the NPM initialization statements showing them after they have been processed by NPM.

Following the processed statements is the virtual storage allocation table listing showing the virtual storage allocated for your system. You may also want to review the performance measurement tables for more detailed information, which you can access by using option 9.4 from the NPM Primary Options (FNM00PRI) panel. The NPM Configuration report follows the virtual storage allocation tables. The NPM Configuration report lists every resource defined to NPM by the NPM initialization statements or the NCP commands obtained from NCP RRTs. The last part of FNMILOG is the Totals By Resource Type table.

Note: If you have a large system, it is recommended that you suppress the Configuration report by coding CONFIG=NO (the default) on the NPM initialization statement. The Configuration report consumes a large amount of space and should only be generated during migration or for NPM problem determination.

SYSPRINT

SYSPRINT contains NPM messages generated during the running of NPM, starting with any messages produced as member FNMSTRT is parsed. SYSPRINT also prefixes the EXEC command number to the error message. This allows system programmers to match the error message to the corresponding EXEC command. Messages which do not have a corresponding statement number will have either a '@' or '*' prefix only.
The following message is the most important SYSPRINT message to verify because it indicates that NPM started without any errors:

FNM025I 'NPM Initialization Completed'

If the following message is generated, you need to examine either FNMILOG or the preceding section of SYSPRINT to determine what the error is and how to correct it:

FNM045W 'NPM Initialization completed with Errors'

If you are executing an EXEC using option 9.8 on the NPM online panels, you should set the LIST NPM EXEC field to YES. This option causes NPM to print the contents of the EXEC in SYSPRINT. This listing is very helpful when trying to determine the cause of an EXEC-generated error message. Each command is numbered, and each error message is prefixed with the number.

See “Reviewing Your NPM Configuration” on page 181 for more information about using FNMILOG and SYSPRINT at NPM startup. See Chapter 4, “Installing NPM Manually” on page 27 for more information about coding the NPM initialization statements. See NPM Reference for examples of NPM initialization statement syntax coding, and NPM Diagnosis for a detailed explanation of how to use FNMILOG and SYSPRINT to diagnose NPM startup problems.

---

**Starting NPM**

Once you have completed the installation steps described in the previous chapters, you are ready to start NPM and verify its installation. The following sections explain how to start NPM.

Enter the following START command to start NPM:

```
$ procname
```

where:

*procname* Is the procedure name in the NPM startup JCL.

If you used the NPM initialization program to install NPM, you named a procedure that was created to start NPM. Use the name you specified while running the NPM initialization program as *procname* (the default is FNMSNPM).

NPM supplies sample JCL to start NPM in NPM.V2R4M0.SFNMJCL1(FNMSNPM). An example of the JCL needed to start NPM under MVS is shown in Figure 18 on page 177. Following the figure are descriptions of the numbered lines.
Figure 18. JCL for Starting NPM under MVS

The components of the JCL in Figure 18 are described in the following sections.

1 EXEC STATEMENT

You specify control information for NPM execution through a combination of parameters on the EXEC statement and control statements in the NPM initialization data set. See the FNMPARM parameter on page 178. Place the parameters that define the member names to be read for initialization statements, operator definitions, operator profiles, and initialization commands in the EXEC statement.

The specified program name must be FNMMAIN.

Note: Include FNMMAIN in the program properties table before starting NPM. See “Step 2. Define NPM for MVS” on page 29 for instructions.

The following list describes the PARM field parameters on the EXEC statement:
INIT Specifies the names of the members of FNMPARM that contain the initialization statements. You can specify two members. You can abbreviate this parameter as I.

OPER Specifies the name of the member of FNMPARM that contains the operator definition statements. You can abbreviate this parameter as O.

PROF Specifies the name of the member of FNMPARM that contains the operator profile statements. You can abbreviate this parameter as PR.

CMDS Specifies the name of the member of FNMPARM that contains the initialization startup commands. You can abbreviate this parameter as CM.

LINECNT Specifies the number of lines per page written to the NPM members of FNMILOG and SYSPRINT. You can set this parameter to any number from 20–200. The default is 55.

PD Specifies the name of the member of FNMPARM that contains certain debugging options. See NPM Diagnosis for more information.

Note: See “Cutting CSA Usage during Session Collection” on page 209 for more information about how you can set SESSNO and BUFFNO to reduce CSA usage.

2 STEPLIB

Identifies the load library containing the modules link-edited by the installation procedure. The library must be APF authorized.

3 FNMLLIB

Identifies an optional data set pointing to the load library containing the NCP RRTs for the NCPs defined in this NPM. The library must be APF-authorized. It can be the same data set named in the STEPLIB DD statement. If you omit the FNMLLIB DD statement, the NCP RRTs must be in the STEPLIB data set.

4 SYSPRINT

Identifies the data set used as a record of initialization commands and messages encountered during the execution of NPM.

5 FNMPARM

Identifies the data set containing the initialization parameters for NPM. This data set is processed as a partitioned data set. The following members are processed at initialization:

FNMINIT Contains the NPM initialization statements.

FNMOPER Contains the definitions for NPM operators.

FNMPROF Contains the definitions for NPM profiles used to describe security controls for an NPM operator.

FNMSSTRT Contains the NPM initialization commands.

Note: The actual member names are specified in the EXEC statement.
6 **FNMILOG**

Defines a SYSOUT data set that contains a listing of the initialization statements and messages.

7 **FNMSPNL0**

Defines the English national language panels. You can define other panel data sets for other national language features. See the documentation supplied with the national language feature for information about defining panel data sets.

8 **FNMPROFS**

Defines an optional partitioned data set that saves operator variables when an NPM operator is not logged on to NPM. A member is created using the operator ID in the FNMPROFS data set that contains the variables set by the operator during the last session. When the operator logs on again, the variables are read back into memory for use at the operator terminal.

9–10 **FNMRREV1/FNMRREV2**

Defines the VSAM KSDSs for network and LAN data. These DD statements are named in the FILE initialization statements. You must have previously defined and initialized these files with a record of binary zeros. The DD name of the data sets must be in the form FNMRREVx, where x is a unique character you assign. Valid characters are A–Z, 0–9, and the at (@), pound (#), and dollar ($) symbols.

11–12 **FNMSSES1/FNMSSES2**

Define VSAM KSDSs for session data. These DD statements are named in the FILE initialization statements. You must have previously defined and initialized these files with a record of binary zeros. The DD name of the data sets must be in the form FNMSSESx, where x is a unique character you assign. Valid characters are A–Z, 0–9, and the at (@), pound (#), and dollar ($) symbols.

13–14 **FNMLWG1/FNMLWG2**

Defines the VSAM KSDSs for NetWare resources collection data. These DD statements are named in the FILE initialization statements. You must have previously defined and initialized these files with a record of binary zeros. The DD name of the data sets must be in the form FNMLWGx, where x is any unique alphanumeric or national character (A–Z, 0–9, #, @, $) you assign.

15 **FNMLCOLL**

Identifies the sequential data set that contains the NetWare resources data collections that were active when NPM terminated or ended abnormally. NPM uses this information to stop DataView collections and to restart as many as possible of the logging collections that were active when NPM terminated. For more information, see the section of the *NPM Diagnosis* manual that describes checkpoint processing.
**16-17 FNLOG1/FNLOG2**

Record NPM data from network, session, or LAN data collections. If log records are being written to SMF, these statements can be omitted. You can define multiple FNLOGx data sets with the FILE initialization statement. These data sets are sequential data sets. The DD name of the data sets must be in the form FNMLOGx, where x is any unique alphanumeric or national character you assigned.

**18 FNMSCMDS**

Identifies a partitioned data set with members containing NPM EXECs and LU lists.

In MVS, to avoid the possibility of members being written in extents that are not known to NPM, you should allocate the FNMSCMDS data set without secondary extents. (NPM issues message FNM261S when members are written in extents that are not known to NPM.)

Use the NPM EXECs to modify configuration information with NPM NCP, LANMGR, LINE, LU, PU, and GENERIC commands, as well as collection commands. Use the LU list members to start or stop session data collection or to summarize session data on the LU names contained in the LU group. See “Working with Groups of Logical Units” on page 166 for more information.

**19–20 FNVLOG1/FN MVLOG2**

Define the VTAM log data sets that contain PIU data logs. The VTAM log is a variable blocked data set that applies to the NPM session subsystem. You can define multiple VTAM log data sets with the FILE initialization statement. The DD name of the data sets must be in the form FNMVLOGx, where x is any unique alphanumeric or national character you assigned.

**21 VTAMLST**

Identifies the data set containing the NCP VTAMLST members used to define NCPs to the network. NPM also uses the members in this data set to define the NCP to NPM.

**22 SYSMDUMP**

Identifies a dump data set that has the same characteristics as the MVS system dump data sets (for example, SYS1.DUMP00). During normal operation, NPM obtains and uses storage in CSA. If NPM abnormally ends, the dump (including CSA storage) is written to the data set described by the SYSMDUMP DD name. Do not allocate extents for this data set. It must occupy a single contiguous space.

See *NPM Diagnosis* for information about using this data set to supply APAR diagnostic information.

**23 FNMSNAP**

Defines a data set containing snap dumps created by NPM. These dumps are created to help debug NPM.

**24 FNMSDUMP**
Identifies a dump data set that has the same characteristics as the MVS system dump data sets (for example, SYS1.DUMP00). This dump data set is used when an SDUMP is requested by an NPM option. Do not allocate extents for this data set. It must occupy a single contiguous space.

Verifying NPM

After you have started NPM, take the following steps to verify that NPM has been installed correctly:

- Review information about your NPM configuration
- Verify the installation of NPM functions

The following sections explain these steps in more detail.

Reviewing Your NPM Configuration

You can find information about your NPM configuration in the SYSPRINT and FNMILOG data sets. Read these data sets to ensure you have properly defined your configuration.

The SYSPRINT data set contains a record of initialization commands and messages encountered during the execution of NPM. The FNMILOG data set provides a listing of your initialization statements, with the parameters you defined, and defaults used for the parameters you did not specify. It also lists error messages and a summary of storage used by NPM.

The NPM configuration report is written to the FNMILOG data set when you specify CONFIG=YES on the NPM initialization statement in the FNMINIT data set.

Reading the NPM Configuration Report

The NPM configuration report summarizes the configuration established during NPM initialization. Use this report to check the values you specified in NPM initialization statements.

The following sections describe the information contained in the report. See NPM Diagnosis for an example of the configuration report.

List of All Initialization Statements

This list shows the value NPM used for each keyword on each initialization statement. If NPM used a default or calculated default value, or adjusted the value you specified, this list shows the exact value NPM used.

NPM truncates statement names to eight characters. For example, the THRESHOLD statement is listed as THRESHOL.
NPM Virtual Storage Allocation

The table in Figure 19 defines how much storage has been allocated during NPM initialization and where it is located. The numbers in the table are rounded to the nearest whole number. For information about how NPM uses this storage, see NPM Concepts and Planning.

<table>
<thead>
<tr>
<th>PRIVATE</th>
<th>CSA</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BELOW 16 MEG</td>
<td>268 K</td>
<td>5 K</td>
</tr>
<tr>
<td>ABOVE 16 MEG</td>
<td>876 K</td>
<td>105 K</td>
</tr>
<tr>
<td>TOTAL</td>
<td>1144 K</td>
<td>110 K</td>
</tr>
</tbody>
</table>

Figure 19. NPM Virtual Storage Allocation

Virtual Storage by Subpool

This part of the configuration report lists NPM’s private virtual storage allocation by subpool:

- **BUF**: NPM buffer pool
- **CAT**: Common address table and related control blocks
- **CBS**: Control block pool
- **CCT**: Communication control task
- **CPT**: Configuration control blocks pool
- **FST**: File service task buffer
- **INIT**: Initialization storage
- **PNL**: Presentation services panel buffer pool
- **RACF**: Related control blocks
- **VDT**: VTAM collection

**Note:** Initialization storage is freed when initialization is complete.

You can use storage allocation information to help tune NPM. Use it in conjunction with the performance measurement table on the Performance Measurement Table Display panel. For example, if the number of FST control blocks is high in the Virtual Storage Allocation by SUBPOOL report, look at the number of free FSTs on the performance measurement table. If this number is high, you should define fewer FSTs at initialization. See Chapter 12, “Tuning NPM” on page 209 for specific tuning information.

Configuration Summary from All NCPs Defined at Initialization

The header and data fields for this report were taken from the NCP RRTs and the NCP commands. The following list describes the headers:

- **Type**: Resource class (configuration hierarchy).
- **Name**: The name of the resource.
- **EA**: The element address.
- **NPA**: Has the resource been enabled for NPM network data collection (Yes|No).
- **DNC**: Has the resource been enabled for dynamic network collection (Yes|No).
Reading the Configuration Report

DR Has the resource been enabled for dynamic reconfiguration (Yes|No).

SCL Session collection status (Y|V|R|X|N). See NPM User’s Guide for more information about how data is compiled and written to the session file.

UP Name of resource above this resource in the network hierarchy.

NEXT The name of the next peer resource in the network hierarchy.

DOWN The name of the first resource below this resource in the network hierarchy.

DWCT Down count: the number of resources defined directly below this resource. An example of this count is the number of PUs directly below a link. This count does not include the LUs below the PU. The PU down count includes all LUs directly below it.

MAXL Maximum number of LUs on which NPM can collect session data, plus the number of LUs to be excluded.

SPEED1 The sending speed on a line in bits per second.

SPEED2 The receiving speed on a line in bits per second.

CPB The address of the NPM control block for that resource.

SCB The address of the session collection control block.

Configuration Hierarchy

The configuration hierarchy is based on all NCPs defined during initialization and on the initialization statements. The configuration hierarchy is defined by the following list:

HOST Host name from the HOST initialization statement.

APPL Application name from the APPL initialization statement.

APPL/S Application synonym from the APPL initialization statement.

NCP NCP name from the NPM NCP command.

LINK Link name from the NCP RRT.

PU Physical unit name from the NCP RRT.

LU Logical unit name from the NCP RRT and LUGROUP definition.

LINE Line name from the NCP RRT.

CLUSTER Cluster name from the NCP RRT.

TERM Terminal name from the NCP RRT.

CDRM Cross-domain resource manager name from the CDRM initialization statement.

LUGROUP LU group name as defined in the FNMSCMDS data set.

PHYLINK NTRI physical link name from the NCP RRT.

LOGLINK NTRI logical link name from the NCP RRT.
Verifying NPM Functions

Verifying NPM Functions

NX25LINK  NPSI link name from the NCP RRT.
XX25LINK  XI link name from the NCP RRT.
NEOLINK   NEO link name from the NCP RRT.
NX25PU    NPSI physical unit name from the NCP RRT.
XX25PU    XI physical unit name from the NCP RRT.
NX25VC    NPSI virtual circuit name from the NCP RRT.
NEOPU     NEO physical unit name from the NCP RRT.
ODLCLNPU  ODLC LAN station physical unit from the NCP RRT.
ODLCLNLK  ODLC LAN physical link from the NCP RRT.
ETHERPL   Ethernet physical link from the NCP RRT.
FRPHYLK   Frame relay physical link from the NCP RRT.
FRLLOGLK  Frame relay logical link from the NCP RRT.
FRPHYST   Frame relay physical station from the NCP RRT.
FRLMIPU   Frame relay local management interface PU from the NCP RRT.

Totals by Resource Type Table
The fields in this table provide a summary listing of the resources in the configuration listing. The LU count is for local and remote LUs.

Configuration Symbol Resource Table
The fields in this table provide the following data:

- Number of entries in the symbol table
- Number of synonym chains
- Average length of a synonym chain
- Maximum length of a synonym chain
- Minimum length of a synonym chain

The number of synonym chains is determined by the value specified by the CSRTSIZE parameter on the BUFFERS initialization statement.

Synonym Chain Distribution Table
The fields for this table are defined as shown in the following list:

Percent  Percentage of the maximum length of the synonym chain
Value    Percent times the maximum length of the synonym chain, rounded upward
Count    Number of elements on the chain.

For optimal CPU utilization, the distribution should be skewed toward the low values, for example, the synonym chains with the shortest length.

Verifying NPM Functions
After you have reviewed NPM’s configuration information, use the following procedure to check that NPM was installed correctly. See NPM User’s Guide for more information or if you have a question concerning any of the panels presented.

Note: If you find problems, first verify that you followed all the steps in Chapter 2, “Installing NPM with the NPM Initialization Program” on page 11 or
Step 1. Bring up the NPM logon panel (FNM0LLOG) by entering the following command:

\[ \text{LOGON APPLID(nnn)} \]

where:

\( nnn \) is the application name of NPM as defined to VTAM.

See “Step 3. Define NPM to VTAM” on page 30 for more information.

---

**Step 2.** Log on to NPM using an operator ID and password that you have defined. Choose an operator ID with a profile that has extensive authority so you can issue collection commands, control commands, and view the data analysis panels. You see the Primary Options panel (FNM00PRI).
Verifying NPM Functions

Step 3. If you have installed optional national language features, type =0.5 in the Select Option field to display the National Language Selection panel (FNM00LNG). Each of the languages you have installed are listed. You can change languages by selecting from this list.

Figure 22. National Language Selection (FNM00LNG)

Step 4. Verify that you have the correct profile definitions in place for this operator ID by typing =0.3 in the Command field. This takes you to the Security Profile panel (FNM00UPB).
Step 5. In the Command field, type =9.5 to display the Data File Display and Management panel (FNM02OFL). Each of the files you have defined to NPM is listed.

Step 6. In the Command field, type =9.4;DOWN to display the Summary Display panel (FNM02OPS). This panel shows various information about the NPM operating environment. Check that the information displayed is correct.
Verifying Session Collection

Use the following procedure to verify that the session collection function was properly installed. If you do not want to verify session collection, go to “Verifying Network Collection” on page 191.

**Step 1.** In the Command field, type =2.1 to display the Start Session panel (FNM02SCL).

**Step 7.** In the Command field, type =9.4;DOWN;DOWN;DOWN to display the Virtual Storage Summary Display panel (FNM02OPV). NPM’s use of virtual storage is presented.
Step 2. In the Resource Name field, type the name of an application that you have previously defined to NPM and press ENTER. You should see message FNM275I, indicating that session collection was started.

Step 3. In the Command field, type =9.4;DOWN;DOWN; to display the File Services Summary panel (FNM02OPR). This panel shows the current logging activity for NPM. The panel is refreshed with current information when you press ENTER. Because you started session collection in the previous step, the total number of session records increase at every session interval.
Step 4. In the Command field, type =2.2 to display the Session Status/Stop (Active) Resource panel (FNM03SSA). This panel allows you to view the status of any session collection and to stop collection if you want. Type P next to the resource for which you started session collection and press ENTER. You should see message FNM275I, indicating that collection was stopped.

![Figure 29. Session Status/Stop (Active) Node Selection (FNM03SSA)](image)

Step 5. In the Command field, type =2.7 to display the NPM Session Analysis Summary panel (FNM02SAN). You can view summary panels of collected session data after providing selection criteria on this panel.

![Figure 30. NPM Session Analysis Summary (FNM02SAN)](image)
Step 6. In the Select Option field, type 1. In the Identifier field, type the name of the application for which you collected data and press ENTER. The NPM Session Analysis Summary - Application panel (FNM03SMN) displays. Verify that the information presented is accurate.

![Figure 31. NPM Session Analysis Summary - Application (FNM03SMN)](image)

**Verifying Network Collection**

This section explains how to verify that the network collection function was properly installed. If you do not need to verify this function, go to “Verifying LAN Bridge Collection” on page 196.

If you have defined any NCPs, either through FNMSTRT or an NPM EXEC, you can verify the network collection function by going to the steps listed below.

If you want to verify network collection but you have not defined any NCPs to NPM, create an NPM EXEC member that contains at least one NCP command. For information about creating NPM EXECs, see “Processing NPM EXEC Members (FNMSCMDS)” on page 160.

Execute the NPM EXEC by typing =9.8 in the Command field and pressing ENTER to display the Execute an NPM EXEC panel (FNM02OEX). In the NPM EXEC Name field, type the EXEC name and press ENTER. You see message FNM114I, indicating that the EXEC processed successfully.

Use the following procedure to verify network collection:

**Step 1.** In the Command field, type =9.3 to display the NPALU Display and Management panel (FNM02OCC). Each of the NCPs that you have defined is listed.

If the NCP that you want to use for testing network collection has a status of DISCONNECTED, type 8 next to it to bind the NPALU. The status changes to CONNECTED.
Verifying Network Collection

Step 2. In the Command field, type \(=1.1\) to display the NCP Management Resource List panel (FNM03RSL). This panel shows the NCP you have defined to NPM. To start network collection on the NCP, type \(S\) in the option field, and press ENTER.

You will now see the NCP Management Network Start panel (FNM03STD). This panel allows you to choose the interval on which collection is to be started, or to set a start and stop time for the collection. Press ENTER.
Verifying Network Collection

You will now see the next NCP Management Network Start panel (FNM03STM). You can use this panel to set monitor thresholds, if required, and to override the default destinations for performance and monitor data. Press ENTER to start the collection.

In the Command field, type =9.4.DOWN;DOWN to display the File Services Summary Panel (FNM02OPR). The File Services Summary panel (FNM02OPR) shows the current logging activity for NPM. The panel is refreshed with current information when you press ENTER. Because you started network collection in the previous step, the total number of network records increases at every network interval.
Verifying Network Collection

---

**Figure 36. File Services Summary (FNM02OPR)**

**Step 3.** In the Command field, type `=1.9.4` to display the Network Review panel (FNM03RVM). You can view detailed information about the collected network data after providing selection criteria on the Network Review panel (FNM03RVM).

---

**Figure 37. Network Review (FNM03RVM)**

**Step 4.** In the Resource Name field, enter the name of the NCP on which you collected network data. For the Data Type, type `DETAIL`. Press ENTER. The Network Review Data panel (FNM03RVP) is displayed. Verify that the information presented is accurate.
Verifying VTAM Statistics Data Collection

This section explains how to verify that the VTAM Statistics collection function was properly installed. If you do not need to verify this function, go to “Verifying LAN Bridge Collection” on page 196.

Use the following procedure to verify VTAM statistics data collection:

**Step 1.** In the Command field, type =5 to display the VTAM Resource Type Selection panel (FNM01VTM). This panel displays the selection options for VTAM statistics collection.

```plaintext
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>View Environment Data</td>
</tr>
<tr>
<td>2</td>
<td>Collect and View Global Data</td>
</tr>
<tr>
<td>3</td>
<td>Collect and View Buffer Pool Data</td>
</tr>
<tr>
<td>4</td>
<td>Collect and View Virtual Route Data</td>
</tr>
<tr>
<td>5</td>
<td>Collect and View Device Data</td>
</tr>
<tr>
<td>6</td>
<td>Collect and View Application Data</td>
</tr>
<tr>
<td>7</td>
<td>Collect and View Address Space Data</td>
</tr>
<tr>
<td>9</td>
<td>Direct Commands</td>
</tr>
</tbody>
</table>
```

**Figure 39. VTAM Verification (FNM01VTM)**
Verifying LAN Bridge Collection

Step 2. In the Select Option field, type 1 to display the View VTAM Environment Data panel (FNM03VE1). This panel displays information from your VTAM installation configuration. See NPM User’s Guide for more information about the fields contained on this panel.

Note: The following message is generated if your VTAM collection was unsuccessful or if you specified VTAMCOLL=NO:

FNM219I NO DATA FOR REQUEST

Verifying LAN Bridge Collection

This section explains how to verify that the LAN bridge collection function was properly installed. If you do not need to verify this function, go to “Verifying LAN Segment Collection” on page 200.

If you have defined any LAN Managers, either through FNMSTRT or an NPM EXEC, you can verify the LAN bridge collection function by completing the steps listed below.

If you want to verify LAN bridge collection but you have not defined any LAN Managers to NPM, create an NPM EXEC member that contains at least one LANMGR command. For information about creating NPM EXECs, see “Processing NPM EXEC Members (FNMSCMDS)” on page 160.

Execute the NPM EXEC by typing =9.8 in the Command field and pressing ENTER to display the Execute an NPM EXEC panel (FNM02OEX). In the NPM exec Name field, type the EXEC name and press ENTER. You should see message FNM114I, indicating that the EXEC processed successfully.

Use the following procedure to verify LAN bridge collection:

Step 1. In the Command field, type =9.9.1 to display the LAN Bridge Control panel (FNM02LBB).
Verifying LAN Bridge Collection

This panel displays parameters that affect all LAN bridge data collections. Ensure that the VSAM field destination for Interval records is set to YES and that the Interval Length field is set to 5 minutes.

<table>
<thead>
<tr>
<th>Command ===</th>
<th>Host Name = LOCAL</th>
</tr>
</thead>
</table>
| Details | Host Name = LOCAL
| Data Destinations | NPMLOG VSAM ALERT GLOBAL
| Monitor | YES YES N/A N/A
| Resolve Monitors | NO NO NO NO
| Interval Length (minutes) | 5 (5,10,15,20,30,60)
| Interval Synchron (minutes) | 0 (Must be less than Interval Length)

To update the list of LAN resources, enter RECONFIG on the Command line.

**Figure 41. LAN Bridge Control (FNM02LBB)**

**Step 2.** In the Command field, type =4.1 to display the LAN Bridge Selection panel (FNM02LBS). Each of the bridges connected to the LAN Managers that you have defined is listed.

<table>
<thead>
<tr>
<th>Bridge</th>
<th>Collection</th>
<th>Status</th>
<th>Status</th>
<th>Forward</th>
<th>Segments</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRIDG#1</td>
<td>INACTIVE</td>
<td>A12 B1B</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#2</td>
<td>INACTIVE</td>
<td>A13 B1C</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#3</td>
<td>INACTIVE</td>
<td>A14 B1D</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#4</td>
<td>INACTIVE</td>
<td>B15 DCC</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#5</td>
<td>INACTIVE</td>
<td>B16 DCC</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#6</td>
<td>INACTIVE</td>
<td>B17 DCC</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#7</td>
<td>INACTIVE</td>
<td>B18 DCC</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRIDG#8</td>
<td>INACTIVE</td>
<td>C13 DCC</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 42. LAN Bridge Selection (FNM02LBS)**

**Step 3.** Next to one of the bridges from which you want to collect data, type C and press ENTER. The LAN Bridge Collection Control panel (FNM03LBC) is displayed.
Verifying LAN Bridge Collection

**Step 4.** In the Command field, type S to indicate that you want to start collection on this bridge and press ENTER. You should see message FNM275I, indicating that LAN Bridge collection has started. You should also see the Collection Status field change from INACTIVE to STARTED.

![Figure 43. LAN Bridge Collection Control (FNM03LBC)](image)

**Step 5.** In the Command field, type =9.4;DOWN;DOWN; to display the File Services Summary panel (FNM02OPR). This panel shows the current logging activity for NPM. The panel is refreshed with current information when you press ENTER. Because you started LAN Bridge collection in the previous step, the total number of LAN records increases at every LAN Bridge interval (5 minutes).

![Figure 44. File Services Summary (FNM02OPR)](image)
**Step 6.** In the Command field, type `=4.1` to display the LAN Bridge Selection panel (FNM02LBS) again. Next to the bridge from which you are collecting data, type `P` and press ENTER. You should see message FNM275I, indicating that collection is completed. The Collection Status field returns to INACTIVE.

![Figure 45. LAN Bridge Selection (FNM02LBS)](image)

**Note:** This display refreshes itself based upon current bridge information. If the list is currently sorted by status (the default) and the status of any of the displayed bridges changes, these bridges may no longer be displayed.

**Step 7.** On the LAN Bridge Selection panel (FNM02LBS), next to the bridge for which you collected data, type `R` and press ENTER. The LAN Review panel (FNM02LBM) is displayed. You can view detailed information about the collected LAN Bridge data after specifying selection criteria on this panel.
Verifying LAN Segment Collection

This section explains how to verify that the LAN segment collection function was properly installed. If you do not need to verify this function, go to “Stopping NPM” on page 206.

If you have defined any LAN Managers, either through FNMSRT or an NPM EXEC, you can verify the LAN segment collection function by completing the following steps.

Step 8. For the Data Type, type DETAIL and press ENTER. The LAN Bridge Data Detail panel (FNOM3LBR) is displayed. Verify that the information presented is accurate.
If you want to verify LAN segment collection but you have not defined any LAN Managers to NPM, create an NPM EXEC member that contains at least one LANMGR command. For information about creating NPM EXECs, see “Processing NPM EXEC Members (FNMSCMDS)” on page 160.

Execute the NPM EXEC by typing =9.8 in the Command field to display the Execute an NPM EXEC panel (FNM02OEX). In the NPM exec Name field, type the EXEC name and press ENTER. You should see message FNM114I, indicating that the EXEC processed successfully.

Use the following procedure to verify LAN segment collection:

**Step 1.** In the Command field, type =9.9.2 to display the LAN Segment Control panel (FNM02LSB). This panel displays parameters that affect all LAN segment data collections. Ensure that the VSAM field destination for Interval records is set to YES and that the Interval Length field is set to 5 minutes.

![LAN Segment Control Panel](Figure 48. LAN Segment Control (FNM02LSB))

**Step 2.** In the Command field, type =4.2 to display the LAN Segment Selection panel (FNM02LSS). Each of the segments connected to the LAN Managers you have defined is listed.
Verifying LAN Segment Collection

Figure 49. LAN Segment Selection (FNM02LSS)

Step 3. Next to one of the segments from which you want to collect data, type C and press ENTER. The LAN Segment Collection Control panel (FNM03LSC) is displayed.

Step 4. In the Command field, type S to indicate that you want to start collection on this segment and press ENTER. You should see message FNM275I, indicating that LAN segment collection has started. You should also see the Collection Status field change from INACTIVE to STARTED.

Figure 50. LAN Segment Collection Control (FNM03LSC)
**Step 5.** In the Command field, type `=9.4;DOWN;DOWN;` to display the File Services Summary panel (FNM02OPR). This panel shows the current logging activity for NPM. The panel is refreshed with current information when you press ENTER. Because you started LAN segment collection in the previous step, the total number of LAN records increases at every LAN segment interval (5 minutes).

```
FNM02OPR  
NPM V2R4 5655-043  
CONTROL FUNCTIONS  
FILE SERVICES SUMMARY  

Command ===>  
Host Name = LOCAL  

<table>
<thead>
<tr>
<th>File Name</th>
<th>Current</th>
<th>Total</th>
<th>Lost</th>
<th>Record Type</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNMSES1</td>
<td>397</td>
<td>397</td>
<td>186</td>
<td>Session</td>
<td>186</td>
</tr>
<tr>
<td>FNMREV1</td>
<td>15</td>
<td>76</td>
<td>0</td>
<td>Network</td>
<td>52</td>
</tr>
<tr>
<td>FNMLWG1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>RTM</td>
<td>0</td>
</tr>
<tr>
<td>FNMLOG1</td>
<td>273</td>
<td>275</td>
<td>15</td>
<td>Accounting</td>
<td>0</td>
</tr>
<tr>
<td>FNMLVLOG2</td>
<td>237</td>
<td>237</td>
<td>0</td>
<td>Command/Event</td>
<td>62</td>
</tr>
<tr>
<td>FNMSMF</td>
<td>262</td>
<td>262</td>
<td>0</td>
<td>LAN</td>
<td>0</td>
</tr>
<tr>
<td>FNMALET</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VTAM</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NetWare</td>
<td>0</td>
</tr>
</tbody>
</table>

Press NEXT or PREVIOUS for additional PMT displays  
PF 1=HELP  2=  3=END  4=  5=  6=  
PF 7=PREVIOUS  8=NEXT  9=  10=TOP  11=BOTTOM  12=RETURN  

Figure 51. File Services Summary (FNM02OPR)

**Step 6.** In the Command field, type `=4.2` to display the LAN Segment Selection panel (FNM02LSS) again. Next to the segment from which you are collecting data, type `P` and press ENTER. You should see message FNM275I, indicating that collection is completed. The Collection Status field returns to INACTIVE.

```
FNM02LSS  
NPM V2R4 5655-043  
LAN MANAGEMENT  
LAN SEGMENT SELECTION  

Command ===>  
Host Name = LOCAL  
Segment Count = 20  
Display = 10.46.23  
Interval = 10.30.00  
Enter Command: SORT <NUMBER/STATUS/UTIL>, LOCATE Segment  
or an Option: C=Collection, P=Stop, D=Detail, R=Review, Z=Stop All  

<table>
<thead>
<tr>
<th>OPTION</th>
<th>SEGMENT</th>
<th>COLLECTION</th>
<th>MONITOR</th>
<th>Data</th>
<th>&lt;-- SEGMENT UTILIZATION --&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>=&gt; P</td>
<td>A01</td>
<td>STARTED</td>
<td>INACTIVE</td>
<td>0</td>
<td>30% 29% 27%</td>
</tr>
<tr>
<td>=&gt; A02</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; A03</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; A04</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; B01</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; B02</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; B03</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; B04</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=&gt; C01</td>
<td>INACTIVE</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PF 1=HELP  2=NATLANG  3=END  4=  5=  6=  
PF 7=  8=FORWARD  9=  10=  11=BOTTOM  12=RETURN  

FNM275I LSEGCOLL COMMAND COMPLETED  

Figure 52. LAN Segment Selection (FNM02LSS)
Verifying LAN Segment Collection

Note: This display refreshes itself based upon current segment information. If the list is currently sorted by status (the default) and the status of any of the displayed segments changes, these segments might not be re-displayed.

Step 7. On the LAN Segment Selection panel (FNM02LSS), next to the segment for which you collected data for, type R and press ENTER. The LAN Review panel (FNM02LBM) is displayed. You can view detailed information about the collected LAN segment data after specifying selection criteria on this panel.

![Figure 53. LAN Review (FNM02LBM)](image)

Step 8. For the Data Type, type DETAIL and press ENTER. The LAN Segment Data Detail panel (FNM03LSR) is displayed. Verify that the information presented is accurate.

![Figure 54. LAN Segment Data Detail (FNM03LSR)](image)
Verifying NetWare Resources Data Collection

This section explains how to verify that the NetWare resources data collection function was properly installed. If you do not need to verify this function, go to “Stopping NPM” on page 206.

If you have defined any NetWare resource, either through FNMSTRRT or an NPM EXEC, you can verify the NetWare resources data collection function by completing the following steps.

If you want to verify NetWare resources data collection but you have not defined any NetWare resources to NPM with the LWGRES command, create an NPM EXEC member that contains at least one LWGRES command. For information about creating NPM EXECs, see “Processing NPM EXEC Members (FNMSCMDS)” on page 160.

Execute the NPM EXEC by typing =9.8 in the Command field to display the Execute an NPM EXEC panel (FNM02OEX). In the NPM exec Name field, type the EXEC name and press ENTER. You should see message FNM114I, indicating that the EXEC processed successfully.

Use the following procedure to verify NetWare resources data collection:

Step 1. Issue an LWGCOLL command from the console on the resource you specified in the LWGRES command. The following figure shows an example of how to code the commands.

```
LWGRES  OPTION=ADD,
        SP=NETWARE1,
        NETWORK=3BCE5A,
        NODEADDR=FFE22A,
        AGENT=SERVER0,
        RESOURCE=NWAGE

LWGCOLL OPTION=START,
        SP=NETWARE1,
        AGENT=SERVER0,
        CLOCK=(/zerodot8./zerodot/zerodot./zerodot/zerodot,16./zerodot/zerodot./zerodot/zerodot),
        DAILY=YES,
        GROUP=ALL,
        DYNAMIC=YES,
        SRVUTIL=(2/zerodot,9/zerodot),
        CONNUTIL=(2/zerodot,9/zerodot),
        VOLUTIL=(2/zerodot,9/zerodot)
```

Step 2. In the Command field, type =9.5 to display the Data File Display and Management panel (FNM02OFL). This panel shows a list of NPM files. If the NetWare resources data collection has started, the list of files shown on the panel is updated, the FNMLWGx file is highlighted, and the status of the file is shown as active.
Stopping NPM

An operator whose profile specifies CONTROL=YES can end NPM at any time, either online or from the system console.

Notes:

1. Do not stop NPM by forcing it from the operating system. If you force NPM from the operating system, you can lose the connection with the VTAM program. When NPM ends, it gives control to NPM ESTAE exit routines to properly close the interfaces. Under certain conditions, after NPM has ended, you should clean up NPM's CSA control blocks. See “Cleaning Up NPM's CSA Control Blocks” on page 207 for information about cleaning up NPM's CSA control blocks.

2. If you have active NetWare collections, issue an LWGSTOP command from the system console before you stop NPM to stop all active NetWare collections and to prevent the NPM NetWare Agent from collecting and forwarding data. See the section of the NPM Diagnosis manual that describes checkpoint processing, for more information.

Stopping under MVS

To end NPM under MVS, enter the following command from the system console:

```
P procname
```

where:

```
procname  Is the name of the procedure used to start NPM.
```
Cleaning Up NPM’s CSA Control Blocks

Clean up NPM’s CSA control blocks under the following conditions:

- You want to run a different release of NPM after migrating
- You used the MVS force command to stop NPM
- NPM was stopped without running its cleanup routines

The NPM program FNMCLNUP deletes any outstanding CSA control blocks. Run this program only when NPM is not running. You do not need to run this job unless one of the conditions listed in the previous paragraph has occurred.

Execute the sample job FNMCLNPJ. This sample job is located in NPM.V2R4M0.SFNMJCL1. FNMCLNUP issues a return code after it runs. The following list describes the possible return codes:

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>All storage was freed. FNMCLNUP completed normally freeing all NPM CSA storage.</td>
</tr>
<tr>
<td>04</td>
<td>Error in FREEMAIN or PAGE FREE. FNMCLNUP was not able to free some CSA storage.</td>
</tr>
<tr>
<td>08</td>
<td>The monitor address table and NPM anchor control block are not defined. No storage was freed. This condition can occur if FNMCLNUP is run twice.</td>
</tr>
<tr>
<td>12</td>
<td>NPM is still active. No storage is freed.</td>
</tr>
<tr>
<td>16</td>
<td>The CVT does not point to the monitor address table. There is an error in the definition of the NPM control blocks. No storage is freed.</td>
</tr>
<tr>
<td>20</td>
<td>FNMCLNUP is not authorized. FNMCLNUP must be link-edited in an APF-authorized library in order to release the CSA control blocks.</td>
</tr>
</tbody>
</table>
Chapter 12. Tuning NPM

After NPM is running, you can tune your NPM definition parameters to make NPM run as fast as possible using the smallest amount of storage.

Tuning is a trade-off. Each function of NPM has costs in terms of processor load, communication controller load, and storage. As you collect data on more network resources or collect more types of data, your costs go up. Therefore, one of the most important steps in tuning is deciding what data you need and collecting only that data.

Tuning is an iterative process. It is unlikely you will be able to select optimal settings in one pass. Use the information in this chapter to select internal parameter settings that will improve performance or reduce storage requirements. Then experiment to find the best settings for your specific installation.

Inactivating Session Collection

If your installation has a small amount of CSA available or you do not need to collect session data, specify HOSTCOLL=NO on the NPM initialization statement. If HOSTCOLL=NO, session collection, SMF management, dynamic network collection, RTM collection, and the graphic subsystem are all inactive. NPM, therefore, uses no CSA or accounting functions with HOSTCOLL=NO.

Cutting CSA Usage during Session Collection

If your network uses too much CSA because of the number of LUs or NCPs in your network, you can limit the number of these resources from which data can be collected at one time. The MAXL parameter on the HOST initialization statement limits the number of LUs.

The MAXL parameter on the NCP command limits the amount of CSA allocated for resources defined to an NCP. You set a limit with MAXL when you do not need to collect on all resources defined to an NCP, especially all DR resources.

If you want to be flexible in your use of CSA, you can issue NCP commands to replace NCPs and vary the size of the value you code for MAXL. Replacing NCPs enables you to specify only the number of LUs on which you are collecting data at a given time. For example, suppose you need to collect data on 200 LUs and you have only enough storage for 100 LUs each time. Code an NCP command for the same NCP, using the REPLNCP parameter and set MAXL=100. When you start collection, you can collect on a different set of 100 LUs each time.

In the same manner, you could start with MAXL=50 and collect on 50 LUs. Then replace the NCP with MAXL=100 and continue to collect on the first 50 LUs plus another 50 LUs.

If you only have a small amount of CSA storage available, delete the NCPs on which you are not collecting data. Deleting NCPs frees CSA for other data you need to collect. To delete an NCP, code OPTION=DELETE on the NCP command for that NCP. If you need to collect data from an NCP that you have deleted, code a new command for that NCP.
Tuning VSAM

See *NPM Concepts and Planning* for information about how to calculate CSA storage requirements for your system.

### Collecting Session Data by Application

You can save CSA without affecting the quality of the session data by collecting data by application rather than by NCP. You can then define the NCPs without session collection (SESSCOLL=NO) to save CSA. You should still reserve the minimum number of LU entries in CSA (MAXL=2) on the HOST and CDRM statements.

You can further minimize private storage by specifying the MAXL value for each application to reflect a maximum number of sessions that will be active during two session intervals.

When collecting session data by application, some LU sessions record transit times that are not meaningful. Examples include printers, NPALUs, other applications, and cross-domain resource managers. These LUs can skew data for the application. The LU list option on the summary displays can be used to identify the LUs that are causing transit time skewing. To prevent these LUs from skewing the data for the application, use the following form of the session collection command:

\[
\text{SESSCOLL LU=printer_name,CDRM=ncp_name,SESSH=V} \\
\text{SESSCOLL LU=npalu_name,CDRM=ncp_name,SESSH=X} \\
\text{SESSCOLL LU=appl_name,CDRM=host_name,SESSH=V} \\
\text{SESSCOLL LU=cdrm_name,CDRM=node_name,SESSH=X}
\]

SESSH=V collects session data but records only traffic volumes, not transit times. SESSH=X collects no data at all for the LU. You must update the MAXL value for the HOST and CDRM statements to reflect the number of LUs for which SESSCOLL commands are issued.

An alternative method for setting NCP resources is to use GENERIC, LINE, LU, and PU statements in NPM EXECs stored in the FNMSCMDS data set, as shown in the following example:

\[
\text{LU NAME=printer_name,SESSH=X} \\
\text{LU NAME=npalu_name,SESSH=X}
\]

With this method, you do not have to issue SESSCOLL commands. For this method to be effective, SESSCOLL=YES must be specified on the NCP command.

### Tuning VSAM File Processing

When you tune NPM, DASD I/O is one area that can greatly affect performance. Because NPM VSAM files are used for online analysis, they are excellent candidates for tuning. You can tune NPM VSAM file organization and use the local shared resource (LSR) option.

**Note:** To ensure that NPM has sufficient storage to process its functions, use the LSR option when defining a large number of VSAM files.
Optimizing Data Set Organization

To optimize data set organization, complete the following steps:

**Step 1.** Keep the index and data components of the VSAM data sets on separate DASD volumes. This allows more overlapping of physical I/O.

**Step 2.** Reorganize the data sets regularly, keeping the FREESPACE at approximately 40% of the total data set size.

**Step 3.** Increase the values coded for BUFND and BUFNI in the FILE statement in FNMINIT. You can allocate up to 255 buffers for each VSAM file.

Using the LSR Option

To get the best VSAM performance, you should use the VSAM LSR option. This is important because NPM lets you collect and analyze VSAM data at the same time.

To use the LSR option, you must specify MACRF=LSR on the FILE statement that defines your VSAM data set to NPM. See NPM Reference. You can also update the BUFND and BUFNI values on the statement.

You must also code TYPE=LSR in the BLDVRP macro to enable the LSR option. See 12 on page 132 for instructions on using the BLDVRP macro.

See VSAM Programming Guide for more information.

Using the Performance Measurement Table

The NPM performance measurement table (PMT) shows the common storage area (CSA) used for the host and lists other system information as it applies to the NPM operating environment. This information can be useful for tuning NPM.

You should pay special attention to low buffer and lost record values. The low buffer value tells you the lowest number of available buffers that NPM reached. If this number is very small, you need to allocate more buffers or collect less data. If this number is large, you have allocated more buffers than you need.

If the lost record value is greater than zero, you are losing data. You must allocate more buffers or reduce the amount of data that you are trying to collect. Adjust the BUFSEGS parameter of the BUFFERS statement accordingly. Changes in the BUFSEGS value affect private storage requirements.

Tuning Initialization Statements

You can adjust the parameters of NPM initialization statements to change the amount of storage used, the amount of network traffic, and the amount of CPU used. This section lists the initialization statements that can affect NPM performance and suggests how to select the best values for their parameters.
Tuning Initialization

BUFFERS Statement

You can set the following parameters on the BUFFERS statement to tune NPM performance:

**BUFFNO**
Determines how much CSA is allocated for full PIU collection and for RTM data. If you only need minimum PIU data and do not need RTM data, code BUFFNO=0. Otherwise, BUFFNO=200 should be adequate for a small- to medium-size environment. If BUFFNO is too small, PIUs and RTM data could be lost. If you code a BUFFNO parameter on the EXEC command used to start NPM, it overrides the value coded on the BUFFERS statement.

**BUFSEGS**
Determines the number of private storage buffer segments that NPM uses for many of its functions.

When coding BUFSEGS, you should consider how many NCPs, new VTAM resources, and LAN Managers will be added dynamically after NPM initialization.

While you can dynamically add to the configuration, the value for BUFSEGS cannot be changed without stopping NPM. A typical NCP with 1200 LUs (SDLCs only) and supporting resources would require approximately 2345 BUFSEGS to collect on all resources.

If the value coded for BUFSEGS is too small, NPM issues buffer pool storage messages. If the number of buffer segments is depleted, NPM will attempt to acquire 1280 additional buffer segments. If NPM is unable to acquire additional buffer segments, it abends with code U0001. Although NPM is able to dynamically obtain more buffer segments, you should try to allocate enough buffer segments initially so this does not happen. This reduces the chance that NPM could abend due to lack of storage, and improves NPM's performance.

You should always adjust BUFSEGS when significant changes to your configuration occur. Use the performance measurement table to determine what further adjustments to make. Tuning BUFSEGS is an iterative process. You should return to the performance measurement table periodically to monitor the effect of any changes made to BUFSEGS.

With the session monitor function, it is possible to generate many messages. Each message generated uses two buffer segments. These segments are in a pool. When a message is generated, NPM gets two buffer segments from this pool, and queues the message to be issued. When the message is issued, the buffer segments are returned to the pool. At the end of the interval, NPM will need two buffer segments for the message for each session monitor exceeded.

**CSRTSIZE**
Determines the number of entries in the configuration symbol resource table. For a starting value, use the number of resources divided by 40. Then, check the configuration symbol table in FNMILOG to determine the number of entries used. Remember to take into account the number of NCPs that will be added dynamically when choosing a value.

A larger CSRTSIZE results in shorter resource chains and therefore better processor performance. In extreme cases, a small CSRTSIZE value may make NPM run so slowly that it appears to stop.
Before increasing CSRTSIZE, consider storage and performance trade-offs. Increasing CSRTSIZE by one uses four additional bytes of both CSA and private storage when HOSTCOLL=YES. When HOSTCOLL=NO, increasing CSRTSIZE by one uses four additional bytes of private storage.

**FSTBUFS**

Specifies the number of 512-byte file services buffers allocated for the file services task. These buffers are used to write session and network records to VSAM databases, SMF, and sequential data sets. If this set of buffers is in use, NPM obtains an additional set of 8 buffers at a time.

The default FSTBUFS is adequate to support collection on all resources for a network of logical units and supporting resources equaling 6K. To select your own value, examine the configuration report in the FNMILOG after NPM initialization. Find the “Totals by Resource Type” portion of the report and locate the total resources. Divide the number of resources by four and use this value for FSTBUFS. This will support collection for all your resources. See “Reviewing Your NPM Configuration” on page 181 for more information.

**Note:** The actual value used by NPM might be slightly higher than the value specified due to internal memory management.

To select a good value for FSTBUFS, consider both the number of resources that you can dynamically add to your configuration and the number of resources for which you will actually collect data. Only those resources for which you will collect data require file services buffers. If you do not plan to collect on all resources, you can save storage by allocating less FSTBUFS buffers.

**MAXGSD**

Specifies the maximum number of records that NPM processes when executing a request for stored data. If the actual number of records retrieved is greater than the maximum number allowed (as specified by this parameter), NPM rejects the request and issues message FNM1109.

NPM uses this value to determine how much storage to obtain when processing a request for stored data. The larger the value, the more temporary storage NPM requires for its work area. You can set MAXGSD to any number from 10–9999. The default is 1000.

**QCBCHN and TPRBCHN**

QCBCHN specifies how queue control blocks are chained and TPRBCHN specifies how the teleprocessing request blocks are chained.

Setting QCBCHN=TOP and TPRBCHN=TOP improves network performance. However, in the event of an NPM error, you lose the diagnostic information. You can retain this diagnostic information by setting these parameters to BOTTOM. You might want to set these parameters to BOTTOM when installing and testing NPM. Then, change them to TOP when you are running NPM as a production system.

**SSBHASH**

Allocates NPM private storage for a hash table. While the default value for SSBHASH (1019) should be adequate, you can improve NPM’s CPU utilization by coding a hash table size that represents a
one-to-one entry-to-resource ratio. To calculate the maximum number of resources for which session data can be collected at any given time, add the values of all MAXL parameters coded on the APPL statements. Use this value as the hash table size. If the value is not a prime number, NPM will round it to the nearest prime number.

Before increasing SSBHASH, consider storage and performance trade-offs. Increasing the value of SSBHASH by one requires sixteen additional bytes of storage.

**SESSNO**
Determines the amount of CSA that NPM allocates for session collection of PIUs. If the value of SESSNO is too small, NPM data is lost, but the actual PIUs in the network are not affected. A good starting value is SESSNO=100.

**SSRTSIZE**
Determines the number of entries in the system resource table for resource names. It must be a prime number. SSRTSIZE=37 should be adequate.

**TPRB**
Determines the number of TPRBs that NPM will generate at initialization. The maximum number of TPRBs is the number needed to support all the active requests to NPALUs, operators, and BSAM files. A quick look at the Virtual Storage Summary Display panel (FNM02OPV) will help you determine a good value for TPRBs. From this panel you can find the difference between the originally estimated count and the lowest number of free TPRBs available. This difference is the maximum number of TPRBS used, so you can use this value for the maximum number of TPRBs.

**Note:** Many problems with a shortage of TPRBs are due to a high number of NETCOLL commands executed at initialization. There are at least two TPRBs associated to each of these commands. To avoid coding one NETCOLL per resource, use the resource bucket commands, such as ALLLINES and ALLPUS.

**CDRM Statement**
You can adjust the following parameter on the CDRM statement to tune NPM performance:

**MAXL**
Limits the maximum number of LUs on which you can perform session collection on a node. Decreasing the MAXL parameter decreases the amount of CSA and private storage used.

**CONTROL Statement**
You can adjust the following parameter on the CONTROL statement to tune NPM performance:

**SMF**
Determines if the operator can use SMF control and statistics functions. Setting SMF=NO saves 4KB of CSA. You can still write NPM data to SMF if SMF=NO.
FILE Statement

You can adjust the following parameter on the FILE statement to tune NPM performance:

MACRF

Determines if NPM uses the VSAM LSR option. See “Tuning VSAM File Processing” on page 210 for information about selecting the MACRF parameter. If you select MACRF=NSR, you can improve performance by allocating more buffers on the BUFND and BUFNI parameters. The amount of private storage used is the number of buffers specified on these parameters times the CISIZE parameter. BUFND and BUFNI are not needed if you select MACRF=LSR, which is usually to your advantage because of improved VSAM performance.

HOST Statement

You can adjust the following parameter on the HOST statement to tune NPM performance:

MAXL

Limits the maximum number of local LUs on which you can perform session collection. Decreasing the MAXL parameter decreases the amount of CSA and private storage used.

NPM Statement

You can adjust the following parameters on the NPM statement to tune NPM performance:

Data Collection Parameters

There are a number of parameters on the NPM statement that control the writing of NPM records:

- CMDS
- DNC
- EVENT
- LAN
- LANX
- LWG
- LWGX
- NETWORK
- NETWORKX
- NSA
- RTM
- SESSION
- SESSIONX
- VTAM
- VTAMX

You can write each class of records to one or more destinations. Because there is overhead for each record written, you should generally only write a particular class of records to a single destination. You can also prevent the writing of entire classes of records. Be sure that you really need the data that you save.

HOSTCOLL

Determines if you collect host data. If you specify HOSTCOLL=NO, NPM will not use any CSA. However, you cannot collect session, DNC, or RTM data if HOSTCOLL=NO.
**TELNET**
Determines if you use the IP address to collect TN3270 and TN3270E session data.

**INACTVR**
Specifies whether or not, if VTAM statistics data collection is enabled, inactive virtual routes are monitored.

**INTERVAL**
Determines the length of the data collection interval. The smaller the interval, the greater the impact of NPM on the NCP and the data sets. For example, more storage is needed, response time is longer, and more I/O is required for the data sets. You should normally set your intervals to collect data every half hour or every hour. The default base interval of 225 seconds results in interval 4 beginning every half hour and interval 5 beginning every hour.

**LBRGINT**
Specifies the interval for collecting LAN bridge data. An interval range of 30 minutes to one hour is usually efficient. The default is 15 minutes. A short interval requires more storage, results in longer response time, and uses more data set I/Os. If you also specify SYNCH, you can alternate NPM collection for LAN bridge data with network and session data collection.

**LSEGINT**
Specifies the interval for collecting LAN segment data. An interval range of 30 minutes to one hour is usually efficient. The default is 15 minutes. A short interval requires more storage, results in longer response time, and uses more data set I/Os. If you also specify SYNCH, you can alternate NPM collection for LAN segment data with network and session data collection.

**MAXNCPS**
Specifies the maximum number of NCPs simultaneously known to the NPM. MAXNCPS specifies internally how many node control blocks are needed to accommodate new NCPs. Decreasing MAXNCPS decreases the storage needed.

**MAXSUBA**
Specifies the largest subarea number for any resource defined to NPM. You can set MAXSUBA to any number from 1–65535. NPM multiplies this number by four and allocates a subarea table of that size in bytes. Decreasing MAXSUBA decreases the storage needed.

**RETRY**
Enables you to increase the delay between bind attempts which limits network traffic related to NPM trying to bind NPALUs. You can also set RETRY=0 and have an operator bind NPALUs manually.

**SESSINT**
Determines the interval for logging session data. The SESSINT value is multiplied by the base interval from the INTERVAL parameter to give the logging interval. For example, to log session data every hour, use the default base interval of 225 seconds and set SESSINT=16. See the INTERVAL parameter.
SYNCH  Synchronizes network and session data collection intervals to the
system clock. To collect every hour, use the default base interval
of 225 seconds and set SYNCH=(5,60). You can use the SYNCH
parameter to alternate collection of network and session data.

Note: For VTAM statistics and NetWare resources data
collection, NPM uses the SYNCH parameter to synchronize VTAM
statistics and NetWare resources collection interval times. This
parameter is not used to synchronize VTAM statistics and
NetWare resources sampling intervals.

SAMP Statement

You can adjust the following parameters on the SAMP statement to tune NPM
performance:

SAMPVHI  Specifies the sampling interval used to collect VTAM application data,
address space data, APPN data, MNPS data, and VTAM global data.
The valid range is 60-120 seconds.

SAMPVMD  Specifies the sampling interval used to collect VTAM device data,
buffer pool, and CSM data. The valid range is 30-60 seconds.

SAMPVLW  Specifies the sampling interval used to collect VTAM virtual route data
and RTP data. The valid range is 10-30 seconds.

Note: Coding a small interval can have a significant impact on performance.

DEFAULTS Statement

You can adjust the following parameters on the DEFAULTS statement to tune NPM
performance:

VTAMINT  Specifies the default interval for VTAM statistics data collection. This
is also the interval that causes NPM to refresh the list of VTAM
resources that are available for collection. If you code
VTAMCOLL=YES, this refresh is done regardless of whether any
collections are active.

LWGINT  Specifies the default interval for NetWare resources data collection.

NGRAPHS Statement

You can adjust the following parameter on the NGRAPHS statement to tune NPM
performance:

MAXMON  Specifies the number of graphics buffers in CSA needed for real time
graphic monitoring. These buffers are not needed for graphing
archived information. Set MAXMON to the number of graphic monitor
users who will access the system at the same time. If no one will use
the real-time graphic monitor, set MAXMON=0.
SYS Statement
You can adjust the following parameter on the SYS statement to tune NPM performance:

SECURITY Determines the security processing used at logon. If you specify SECURITY=RACF, see Resource Access Control Facility: System Programming Library for the performance considerations of using RACF. To use the RACF product, you must have RACF Release 6 or later for MVS.

TASK Statement
You can adjust the following parameters on the TASK statement to tune NPM performance:

FNMSDC00 If you specify HOSTCOLL=NO on the NPM statement, you do not need the FNMSDC00 task and you should not use it.

FNMVDC00 If you specify VTAMCOLL=NO on the NPM statement, you do not need the FNMVDC00 task and you should not use it.

FNMFST00 Enables you to take advantage of multiprocessing of your I/O by coding up to six FNMFST00 tasks, one for each possible destination for your data. The cost of separating the FNMFST00 tasks is minimal. For more information about the TASK statement, see NPM Reference.

Tuning Performance Data Collection Commands
You can improve NPM’s data collection performance by adjusting the data collection commands, monitor thresholds, and collection intervals. The following sections provide details on these adjustments.

Tuning Network Collection
You can improve network performance for network collection by adjusting the NETCOLL command, the network monitor thresholds, and the network collection intervals.

NETCOLL Command
When you code a collection command, you can enter a list of resources. It is more efficient to enter more than one resource on a single collection command than it is to code multiple commands with one resource each. However, when you need to collect data on all lines, LUs, or PUs, specify the resource as ALLLINES, ALLLUS, or ALLPUS instead of listing the individual resources. However, do not collect data on a large group of resources unless it is really needed.

Network Monitor Thresholds
If you set network monitor threshold values too close to normal operating ranges, NPM generates too many event messages and records. If you set threshold values too far from normal operating ranges, NPM might not generate any event messages and records. Choose thresholds that consistently indicate a performance problem.
Tuning Performance Data Collection Commands

Network Collection Intervals
Set your base network collection interval no smaller than 225 seconds. Network tuning measurements typically use intervals in the range of 225 seconds to one hour. The shorter the collection interval, the greater the impact of NPM data collection on the NCP.

Capacity planning measurements will typically use intervals of one hour. Collection intervals larger than one hour are not recommended because a failure in NCP, VTAM, or NPM could cause collected data to be lost.

Tuning Session Collection
You will improve overall system performance by migrating to VTAM 3.4.1 or a later release.

In large networks that require large amounts of data collection, you may want to run more than one copy of NPM. One copy of NPM can collect session data from a host and you can run another copy of NPM for each NCP. Keep in mind that this approach requires more host storage, but maximizes the performance of your network.

Tuning VTAM Collection
You can improve the host performance for VTAM statistics collections by adjusting the VTAM collection (VxxxCOLL) commands, the VTAM monitor thresholds, and the VTAM collection intervals.

VxxxCOLL Commands
When you code a collection command, you can enter a list of resources. It is more efficient to enter more than one resource on a single collection command than it is to code multiple commands with one resource each. However, when you need to collect data on all buffer pools, applications, address spaces, channel-attached devices, virtual routes, RTP, or MNPS, specify the resource as ALLBP, ALLAPPL, ALLADR, ALLDEV, ALLVR, ALLRTP, or ALLMNPS, instead of listing the individual resources. Do not collect data on a large group of resources unless it is really needed.

VTAM Monitor Thresholds
If you set VTAM monitor threshold values too close to normal operating ranges, NPM generates too many event messages and records. NPM generates monitor messages and alerts when the threshold value falls outside the threshold setting. If you set threshold values too far from normal operating ranges, NPM may not generate any event messages and records. You should choose thresholds that consistently indicate a performance problem. No messages or alerts are generated when the threshold value is equal to the threshold setting.

VTAM Collection Intervals
Set your base VTAM collection interval no smaller than 225 seconds. VTAM tuning measurements typically use intervals in the range of 225 seconds to one hour. The shorter the collection interval, the greater the impact of NPM data collection on the host.

Capacity planning measurements typically use intervals of one hour. Collection intervals larger than one hour are not recommended because they could result in
the loss of collected data if a failure occurs in either VTAM or the resources managed by VTAM or NPM.

Tuning LAN Collection
The performance of LAN bridge and LAN segment collection is influenced by a number of factors, including the LAN configuration, the hardware involved, the network load, and NPM collection options. To improve the system performance, throughput, LAN bridge response times, and LAN segment response times, try the following suggestions:

1. If NPM cannot collect LAN data in the specified interval because of heavy traffic, collect on fewer resources.
2. Use a larger collection interval. In large, busy networks, some LAN Managers might not be able to respond in 5- or 10-minute intervals.
3. In the DSIPARM member FNMDSTD (MVS), set the DSRBO value equal to the number of LAN Managers that are defined to NPM.
4. If you are performing LAN data collection for multiple NPMs in the same host, it is recommended that you collect LAN data on only one NPM to avoid both slower network response time and the collection of duplicate LAN data.

Tuning NetWare Resources Data Collection
You can improve NetWare resources data collection by adjusting the LWGCOLL command, the NetWare resources monitor thresholds, and the NetWare resources collection intervals.

**LWGCOLL Command**
You must code a separate command for each resource on which you want to collect data. However, when you need to collect data on all servers, agents, or routers, specify the resource as ALLNWSRV, ALLNWAGE, or ALLNWRTR, instead of listing the individual resources. However, do not collect data on a large group of resources unless it is really needed.

If NPM cannot collect NetWare resources data in the specified interval because of heavy traffic, collect on fewer resources.

**NetWare Resources Monitor Thresholds**
If you set NetWare resources monitor threshold values too close to normal operating ranges, NPM generates too many event messages and records. If you set threshold values too far from normal operating ranges, NPM might not generate any event messages and records. Choose thresholds that consistently indicate a performance problem.

**NetWare Resources Collection Intervals**
Set your base network collection interval no smaller than 30 seconds. NetWare resources tuning measurements typically use intervals in the range of 30 seconds to one hour. The shorter the collection interval, the greater the impact on performance and network load. If you encounter performance problems, use a larger collection interval.

The performance of NetWare resources data collection is influenced by a number of other factors, including the NetWare resources configuration, the hardware
involved, the network load, and NPM collection options. To improve the system performance, throughput, and NetWare resources response times, try the following:

1. In the DSIPARM member FNMDSTD (MVS), set the DSRBO value equal to the number NPM NetWare Agents defined as collection-point file servers and configured as NPM service points. If you have also installed LAN data collection, set the DSRBO value equal to the number of NPM NetWare Agents defined as collection-point file servers and configured as NPM service points plus the number of LAN Managers that are defined to NPM.

2. If you are performing NetWare resources data collection for multiple NPMs in the same host, it is recommended that you collect NetWare resources data on only one NPM to avoid both slower network response time and the collection of duplicate NetWare resources data.

Tuning Accounting Collection
Avoid small PIU and byte thresholds. Try initial values of BYTE=1000000 and PIU=1000 on the NGAMDFY and NSAMDFY commands and adjust these values based on your network performance.

Tuning Installation-Wide Exits
Installation-wide exits add additional path length and processor load to NPM. However, you can use the NPM installation-wide exit to improve performance. For example, the NPM installation-wide exit allows you to discard unwanted data instead of writing it to your log file, thus reducing the amount of data logged and improving performance.

Tuning NPM for Large Numbers of NCPs
Dynamic configuration support enhances the usability of NPM by replacing a static configuration process with a dynamic process. In earlier releases of NPM, network configuration information was collected from resource resolution tables that you generated before starting NPM. Information is now gathered dynamically, allowing configuration changes to be made without stopping NPM. However, you might notice longer response times for NPM initialization as a result.

To initialize NPM with a large number of NCPs, follow these suggestions to improve the response time of the initialization:

- Select optimal blocking for the VTAMLST partitioned data set that contains the NCP definitions for the DASD device. This will reduce the number of I/Os required to retrieve the definitions and will make each I/O more productive. For example, for 3380 DASD, a blocking value of 9040 (for 80 column, fixed-block format) is a good initial value.

- Minimize the number of full record comments in the NCP definitions. These records are not used in the configuration process and only add to the size of the VTAMLST member that must be scanned.

- If a group of NCPs can be identified that form the basic production configuration, consider placing these NCP statements in the FNMSTRT member rather than referencing individual members in FNMSCMDS. This reduces EXEC processing for each NCP, but trades flexibility for performance.
Chapter 13. Customizing NPM Messages

Note: This chapter contains general-use programming interface and associated guidance information.

This chapter explains how to modify the text of NPM messages and how to determine where each type of message is sent. It describes how to update and edit the NPM message table and how to define NPM messages, NPM message replacement strings, and NPM message destinations. It also describes the conditions under which messages are sent to specific destinations.

The primary factor that determines message routing is the ROUTE parameter in the message definition. All NPM message definitions are in the NPM message table, which is provided in source format in the following member:

NPM.V2R4M0.SFNMSRC1(FNMMSG00)

Refer to this sample, or to the modified source if your installation has customized the message table, to find the value of the ROUTE parameter for a specific message.

The NPM installation-wide exit can change the routing of a message or discard a message. If your installation is using the NPM installation-wide exit to customize NPM processing, check the code associated with function code X'0A' in the exit to see whether it is modifying message processing.

Updating the NPM Message Table

The NPM message table contains all the messages issued by the NPM online system and for most of the batch programs. You can update the text of the message, the placement of inserted data, the routing code for the messages, and the text strings inserted into messages by editing this table. NPM provides the message table in source form in NPM.V2R4M0.SFNMSRC1(FNMMSG00).

Notes:

1. The NPM graphic subsystem does not use the NPM message table.
2. You cannot code two consecutive blanks when you customize FNMMSG00.

Defining an NPM Message (FNMMMSG)

The FNMMMSG macro defines the text of a message to be formatted. It is combined with other FNMMMSG calls to form a message module.
The following list explains each of the parameters in the FNMMSG macro:

- \texttt{MSG=message\_id}\hfill

Specifies the message number used as the key to locate the message in the message table. \texttt{message\_id} is a character string with the following format:

\begin{verbatim}
FNMnnni
\end{verbatim}

where:

- \texttt{nnnn} Is the message number. Values from 000–1100 are valid. Three digit messages do not require a leading zero. The messages defined by the FNMMSG macro must be in ascending order and the first message defined must be FNM000W because this message is issued when a message cannot be found in the table.

- \texttt{i} Is the message severity indicator. This indicator is used to set the return code of the message. Table 16 shows the six severity indicators:

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\textbf{Indicator} & \textbf{Meaning} & \textbf{Return Code} \\
\hline
RC=00 & Informational & (I) \\
\hline
RC=04 & Warning & (W) \\
\hline
RC=08 & Error & (E) \\
\hline
RC=12 & Severe & (S) \\
\hline
\end{tabular}
\caption{(Page 1 of 2). FNMMSG Message Severity Indicators}
\end{table}
Updating the NPM Message Table

You can override the value of the return code with the RC keyword.

\[\text{MSG=END} \]

Ends the message table and all the following parameters are ignored. This parameter is required as the last FNMMMSG macro in the message table.

\[\text{TEXT=('string')}\]

Specifies the text of the message as it appears to the user. The text of the message must include the message ID. The message ID is the first character string embedded in the message text. The first blank ends the ID.

NPM searches for quotes and ampersands in the text string to ensure that the text can be assembled.

Multiple text strings are concatenated into one string. Special control characters in the text of the message control the formatting of inserts and the positioning of text within the message. For example, greater than (>) and less than (<) characters in the text of a message denote the placement and formatting of an insert.

\[\text{FNMMMSG \ MSG=FM000W, \ TEXT='FM000W MESSAGE <1F> NOT FOUND'}\]

The maximum length of a formatted message after inserts have been processed is 160 characters. See “Message Inserts” on page 229 for more information about inserts.

\[\text{OPT=(BLKSUP \ HEXQUOT \ NOBLKSUP \ PRTY \ TOP)}\]

Specifies the following list of options used to format the message:

- **BLKSUP**: Specifies that the resulting message text has multiple blank characters reduced to one blank character.
- **HEXQUOT**: Specifies that hexadecimal inserts are formatted as X'xx'.
- **NOBLKSUP**: Specifies that multiple blank characters are not suppressed. This option overrides the specification of BLKSUP.
- **PRTY**: Specifies that the message queued to a terminal is queued ahead of messages that do not have OPT=PRTY.
- **TOP**: Specifies that the message is a monitor message written to a terminal, appearing on the top of the screen instead of the bottom.

---

**Table 16 (Page 2 of 2). FNMMMSG Message Severity Indicators**

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Meaning</th>
<th>Return Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC=16</td>
<td>Termination</td>
<td>(T)</td>
</tr>
<tr>
<td>RC=04</td>
<td>Action</td>
<td>(A)</td>
</tr>
</tbody>
</table>

You can override the value of the return code with the RC keyword.
Updating the NPM Message Table

Specifies the routing of this message with the following options:

**ALL**
Specifies that you want NPM to route the message to all active terminals and to the system console.

**CONSOLE**
Specifies that you want NPM to route the message to the system console. If you specify the CNSID keyword, set MCSFLAG=(REG0) to route the message to the console.

**EXTERNAL**
Specifies that you want NPM to route the message to the NPM operator’s external ID when the NPM operator is not logged on to NPM. See *NPM Reference* for information on how to enable sending messages to an external ID.

**GLOBAL**
Specifies that you want NPM to route the message to all NPM operators whose security profile specifies GLOBAL=YES. If such an operator is not logged on, write the message to the system console.

**NOCNSL**
Specifies that you do not want NPM to route the message to the system console.

**NOGTERM**
Specifies that you do not want NPM to route messages related to I/O errors to the NPM operator terminal. The message is sent to the system console.

**PRINT**
Specifies that you want NPM to route the message to the SYSPRINT log data set. Messages routed to a global terminal operator or to the console are always routed to PRINT.

**TERM**
Specifies that you want NPM the route the message to the NPM operator terminal.

Note: TERM message destination routing applies only to NPM 3270 operator terminals. Messages will continue to be sent to NPM Desk/2 workstations, even if you do not code TERM on the ROUTE parameter.

See “Writing Messages to the System Console” on page 232 for more information about determining the destination of a message.

**Note:**

1. MVS Only
Updating the NPM Message Table

Specifies the following write-to-operator (WTO) routing codes assigned to the message:

1  Indicates master console action
2  Indicates master console information
3  Indicates tape pool
4  Indicates direct access pool
5  Indicates tape library
6  Indicates disk library
7  Indicates unit record pool
8  Indicates teleprocessing control
9  Indicates system security
10 Indicates system error or maintenance
11 Indicates programmer information
12 Indicates emulators
13-15 Indicates reserved for customer use
16 Indicates reserved for future expansion

Use ROUTCDE=8 for WTO messages. See the appropriate MVS manual for more information about routing codes.

Note: ROUTCDE applies only to messages that appear on the system console. If a message is routed to the system console, but does not specify ROUTCDE, NPM uses ROUTCDE=8.

```
┌┐ ──,DESC=7 ────────────────────────
/SM590000─ ──┼ ┼───────────────────────────────── ─/SM590000
│ │ ┌┐─,───────────────
└┘ ──,DESC──
(1) = ───/SV040000 ┴─

descriptor_code

Note:
1  MVS Only
```

Specifies the following WTO message descriptor codes assigned to the message:

1  Indicates system failure
2  Indicates immediate action required
3  Indicates eventual action required
4  Indicates system status
5  Indicates immediate command response
6  Indicates job status
7  Indicates application program or processor
8  Indicates out-of-line message
9  Indicates operator request
Updating the NPM Message Table

10 Indicates dynamic status displays
11 Indicates critical eventual action required
12-16 Indicates reserved

Descriptor codes 1 through 6 and 11 are mutually exclusive. You can code 7 through 10 in combination with any other code. Use DESC=(7) for WTO messages.

Note: DESC applies only to messages that appear on the system console. If a message is routed to the system console, but does not specify DESC, NPM uses DESC=7.

Note: 1 MVS Only

Specifies that the macro expansion should set bits in the MCSFLAG field as indicated by each name coded. The flag names and meanings, including write-to-operator with reply (WTOR), are shown in the following list:

BRDCST Specifies that you want NPM to broadcast the message to all active consoles.
HRDCPY Specifies that you want NPM to queue messages for hard copy only.
NOCPY Specifies that you do not want NPM to queue messages for hard copy if the WTO or WTOR macroinstruction is issued by a program in supervisor state. Otherwise, ignore this parameter.
NOTIME Specifies that you do not NPM to append time to messages.
QREG0 Specifies that you want NPM to queue messages unconditionally to the console corresponding to the source ID passed in register 0.
REG0 Specifies that you want NPM to queue messages to the console corresponding to the source ID passed in register 0.
REPLY Specifies that you want NPM to reply to a WTOR.
RESP Specifies that you want NPM to respond to the command immediately.

RC=nnn

Specifies a return code received by the caller when the message is formatted. This return code can be a decimal value from 0–255. It replaces the default value taken from the MSG keyword.
Message Inserts

The caller of the message formatter provides a list of inserts included in the message. The text of the message contains control parameters indicating where in the text the insert is placed, which insert is formatted, and how the insert is formatted. Greater than (>) and less than (<) characters indicate a control parameter. The general format of a control parameter is as follows:

\(<NT>\)

The following list explains each possible control parameter:

where:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Specifies which insert is placed into the text of the message. 1 is the first insert passed by the caller, 2 is the second, and so on. If N is not specified, the next insert number is used. Inserts can be positioned in any order or repeated any number of times. For example, the message can specify that insert 3 come before insert 2. It is not required that all inserts passed by the FNMOPMSG caller be used. None, any, or all can be used.</td>
</tr>
<tr>
<td>T</td>
<td>Specifies the format of the insert placed in the text of the message. If not specified, character data is assumed. The following types of data are supported:</td>
</tr>
<tr>
<td></td>
<td>• C - Character data</td>
</tr>
<tr>
<td></td>
<td>The insert data is moved into the message text according to the insert parameter length. The maximum length of insert data is sixty-four characters.</td>
</tr>
<tr>
<td></td>
<td>• D - Date string</td>
</tr>
<tr>
<td></td>
<td>The insert data is used to format a date in the text of the message. The format is a packed decimal fullword of the form: yyymmddf. The J insert type must be specified before this type to control the format of the date string in the message. The length of insert data must be four characters.</td>
</tr>
<tr>
<td></td>
<td>• F - Signed binary fullword</td>
</tr>
<tr>
<td></td>
<td>The insert data is a binary fullword converted to a decimal number. Leading zeros are converted to blanks. If the number is negative, a minus sign is inserted before the first significant digit. The maximum length of insert data is four characters.</td>
</tr>
<tr>
<td></td>
<td>• H - Signed binary halfword</td>
</tr>
<tr>
<td></td>
<td>The insert data is a binary halfword converted to a decimal number. Leading zeros are converted to blanks. The maximum length of insert data is four characters.</td>
</tr>
<tr>
<td></td>
<td>• I - Time string</td>
</tr>
<tr>
<td></td>
<td>The insert data is used to format a time in the text of the message. The format is a packed decimal fullword of the form: hhmmssF. The time is formatted as hh.mm.ss. The length of the insert data must be four characters.</td>
</tr>
</tbody>
</table>
Updating the NPM Message Table

- **J - Date format control variable**
  The insert data is a binary fullword or halfword that contains a value from 1–3. This value corresponds to the DATEF parameter of the DEFAULTS initialization statement and controls the format of the date string. The maximum length of insert data is four characters. This insert does not cause any data to be formatted in the message. It only provides the message formatter with a means of controlling the format of the date.

  **Note:** Although it is not important which insert provides the format code, the J specification must appear before the first D specification in the message.

- **S - String replacement**
  The insert data is used to locate a replacement string defined by the FNMMRS macro. The insert data is a binary fullword or halfword that contains a value from 1–65535. This value corresponds to the FNMMRS string number. The maximum length of insert data is four characters.

- **T - Tab to offset**
  Blanks are inserted in the message text to the column defined by the N parameter, unless the column has already been passed. The maximum tab column is 128.

  **Note:** OPT=NOBLKSUP must be specified for a message that uses this control parameter.

- **X - Hexadecimal data**
  The insert data is converted to printable EBCDIC and moved into the message text. Each insert byte converts to two bytes of message text. The maximum length of insert data is four characters.

Figure 56 provides example messages that include these parameters.

```assembly
FNMSGO00 CSECT
  FNMSGG MSG=FNM000W, C
  TEXT='FNM000W MESSAGE <1F> NOT DEFINED', C
  RC=4
  FNMSGG MSG=FNM008E, C
  TEXT='FNM008E NCP <1C> HAS NOT BEEN DEFINED TO NPM', C
  ROUTCDE=8,DESC=7,RC=4
  FNMSGG MSG=FNM0251, C
  TEXT='FNM0251 <3J> <1D> <2I> INITIALIZATION COMPLETE', C
  ROUTCDE=8,DESC=7,ROUTE=ALL
  FNMSGG MSG=FNM0BBW, C
  TEXT='FNM0BBW VTAM EXCEPTION FOR LU <1C>. REQ=<2X>,RTNCDC ==<3X>,SENSE=<4X>', C
  ROUTCDE=8,DESC=2
  FNMSGG MSG=FNM113W, C
  TEXT=('FNM113W EXECUTE COMPLETED WITH <1S>, MEMBER=<2C>')
  FNMSGG MSG=END
END
```

Figure 56. Example of Message Inserts
Defining an NPM Message Replacement String (FNMMRS)

The FNMMRS macro defines the text of a message replacement string.

```
label FNMMRS TYPE=START
nr, 'string'

END
```

The following list describes the parameters used in the FNMMRS macro:

- **label**: Specifies the name of the control section. You must specify a label with TYPE=START.

- **nr**: Specifies the string replacement number. This number is a key to locate the string. Each invocation of the FNMMRS macro defining a replacement string must specify a number. The numbers in each FNMMRS macro must be in ascending sequence. Duplicate values are not allowed. The value can be from 1–65535.

- **'string'**: Specifies the replacement character string enclosed in quotes. The maximum length of the string is 255 characters. However, the maximum length of a message after all inserts are moved into the text of the message is 160 characters.

- **,TYPE=START**: Specifies the beginning of the FNMMRS macro specifications. The first invocation of the macro must specify TYPE=START and a label.

The first FNMMRS macro in the table should be the text returned when an entry cannot locate the message replacement string table, as shown in Figure 57.

```
FNMMRS00 CSECT
   FNMMG MSG=FNMM0081,
       TEXT='FNMM0081 RESOURCE <1C> IS <2S>'
   .
   .
   FNMMG MSG=END

FNMMRS00 FNMMRS TYPE=START
   FNMMRS 1,'* - NOT FOUND - *'
   FNMMRS 44,'ACTIVE'
   FNMMRS 45,'INACTIVE'
   FNMMRS TYPE=END
```

*Figure 57. Example of FNMMRS Macroinstruction*
Writing Messages to the System Console

This section describes how to determine if a message is written to the system console. If you are using NPM messages with console automation, you need to be sure that the messages you are automating from are always written to the system console. You must also specify the CNSID parameter of the CONSOLE initialization statement in the FNMINIT data set with a value greater than zero to enable the sending of messages to the system console.

The following values are used on the ROUTE parameter:

**CONSOLE**

Specifies that messages are always written to the system console. For MVS systems, you can also use the ROUTCDE and DESC parameters in the message definition to use the write-to-operator (WTO) function for messages written to the system console.
GLOBAL  Specifies that all messages are written to the system console when no operators with GLOBAL=YES specified in their security profile are logged on to NPM.

NOGTERM  Specifies that messages caused by I/O errors in the session between an NPM operator and NPM are written to the system console.

ALL  Specifies that all messages are written to the system console.

NOCNSL  Specifies that messages are never written to the system console. For example, if a message specifies ROUTE=(GLOBAL,NOCNSL), the message is not written to the system console even if no operator is logged on to NPM.

Writing Messages to the SYSPRINT Data Set

This section describes how to determine if a message is written to the SYSPRINT data set. The SYSPRINT data set is defined in the NPM startup JCL.

The following values are used on the ROUTE parameter:

PRINT  Specifies that messages are written to the SYSPRINT data set, except for error messages related to NPM initialization. Initialization messages are written to the FNMILOG data set.

GLOBAL  Specifies that messages are written to the SYSPRINT data set when no operators with GLOBAL=YES specified in their security profile are logged on to NPM.

NOGTERM  Specifies that messages caused by I/O errors in the session between an NPM operator and NPM are written to the SYSPRINT data set.

ALL  Specifies that all messages are written to the SYSPRINT data set.

TERM  Specifies that messages are written to the SYSPRINT data set when they are the result of a command issued by an NPM EXEC. An NPM EXEC acts like an operator issuing the commands that make up the EXEC. When NPM returns a message to the EXEC, the EXEC writes the message to the SYSPRINT data set.

For example, if the EXEC contained a NETCOLL command, NPM returns message FNM275I to the EXEC to indicate that the collection has started. The EXEC then writes that message to the SYSPRINT data set. However, if the collection generates an exception message after the EXEC has completed, then the collection message FNM200I is routed to the operator who issued the EXEC, based on the default of ROUTE=TERM for FNM200I.
Sending Messages to the Operator

**Sending Messages to an NPM Operator Logged on to NPM**

This section describes how to determine if a message is sent to an operator logged on to NPM. As a general rule, messages that result from commands entered by a particular operator are displayed for that operator only.

**Note:** A host operator cannot send messages to an NPM Desk/2 workstation, and an NPM Desk/2 operator cannot send messages to the host.

The following values are used on the ROUTE parameter:

**TERM** Specifies that messages are sent to the NPM operator determined by NPM. Messages that result from an operator entering a command such as TIME or WHO are always sent to the operator who entered the command. Error messages that result from incorrect entry on a panel are always sent to the operator who made the incorrect entry.

If ROUTE=TERM for a network monitor (TERMINAL was selected as a destination), monitor exception and monitor resolution records are sent to the operator who started the monitor online. If the monitor was started by a NETCOLL command, the messages are sent to the operator named by the XROUTE parameter of the command. However, LAN monitor messages are not sent in the same manner. Even though the message specifies ROUTE=TERM, it is overridden as ROUTE=GLOBAL when issued. Therefore, changes in the routing of monitor event messages affect only network monitors.

Messages that result from the online SEND MESSAGE panel are sent to the operator named in the User ID field on that panel.

**ALL** Specifies that messages are sent to all NPM operators logged on to NPM.

**Sending Messages to an NPM Operator with Global Authority**

This section describes how to determine if a message is sent to an operator with global authority. An operator with global authority is an operator whose security profile specifies GLOBAL=YES. If an operator has more than one security profile defined, GLOBAL=YES must be specified in the profile that the operator used to log on to NPM.

Global authority enables an operator to receive additional messages. An operator with global authority also receives messages based on the rules for an operator without global authority.

The following values are used on the ROUTE parameter:

**GLOBAL** Specifies that messages are written to all operators with global authority who are logged on to NPM when the message is issued. If no operator with global authority is logged on to NPM when a message is issued, the message is written to the system console.
NOGTERM Specifies that messages caused by I/O errors in the session between an NPM operator and NPM are written to all operators with global authority.

ALL Specifies that messages are written to all operators with global authority.

Sending Messages to an NPM Operator Not Logged on to NPM

This section describes how to determine if a message is sent to an external operator. External operators are NPM operators who are not logged on to NPM but who are logged on to TSO or CMS on the same host.

To enable sending messages to an external operator, the OPERATOR statement in the FNMOOPER data set for the operator must specify EXTERNAL=Yes. The NPM operator ID specified on the NAME parameter of the OPERATOR statement must be the same as the TSO or CMS user ID. The following value is used on the ROUTE parameter:

EXTERNAL Specifies that messages are sent to the external TSO or CMS user ID if the NPM operator is not logged on to NPM but is logged on to TSO or CMS. The rules for determining which external user ID NPM sends the message to are the same as the rules for determining which NPM operator ID messages are sent to. See “Sending Messages to an NPM Operator Logged on to NPM” on page 234.

Sending Alerts to the NetView Program

Alerts sent to the NetView program are not considered messages by NPM, although they are closely related. The creation of network alerts is controlled by the network monitor function and the NETCOLL command. The creation of LAN alerts is controlled by the LAN bridge and LAN segment monitor function, the LBRGCOLL command, the LSEGCOLL command, and session alerts. NetWare resources data exception records are formatted into alerts and sent to NetView, if DESTX=ALERT is coded on the LWGCOLL command and ALERT=YES is coded on the NPM statement in FNMINIT. See NPM User’s Guide for more information about these commands.

Message Table Changes for NPM V2R4

If you want to reuse a customized message table from an earlier release of NPM, you must update your table with the new and changed messages for this release, which are listed in Table 17 on page 236.
**Table 17. New and Changed Messages for NPM V2R4**

<table>
<thead>
<tr>
<th>Message Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNM340W</td>
</tr>
<tr>
<td>FNM342E</td>
</tr>
<tr>
<td>FNM1200I</td>
</tr>
<tr>
<td>FNM1201I</td>
</tr>
</tbody>
</table>

See *NPM Messages and Codes* for information about these new messages.
Chapter 14. Customizing NPM Processing with Installation-Wide Exits

**Note:** This chapter contains general-use programming interface and associated guidance information.

You can change or extend the way NPM works by using NPM's installation-wide and path information unit (PIU) analysis exits. These exits enable your code to receive control during NPM processing. By coding different exit functions, you can change the way NPM processes data to address the specific needs of your installation. The NPM installation-wide exit is a general purpose exit that you can use to perform special processing of NPM data. The PIU analysis exit is used by NPM's session collection function to determine the type of PIUs that are flowing through the network.

The NPM installation-wide exit and PIU analysis exit are optional. You should only code them if you need the functions they provide.

This chapter describes the functions provided by NPM's installation-wide and PIU analysis exits and explains when they are called and the parameters that are passed.

**The NPM Installation-Wide Exit**

The NPM installation-wide exit enables your code to receive control during any of the following events that occur during NPM operation:

- Start and stop of NPM
- Logon and logoff of NPM operators
- Connect and disconnect of NPALUs
- Connect and disconnect of remote NPMs
- Dynamic network collection (DNC) authorization
- NPM message processing
- The writing of records to output data sets

When executing under MVS/XA, the exit routine is called in a supervisor state, in key 6, and in AMODE of 31, because the parameter list and some parameters can reside above the 16-megabyte line. Processing or program errors in the exit routine can affect the performance and data integrity of NPM and the operating system. No assumption should be made about synchronization of control when the exit routine is called. You must code the exit routine so that it is reentrant.

**Register Contents**

The following list describes the register contents when NPM passes control to the NPM exit routine:

**Register 1** Address of a variable-length list of virtual storage addresses that point to fields of parameter data. The end of the parameter list is indicated by a 1 in the high-order bit of the last word.

**Register 13** Address of an 18-fullword save area.

**Register 14** Return address.
Register 15  Address of the entry point of the routine.

All registers should hold their original contents upon return from the exit routine, except register 15 which holds a return code. This return code varies for each function. The sections pertaining to the individual functions later in this chapter list the possible contents of register 15 when NPM regains control.

NPM Installation-Wide Exit Functions

The NPM installation-wide exit provides the following functions:

- Begin
- End
- Operator connect
- Operator disconnect
- NPALU connect
- NPALU disconnect
- NPM connect
- NPM disconnect
- DNC authorization
- File switch
- Message
- NPMLOG record
- VTAMLOG record
- SESSION record
- REVIEW record
- SMF record
- Performance alert
- NetWare record
- Session interval analysis
- PIU analysis

The functions you use depend on the needs and procedures of your installation. The PIU analysis function is covered in “The PIU Analysis Exit” on page 259, it is discussed separately as it can be used as a separate exit function. The begin function is the only required function. All other functions are optional.

Begin Function (Function Code X'FE')

The begin function selects the other functions that are processed by updating the bits in the exit options field (named BFLAG1 in sample FNMUEXIT) as shown in Table 40 on page 254. This function is called only once, when NPM is started, before any other function is processed. The begin function is required. If the exit routine is not called at this time, it is not invoked for the other functions.
Table 18 shows the begin parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 18. Begin Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the exit options</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the installation-wide exit parameter field (UPARM)</td>
</tr>
</tbody>
</table>

The begin function routine should return a return code of 0 in register 15. If it returns another return code, or incorrect data as a result of processing the begin function, NPM continues as if no exit routine exists and issues message FNM966S to the operator.

End Function (Function Code X’FF’)

The end function is called when NPM ends. This function performs any cleanup required and returns control to NPM. The end function is not processed if NPM abnormally ends.

Table 19 shows the end parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 19. End Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the exit options</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the installation-wide exit parameter field (UPARM)</td>
</tr>
</tbody>
</table>

The end function routine should return a return code of 0 in register 15. If it returns another return code, NPM continues ending and issues message FNM966S to the operator. NPM termination hangs if the end function routine does not complete. In this case, the operator must issue the MVS CANCEL command.

Operator Connect Function (Function Code X’01’)

The operator connect function is called to verify input parameters each time an NPM operator requests a logon. At this time, the operator ID, password, and profile have not yet been verified. The operator connect function can inspect values in the user security area and change them.
Table 20 shows the logon parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

**Table 20. Logon Function Parameter List**

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the operator user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the user security area</td>
</tr>
</tbody>
</table>

The logon function routine should return one of the following return codes in register 15:

0  The session is authorized. The operator ID, password, and profile names are checked to allow the operator to use NPM.

4  The session is not authorized. NPM issues a message indicating that the session is not authorized. The operator can then change the user or profile names and try to log on again (until exhausting the maximum number of logon attempts allowed).

If the routine returns another return code, the session is authorized and NPM issues message FNM966S to the operator.

**Operator Disconnect Function (Function Code X'02')**

The operator disconnect function is called when an NPM operator logs off NPM.

Table 21 shows the logoff parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

**Table 21. Logoff Function Parameter List**

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the operator user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF data area (SMF28OGF)</td>
</tr>
</tbody>
</table>

The logoff function routine should return a return code of 0 in register 15. If it returns another return code, the session is logged off and NPM issues message FNM966S to the operator.
NPALU Connect Function (Function Code X'03')
The NPALU connect function is called when an NPALU connects successfully with NPM.

Table 22 shows the NPALU connect parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of a zero user word</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF data area (SMF28NNC)</td>
</tr>
</tbody>
</table>

The NPALU connect function routine should return a return code of 0 in register 15. If it returns another return code, the session continues and NPM issues message FNM966S to the operator.

NPALU Disconnect Function (Function Code X'04')
The NPALU disconnect function is called when an NPALU is disconnected from NPM.

Table 23 shows the NPALU disconnect parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of zero user word</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF data area (SMF28NND)</td>
</tr>
</tbody>
</table>

The NPALU disconnect function routine should return a return code of 0 in register 15. If it returns another return code, the session is disconnected and NPM issues message FNM966S to the operator.
NPM Connect Function (Function Code X'05')
The NPM connect function is called when a remote NPM connects with the local NPM.

Table 24 shows the NPM connect parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 24. NPM Connect Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of a zero user word</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF data area (SMF28NPC)</td>
</tr>
</tbody>
</table>

The NPM connect function routine should return a return code of 0 in register 15. If it returns another return code, the session is started and NPM issues message FNM966S to the operator.

NPM Disconnect Function (Function Code X'06')
The NPM disconnect function is called after a session with a remote NPM ends.

Table 25 shows the NPM disconnect parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 25. NPM Disconnect Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of a zero user word</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF data section (SMF28NPD)</td>
</tr>
</tbody>
</table>

The NPM disconnect function routine should return a return code of 0 in register 15. If it returns another return code, the session is ended and NPM issues message FNM966S to the operator.

DNC Authorization Function (Function Code X'07')
After a session between two logical units is started or ended, NPM is posted with information about the session. If both logical units are defined with DNC=YES and the secondary logical unit is defined in an NCP, the DNC authorization function is called to authorize the starting or stopping of network data collection.
Table 26 shows the DNC parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 26. DNC Session Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the SMF session management area (SMF28SMA)</td>
</tr>
</tbody>
</table>

The DNC session function routine should return one of the following return codes in register 15:

0  The collection of network data is started or stopped for DNC, as appropriate.
4  The collection of network data is not started or stopped for DNC.

If the routine returns another return code, DNC collection is started or stopped and NPM issues message FNM966S to the operator.

File Switch Function (Function Code X'09')

The file switch function is called when a file is closed that was being used as an output data set. This function informs you that the file is ready for processing.

A secondary function code is also available for this function. It indicates whether NPM is ending. See Table 39 on page 253 for more information about this code.

Table 27 shows the switch file parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 27. File Switch Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the DD name</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Address of the data set name</td>
</tr>
<tr>
<td>24(18)</td>
<td>4</td>
<td>Address of the volume serial</td>
</tr>
</tbody>
</table>

The file switch function routine should return a return code of 0 in register 15. If it returns another return code, the switch function is processed and NPM issues message FNM966S to the operator.
Message Function (Function Code X'0A')
The message function is called when a message is formatted for the online system. This function can discard the message, inspect or change the text of the message, or modify the routing of the message.

Table 28 shows the message parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the operator user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the message parameter list</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Address of the write-to-operator (WTO) descriptor and routing codes</td>
</tr>
</tbody>
</table>

The message function routine should return one of the following return codes in register 15:

0  The message is processed.
4  The message is discarded.

If the routine returns another return code, the message is processed and NPM issues message FNM966S to the operator.

NPMLOG Record Function (Function Code X'10')
The NPMLOG record function is called before a record is written to one the FNMLOGx data sets. It enables you to modify or discard records before they are written to the data set.

Table 29 shows the NPMLOG record parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF record header (SMF28RCD)</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Length of the SMF record header</td>
</tr>
</tbody>
</table>
The NPMLOG record function routine should return one of the following return codes in register 15:

0  The record is written to the FNMLOGx data set
4  The record is not written

If the routine returns another return code, the record is written and NPM issues message FNM966S to the operator.

VTAMLOG Record Function (Function Code X'11')
The VTAMLOG record function is called before a record is written to one of the FNMVLOGx data sets. It enables you to modify or discard records before they are written to the data set.

Table 30 shows the VTAMLOG record parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the VTAMLOG data record (FNMGTF)</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Length of the VTAMLOG record header</td>
</tr>
</tbody>
</table>

The VTAMLOG record function routine should return one of the following return codes in register 15:

0  The record is written to the FNMVLOGx data set
4  The record is not written

If the routine returns another return code, the record is written and NPM issues message FNM966S to the operator.

Session Record Function (Function Code X'12')
The session record function is called before a record is written to one of the FNMSESx data sets. It enables you to modify or discard records before they are written to the data set.

Table 31 shows the session record parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
</tbody>
</table>
The session record function routine should return one of the following return codes in register 15:

0  The record is written to the FNMSESx data set
4  The record is not written

If the routine returns another return code, the session record is written and NPM issues message FNM966S to the operator.

Review Record Function (Function Code X’13’)
The review record function is called before a record is written to one of the FNMREVx data sets. It enables you to modify or discard records before they are written to the data set.

Table 32 shows the review record parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

The review record function routine should return one of the following return codes in register 15:

0  The record is written to the FNMREVx data set
4  The record is not written

If the routine returns another return code, the review record is written and NPM issues message FNM966S to the operator.

SMF Record Function (Function Code X’14’)
The SMF record function is called before a record is written to the SMF log. It enables you to modify or discard records before they are written to SMF.
Table 33 shows the SMF parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 33. SMF Record Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the SMF record header (NPMLOG)</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Length of the SMF record header</td>
</tr>
</tbody>
</table>

The SMF record function routine should return one of the following return codes in register 15:

0  The record is written to SMF
4  The record is not written

If the routine returns another return code, the record is written and NPM issues message FNM966S to the operator.

Performance Alert Function (Function Code X'15')

The performance alert function is called before each alert is sent to the NetView program over the NetView program-to-program interface. It enables you to modify or discard alerts before they are sent to the NetView program.

Table 34 shows the performance alert parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 34. Performance Alert Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the alert NMVT</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Address of the length of the alert NMVT</td>
</tr>
</tbody>
</table>

The performance alert function routine should return one of the following return codes in register 15:

0  The alert is sent to the NetView program
4  The alert is discarded

If the routine returns another return code, the alert is sent and NPM issues message FNM966S to the operator.
Installation-Wide Exits

NetWare Record Function (Function Code X'16')
The NetWare record function is called before a record is written to one of the FNMLWGx data sets. It enables you to modify or discard records before they are written to the data set.

Table 35 shows the NetWare record parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 35. NetWare Record Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the file user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the VSAM NetWare record (FNMKVSM)</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Length of the VSAM record header</td>
</tr>
</tbody>
</table>

The NetWare record function routine should return one of the following return codes in register 15:

0 The record is written to the FNMLWGx data set
4 The record is not written

If the routine returns another return code, the NetWare record is written and NPM issues message FNM966S to the operator.

Session Interval Analysis (Function Code X'18')
The session interval analysis function is called for each LU-LU session pair at the end of every session interval. The function can inspect the data collection for the interval, change it, and decide whether the data collected should be written or discarded.

Table 36 shows the session analysis parameter list pointed to by register 1. The parameters are described in “Parameter Descriptions” on page 249.

Table 36. Session Analysis Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the session user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the session transit data (SMF28STT)</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Address of the session volume data (SMF28SAA)</td>
</tr>
</tbody>
</table>
Installation-Wide Exits

The session analysis function routine should return one of the following return codes in register 15:

0  The session data is written and summarized
4  The session data is not written or included in any summary records

If the routine returns another return code, the session data is written and NPM issues message FNM966S to the operator.

Parameter Descriptions

The parameters NPM passes to the installation-wide exit routine vary depending on the exit routine’s function. However, the basic format of the parameter list is the same for all functions. For example, input for each function appears at the same location in each parameter list. The following sections describe the possible parameters that may be passed to the installation-wide exit functions. See the sections pertaining to the individual functions for the specific parameters that apply to each function.

Environment Vector List

The address of the environment vector list is the first parameter of the function parameter list for all function codes. These vectors, which are preceded by a 2-byte field, make it possible for your installation to use a single exit routine, installed in each host, to collect data.

The environment vector list uses the same format as the resource identification vector list described in VTAM Programming. The format is as follow:

<table>
<thead>
<tr>
<th>Byte</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 and 1</td>
<td>Total length of the vector list (including bytes 0 and 1)</td>
</tr>
<tr>
<td>2 through m</td>
<td>First vector</td>
</tr>
<tr>
<td>m+1 through n</td>
<td>Second vector</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>... through x</td>
<td>...</td>
</tr>
<tr>
<td>x+1 through y</td>
<td>Last vector</td>
</tr>
</tbody>
</table>

The installation-wide exit searches the environment vector list to find a particular vector. Table 37 lists the vectors for each NPM environment.

Table 37 (Page 1 of 4). NPM Environment Vectors

<table>
<thead>
<tr>
<th>Key (Hex)</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>4</td>
<td>VTAM release level vector, which contains the following information:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Product code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Version code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Release code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modification code</td>
</tr>
<tr>
<td>02</td>
<td>8</td>
<td>VTAM application network name</td>
</tr>
<tr>
<td>03</td>
<td>8</td>
<td>VTAM application ACBNAME name</td>
</tr>
<tr>
<td>04</td>
<td>14</td>
<td>VTAM component vector</td>
</tr>
<tr>
<td>05</td>
<td>10</td>
<td>VTAM function list vector</td>
</tr>
</tbody>
</table>
### Table 37 (Page 2 of 4). NPM Environment Vectors

<table>
<thead>
<tr>
<th>Key (Hex)</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>06</td>
<td>8</td>
<td>VTAM network ID vector</td>
</tr>
<tr>
<td>07</td>
<td>8</td>
<td>VTAM SSCP name vector</td>
</tr>
<tr>
<td>08</td>
<td>8</td>
<td>VTAM host PU vector</td>
</tr>
<tr>
<td>09</td>
<td>4</td>
<td>VTAM host subarea vector</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>NPM release level vector, which contains the following information:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Product code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Version code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Release code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modification code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value for NPM Version 2 is “5220”.</td>
</tr>
<tr>
<td>14</td>
<td>10</td>
<td>NPM component ID vector</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This vector has the values 5655-04301 for MVS/XA or MVS/ESA.</td>
</tr>
</tbody>
</table>
## Table 37 (Page 3 of 4). NPM Environment Vectors

<table>
<thead>
<tr>
<th>Key (Hex)</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>6</td>
<td>NPM function list vector</td>
</tr>
</tbody>
</table>

The first byte has the following bits defined:
- 1... NCP bind correlation
- .1.. NPM-to-NPM support
- ..1. Multiple operating system support
- ...1 Dynamic network collection
- .... 1... Network session accounting
- ..... .1.. Gateway session accounting
- ..... ..1. NCP dynamic reconfiguration
- ..... ...1 Extended network addresses supported for network collection

The value for NPM Version 2 is B'11111111'.

The second byte has the following bits defined:
- 1... Dynamic definite response
- .1.. Extended WTO routing codes
- ..1. Network performance alerts
- ...1 Session manager support
- .... 1... NTRI network data collection support
- ..... .1.. NPSI network data collection support
- ..... ..1. Generic NEO data collection support
- ..... ...1 NPM-NPM unbind support

The value for NPM Version 2 is B'11111111'.

The third byte has the following bits defined:
- 1... Dynamic configuration support
- .1.. LAN bridge collection
- ..1. Extended console commands
- ...1 VTAM performance interface
- .... 1... Frame relay DTE resource support
- ..... .1.. Session alert support
- ..... ..1. Frame relay DCE resource support
- ..... ...1 Ethernet resource support

The value for NPM Version 2 is B'11111111'.

The fourth byte has the following bits defined:
- 1... ODLC resource support
- .1.. LAN segment collection support
- ..1. LU 6.2 support
- ...1 VTAM statistics support
- .... 1... Enhanced unbind support
- ..... .1.. NetWare resources support
- ..... ..1. Reserved
- ..... ...1 Reserved

The value for NPM Version 2 is B'11110000'.

16 8 NPM FMID/ID vector

The value for NPM Version 2 is “HNR2220” for MVS.

18 8 NPM host name vector

This vector is taken from the HOST initialization statement.
Installation-Wide Exits

Table 37 (Page 4 of 4). NPM Environment Vectors

<table>
<thead>
<tr>
<th>Key (Hex)</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
</table>
| 19        | 16           | NPM location name vector  
This vector is taken from the NAME keyword of the VTAM initialization statement. |
| 1B        | 4            | NPM time zone value  
This value is set using Greenwich mean time. It is the number of microseconds in store-clock (STCK) format (first word only). |
| 1C        | 1            | NPM operating system ID  
This vector has the value X for MVS/XA and MVS/ESA |
| 1D        | 4            | NPM TP buffer size  
This vector has a value of 2188 bytes. |
| 1E        | 8            | NPM system operator ID  
This vector is taken from the OPERATOR parameter of the CONSOLE initialization statement. |
| 1F        | 8            | NPM procedure name  
MVS This vector is the procedure name that is used in the MODIFY or STOP commands for NPM. |

The following list shows the format of each vector in Table 37 on page 249.

**Byte** | **Contents**
---|---
0 | Length of the vector (including byte 0)
1 | ID of the vector
2 through p | Vector data

See VTAM Programming for more information about the VTAM environment vectors.

**Exit Routine Function Codes**
The exit routine function code is the second parameter in the function parameter list for all functions. This code determines the exit routine functions that are processed. The first byte of the function code is the primary function code. All parameters passed to the exit routine depend on this code. The second byte is the secondary function code or a flag field. The NPM installation-wide exit can use the secondary function code to help determine the appropriate action to take.
Table 38 lists the possible primary function codes. Table 39 lists the primary function codes with secondary function codes.

**Table 38. Exit Routine Function Code**

<table>
<thead>
<tr>
<th>Byte</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Primary function code</td>
</tr>
<tr>
<td>X'FE'</td>
<td>Begin function. The exit routine has been called to select the functions it will process. This function is processed only once.</td>
</tr>
<tr>
<td>X'FF'</td>
<td>End function. The exit routine has been called to perform required cleanup when NPM ends. This function is processed only once when NPM ends.</td>
</tr>
<tr>
<td>X'01'</td>
<td>The operator connect function.</td>
</tr>
<tr>
<td>X'02'</td>
<td>The operator disconnect function.</td>
</tr>
<tr>
<td>X'03'</td>
<td>The NPALU connect function.</td>
</tr>
<tr>
<td>X'04'</td>
<td>The NPALU disconnect function.</td>
</tr>
<tr>
<td>X'05'</td>
<td>The NPM connect function.</td>
</tr>
<tr>
<td>X'06'</td>
<td>The NPM disconnect function.</td>
</tr>
<tr>
<td>X'07'</td>
<td>The dynamic network collection session authorization function.</td>
</tr>
<tr>
<td>X'09'</td>
<td>The file switch function.</td>
</tr>
<tr>
<td>X'0A'</td>
<td>The message function.</td>
</tr>
<tr>
<td>X'10'</td>
<td>The NPMLOG record output function.</td>
</tr>
<tr>
<td>X'11'</td>
<td>The VTAMLOG record output function.</td>
</tr>
<tr>
<td>X'12'</td>
<td>The SESSION record output function.</td>
</tr>
<tr>
<td>X'13'</td>
<td>The REVIEW record output function.</td>
</tr>
<tr>
<td>X'14'</td>
<td>The SMF record output function.</td>
</tr>
<tr>
<td>X'15'</td>
<td>The performance alert output function.</td>
</tr>
<tr>
<td>X'16'</td>
<td>The NetWare record output function.</td>
</tr>
<tr>
<td>X'18'</td>
<td>The session interval analysis function.</td>
</tr>
<tr>
<td>X'19'</td>
<td>The PIU analysis function (TRANSIT=DR).</td>
</tr>
<tr>
<td>X'1A'</td>
<td>The PIU analysis function (TRANSIT=DFC).</td>
</tr>
</tbody>
</table>

**Table 39. Primary Function Codes with Secondary Function Codes**

<table>
<thead>
<tr>
<th>Primary Function</th>
<th>Secondary Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'09'</td>
<td>X'00'</td>
<td>File switch - NPM not ending</td>
</tr>
<tr>
<td></td>
<td>X'04'</td>
<td>File switch - NPM ending</td>
</tr>
<tr>
<td>X'19'</td>
<td></td>
<td>See Table 47 on page 260 for the contents of this field.</td>
</tr>
</tbody>
</table>
User Data Field  
The user data field is the third parameter in the function parameter list for all functions. This field can be used for any purpose. For example, it can store the address of dynamically obtained storage area. NPM saves the contents of this user field, making it available to the exit routine the next time it is called. NPM returns the address of this field on subsequent calls, not the data itself.

Exit Options Field  
The exit options field is the fourth parameter in the function parameter list for the begin and end functions. This 4-byte field indicates the functions for which the exit routine is called. It is modified by the exit routine during begin function processing.

The exit options field is named BFLAG1 in the sample FNMUEXIT. The bit definitions are described in Table 40.

**Table 40. Exit Options**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B '1... ....'</td>
<td>The NPM operator session connect and disconnect exits are called.</td>
</tr>
<tr>
<td>B '1... ....'</td>
<td>The NPALU session connect and disconnect exits are called.</td>
</tr>
<tr>
<td>B '...1 ....'</td>
<td>The NPM-to-NPM session connect and disconnect exits are called.</td>
</tr>
<tr>
<td>B '...1 ....'</td>
<td>The dynamic network collection authorization function is called.</td>
</tr>
<tr>
<td>B '.... 1...'</td>
<td>The file switch function is called.</td>
</tr>
<tr>
<td>B '.... .1..'</td>
<td>The message output function is called.</td>
</tr>
<tr>
<td>B '.... ..1.'</td>
<td>The output record and alert functions are called.</td>
</tr>
<tr>
<td>B '.... ...1'</td>
<td>The transit session analysis function is called.</td>
</tr>
</tbody>
</table>

Operator User Data Field  
The operator user data field is the fourth parameter in the function parameter list for the operator connect, operator disconnect, and message functions. This field contains the address of a word that is associated with an operator session.

The function processing routines can use the operator user data field for any purpose. For example, if the value of the parameter is -1 during message processing, the message is not directed to a specific operator. During logon processing, the contents of the field are saved. The field is initialized to zeros.

File User Data Field  
The file user data field is the fourth parameter in the function parameter list for the file switch and file output functions. This 4-byte field, originally initialized to zero, can be used for any purpose. For example, it can store the address of dynamically obtained storage. NPM saves the contents of this user field so that it is available to the exit routine the next time it is called. NPM returns the address of this field on subsequent calls, not the data itself.

There is one file user field for each type of file: FNMSESx, FNMREVx, FNMLOGx, FNMVLOGx, FNMSMF, FNMALET, and FNMLWGx.
Session User Data Field

The session user data field is the fourth parameter in the function parameter list for the session interval analysis and the PIU analysis functions. Table 41 shows the format of this field.

Table 41. Session User Data Area

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>8</td>
<td>Primary LU name.</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Primary LU subarea number.</td>
</tr>
<tr>
<td>12(C)</td>
<td>2</td>
<td>Primary LU element address.</td>
</tr>
<tr>
<td>14(E)</td>
<td>8</td>
<td>Secondary LU name.</td>
</tr>
<tr>
<td>22(16)</td>
<td>4</td>
<td>Secondary LU subarea number.</td>
</tr>
<tr>
<td>26(1A)</td>
<td>2</td>
<td>Secondary LU element address.</td>
</tr>
<tr>
<td>28(1C)</td>
<td>8</td>
<td>Job name or operator ID for the session. For TSO sessions, this is the TSO operator ID.</td>
</tr>
<tr>
<td>36(24)</td>
<td>8</td>
<td>User data field. This field is initialized to zero for the first PIU during a session interval. It is moved to the LSCDUWRD field in the SMF28SCD data section, where the detail data for the session is written.</td>
</tr>
<tr>
<td>44(2C)</td>
<td>8</td>
<td>Session manager ID. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record.</td>
</tr>
<tr>
<td>52(34)</td>
<td>8</td>
<td>Session manager operator ID. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record. If an application uses the NSI interface to report operator IDs or if the session is a TSO session, this field might not be blank.</td>
</tr>
<tr>
<td>60(3C)</td>
<td>8</td>
<td>Session manager group ID. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record.</td>
</tr>
<tr>
<td>68(44)</td>
<td>8</td>
<td>Session manager account number. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record.</td>
</tr>
<tr>
<td>76(4C)</td>
<td>8</td>
<td>Session manager PLU name. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record.</td>
</tr>
<tr>
<td>84(54)</td>
<td>8</td>
<td>Session manager SLU name. This field is blank if the LU session is not controlled by a session manager. If the field is not blank, it is copied to the configuration section of the session detail record.</td>
</tr>
</tbody>
</table>
Installation-Wide Exit Parameter
The exit parameter field is the fifth parameter in the function parameter list for the begin function. It contains the address of a 16-character field entered from the SYS initialization statement. You can set this field to a character string for inspection by the begin function using the PARM keyword. The default value is blanks.

User Security Area
The user security area is the fifth parameter of the function parameter list for the operator connect function. Table 42 shows the format of this area.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>2</td>
<td>Length of the user security area</td>
</tr>
<tr>
<td>2(2)</td>
<td>1</td>
<td>Length of the operator ID (8)</td>
</tr>
<tr>
<td>3(3)</td>
<td>8</td>
<td>The symbolic name for the operator ID</td>
</tr>
<tr>
<td>11(B)</td>
<td>1</td>
<td>Length of the password (8)</td>
</tr>
<tr>
<td>12(C)</td>
<td>8</td>
<td>The password character string</td>
</tr>
<tr>
<td>20(14)</td>
<td>1</td>
<td>Length of the new password (8)</td>
</tr>
<tr>
<td>21(15)</td>
<td>8</td>
<td>The new password character string</td>
</tr>
<tr>
<td>29(1D)</td>
<td>1</td>
<td>Length of the profile name (8)</td>
</tr>
<tr>
<td>30(1E)</td>
<td>8</td>
<td>The symbolic name for the profile</td>
</tr>
<tr>
<td>38(26)</td>
<td>1</td>
<td>Length of the RACF group name (8)</td>
</tr>
<tr>
<td>39(27)</td>
<td>8</td>
<td>The symbolic name for the RACF group name</td>
</tr>
<tr>
<td>47(2F)</td>
<td>1</td>
<td>Length of the application name (8)</td>
</tr>
<tr>
<td>48(30)</td>
<td>8</td>
<td>The symbolic name for the NPM application</td>
</tr>
<tr>
<td>56(38)</td>
<td>1</td>
<td>Length of the terminal name (8)</td>
</tr>
<tr>
<td>57(39)</td>
<td>8</td>
<td>The symbolic name for the terminal</td>
</tr>
<tr>
<td>65(41)</td>
<td>1</td>
<td>Length of the job name (8)</td>
</tr>
<tr>
<td>66(42)</td>
<td>8</td>
<td>The symbolic name for the NPM job name</td>
</tr>
</tbody>
</table>

Any of the fields in the user security area can be changed by the exit function routine. If you are using RACF security, the application, terminal, and job name fields are passed to the RACINIT function. The GROUP name is also passed if the field is filled by the exit routine.

Data Definition Name (DD Name)
The DD name field is the fifth parameter in the function parameter list for the file switch function. This eight-byte character field contains the data definition name of the file that is being closed or switched.
Message Parameter List
The message parameter list field is the fifth parameter in the function parameter list for the message function. The WTO format list contains the length and text of the message. Table 43 shows the format of the message parameter list.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>2</td>
<td>Length of the message text (n)</td>
</tr>
<tr>
<td>2(2)</td>
<td>2</td>
<td>WTO MCSFLAG field</td>
</tr>
<tr>
<td>4(4)</td>
<td>(n)</td>
<td>Message text</td>
</tr>
</tbody>
</table>

Data Set Name
The data set name field is the sixth parameter in the function parameter list for the file switch function. This 44-byte character field contains the data set name of the file being closed or switched.

WTO Descriptor and Routing Codes (MVS Only)
The WTO descriptor and routing codes are the sixth parameter in the function parameter list for the message function. This 20-byte field contains the descriptor and routing codes in the format shown in Table 44.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>2</td>
<td>Descriptor codes (1–16)</td>
</tr>
<tr>
<td>0(2)</td>
<td>2</td>
<td>Reserved</td>
</tr>
<tr>
<td>0(4)</td>
<td>16</td>
<td>Routing codes (1–128)</td>
</tr>
<tr>
<td>0(4)</td>
<td>2</td>
<td>Routing codes (1–16)</td>
</tr>
<tr>
<td>6(6)</td>
<td>14</td>
<td>Routing codes (17–128)</td>
</tr>
</tbody>
</table>

The descriptor codes 1, 2, 3, 4, 5, 6, and 11 are mutually exclusive. However, NPM does not check for this condition.

If ROUTCDE=0 is specified on the CONSOLE initialization statement, the default route codes are set to zeros. If the message is not defined with routing codes by the macro FNMMMSG, the message is not written to the system console. Routing codes 29 through 41 are ignored and set to zero. Routing codes 42 through 128 are reserved for control program use. If the routing code field is set to all zeros, the message is not written to the system console. See MVS/XA Supervisor Services and Macro Instructions for the meaning of each code.

Volume Serial Field
The file volume serial field is the seventh parameter in the function parameter list for the file switch function. This 6-byte character field contains the volume serial name of the file that is being closed or switched.
SMF Data Areas

Some of the data areas passed to the exit functions are SMF data sections. Table 45 lists the SMF data section passed to each function. The exit routine can modify the data in each data section passed to it. However, if an NPM program reads the data, the results of the modifications are unknown.

Table 45. SMF Data Areas

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameter</th>
<th>Data Area</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'02'</td>
<td>5</td>
<td>SMF28OGF</td>
<td>Operator LOGOFF data section</td>
</tr>
<tr>
<td>X'03'</td>
<td>5</td>
<td>SMF28NNC</td>
<td>NPALU connect data section</td>
</tr>
<tr>
<td>X'04'</td>
<td>5</td>
<td>SMF28NND</td>
<td>NPALU disconnect data section</td>
</tr>
<tr>
<td>X'05'</td>
<td>5</td>
<td>SMF28NPC</td>
<td>NPM connect data section</td>
</tr>
<tr>
<td>X'06'</td>
<td>5</td>
<td>SMF28NPD</td>
<td>NPM disconnect data section</td>
</tr>
<tr>
<td>X'07'</td>
<td>5</td>
<td>SMF28SMA</td>
<td>Session management data section</td>
</tr>
<tr>
<td>X'10'</td>
<td>5</td>
<td>SMF28RCD</td>
<td>SMF data record</td>
</tr>
<tr>
<td>X'11'</td>
<td>5</td>
<td>FNMGTF</td>
<td>VTAMLOG data record</td>
</tr>
<tr>
<td>X'12'</td>
<td>5</td>
<td>FNMLVSM</td>
<td>VSAM data record</td>
</tr>
<tr>
<td>X'13'</td>
<td>5</td>
<td>FNMLVSM</td>
<td>VSAM data record</td>
</tr>
<tr>
<td>X'14'</td>
<td>5</td>
<td>SMF28RCD</td>
<td>SMF data record</td>
</tr>
<tr>
<td>X'16'</td>
<td>5</td>
<td>FNMKVSM</td>
<td>VSAM data record</td>
</tr>
<tr>
<td>X'18'</td>
<td>5</td>
<td>SMF28STT</td>
<td>Session transit data section</td>
</tr>
<tr>
<td>X'18'</td>
<td>6</td>
<td>SMF28SAA</td>
<td>Session accounting data section</td>
</tr>
</tbody>
</table>

Installing the NPM Installation-Wide Exit

The name of the NPM exit routine is defined in the SYS initialization statement by the EXIT parameter. By changing the value of the EXIT parameter, it is possible to create several different exit routines. However, only one exit routine can be active during the execution of NPM.

NPM supplies a sample installation-wide exit named FNMUEXIT. You can locate the sample in NPM.V2R4M0.SFNMSRC1(FNMUEXIT) for MVS. Modify this sample exit so that it performs the functions you need. You must code this exit as a reentrant program.

Ensure that the NPM macro library and operating system library are both available to the assembler. The assembler needs these libraries to resolve any external references and to expand any system macros. For MVS, the NPM macro library is in NPM.V2R4M0.SFMMAC1.

The begin function of the NPM installation-wide exit contains a flag that enables all of the other functions in the exit. You must code the begin function and update the exit options flag to enable the exit functions you want to use.
Installing the NPM Installation-Wide Exit for MVS

To install FNMUEXIT on MVS, you must specify the proper AMODE for MVS systems with extended addressing. The NPM installation-wide exit load module must be in LINKLIB, JOBLIB, a STEPLIB, or a library concatenated to one of these libraries. The library must be authorized.

To install an exit to your application, complete the following steps:

Step 1. Decide what functions you need. The functions determine when NPM passes control to your routine during NPM processing.

Step 2. Copy FNMUEXIT from the source library, NPM.V2R4M0.SFNMSRC1.

Step 3. Read the code and comments of FNMUEXIT to understand how to code an installation-wide exit.

Step 4. Create your own exit routine or modify FNMUEXIT to pass control to your functions.

Step 5. Ensure that the begin function (X'FE') updates the exit options field (named BFLAG1 in the sample FNMUEXIT) to enable the functions you want to use. The exit options field is shown in Table 40 on page 254.

Step 6. Assemble the installation-wide exit and link-edit it as a reentrant program. The SMP/E USERMOD automates this step. See step 14 on page 134.

The PIU Analysis Exit

The PIU analysis exit determines the type of PIUs flowing through the network and specifies the way NPM uses the PIU to calculate transit times. It is called for each user PIU that NPM is using to capture transit time data.

Register Contents

The register contents when NPM passes control to the PIU exit routine are the same as those for the installation-wide exit. See “Register Contents” on page 237 for these contents.

The PIU analysis routine returns one of the following return codes in Register 15:

00 The PIU is not used to calculate transit times.
04 The PIU is an input PIU to be used as the start of the host component of the transit time.
08 The PIU is an output PIU that ends the host component of the transit time and begins the network component.
12 The PIU is an inbound PIU that contains the definite response to end the network component of the transit time.
16 The PIU is an output response that ends transit time calculations.
20 The PIU is an outbound data flow control (DFC) request. This starts the timing to calculate DFC network transit timing.
24 The PIU is an outbound DFC response. This stops the timing to calculate DFC network transit timing.
28 Reserved.
PIU Analysis Exit

32 Do not call the PIU analysis exit again for this session interval. Call the NPM standard session analysis exit.

36 Call the NPM standard session analysis exit. Call this exit again for the next PIU.

Note: For return codes 32 and 36, the standard session analysis exit routine is called to determine the type of PIU.

If the exit routine returns another return code, the PIU is ignored and NPM issues message FNM966S to the operator.

Parameter Descriptions

You can code the PIU analysis exit as a function in the NPM installation-wide exit (function code X'19'). As a result, its parameters are the same as those listed in “The NPM Installation-Wide Exit” on page 237. In addition, it contains two other parameters:

- The VTAM transmission header and request header
- The PIU data area

VTAM Transmission Header and Request Header

The VTAM transmission header and request header are the fifth parameter in the function parameter list for the PIU analysis exit. The transmission header is 26 bytes and is immediately followed by the 3-byte request header.


PIU Data Area

The PIU data area field is the sixth parameter in the function parameter list for the PIU analysis exit. Table 46 shows the format of this data area.

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>2</td>
<td>Size of RU data buffer</td>
</tr>
<tr>
<td>2(2)</td>
<td>256</td>
<td>RU data buffer</td>
</tr>
</tbody>
</table>

Note: The amount of RU data provided depends on the value of MINRUSZ.

Exit Routine Secondary Function Codes

Table 47 describes the secondary function code field of the exit routine function code parameter. This field is used only by the PIU analysis exit.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B '1... ....'</td>
<td>Request/response indicator</td>
</tr>
<tr>
<td>0</td>
<td>Request PIU</td>
</tr>
<tr>
<td>1</td>
<td>Response PIU</td>
</tr>
</tbody>
</table>
Table 47 (Page 2 of 2). PIU Analysis Flags

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B '.1. ....'</td>
<td>PIU direction indicator</td>
</tr>
<tr>
<td>0</td>
<td>Outbound PIU</td>
</tr>
<tr>
<td>1</td>
<td>Inbound PIU</td>
</tr>
<tr>
<td>B '..1. ....'</td>
<td>Begin bracket indicator</td>
</tr>
<tr>
<td>0</td>
<td>Not a begin bracket PIU</td>
</tr>
<tr>
<td>1</td>
<td>Begin bracket PIU</td>
</tr>
<tr>
<td>B '...1 ....'</td>
<td>End bracket indicator</td>
</tr>
<tr>
<td>0</td>
<td>Not an end bracket PIU</td>
</tr>
<tr>
<td>1</td>
<td>End bracket PIU</td>
</tr>
<tr>
<td>B '.... 1...'</td>
<td>Definite response indicator</td>
</tr>
<tr>
<td>0</td>
<td>Not a definite response PIU</td>
</tr>
<tr>
<td>1</td>
<td>Definite response PIU (DR1 or DR2)</td>
</tr>
<tr>
<td>B '.... .1..'</td>
<td>Sense data indicator</td>
</tr>
<tr>
<td>0</td>
<td>No sense data in PIU</td>
</tr>
<tr>
<td>1</td>
<td>Sense data in PIU</td>
</tr>
<tr>
<td>B '.... ..1.'</td>
<td>Begin chain indicator</td>
</tr>
<tr>
<td>0</td>
<td>Not a begin chain PIU</td>
</tr>
<tr>
<td>1</td>
<td>Begin chain PIU</td>
</tr>
<tr>
<td>B '.... ...1'</td>
<td>End chain indicator</td>
</tr>
<tr>
<td>0</td>
<td>Not an end chain PIU</td>
</tr>
<tr>
<td>1</td>
<td>End chain PIU</td>
</tr>
</tbody>
</table>

See “Exit Routine Function Codes” on page 252 for a description of the primary function code field.

PIU Analysis Parameter List

Table 48 table shows the PIU analysis parameter list pointed to by register 1. The parameters are described in the previous section and in “Parameter Descriptions” on page 249.

Table 48. PIU Analysis Function Parameter List

<table>
<thead>
<tr>
<th>Dec(Hex) Offset</th>
<th>Size (Bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(0)</td>
<td>4</td>
<td>Address of the environment vectors</td>
</tr>
<tr>
<td>4(4)</td>
<td>4</td>
<td>Address of the exit routine function code</td>
</tr>
<tr>
<td>8(8)</td>
<td>4</td>
<td>Address of the common user data field</td>
</tr>
<tr>
<td>12(C)</td>
<td>4</td>
<td>Address of the session user data field</td>
</tr>
<tr>
<td>16(10)</td>
<td>4</td>
<td>Address of the VTAM transmission header and request header</td>
</tr>
<tr>
<td>20(14)</td>
<td>4</td>
<td>Address of the PIU data</td>
</tr>
</tbody>
</table>
Installing a PIU Analysis Exit

The PIU analysis exit is not normally installed as part of FNMUEXIT. Instead, it is placed in its own load module and named in the EXIT parameter on the APPL statement. Each application can have its own exit routine for PIU analysis. When applications do not specify an exit routine, NPM defines its own exit routine for PIU analysis.

To install a PIU analysis exit for MVS, complete the following steps:

**Step 1.** Write your exit routine.

*Note:* The PIU analysis exit is difficult to code. You should test your exit thoroughly on a test system before putting it into production. See “Testing a PIU Analysis Exit” for information about using the transit analysis program (FNMTAP) to test your exit.

**Step 2.** Assemble and link-edit the exit routine.

Testing a PIU Analysis Exit

You can use the transit analysis program (FNMTAP) to test an application session collection installation exit. FNMTAP processes a subset of NPM installation-wide exit functions:

- **BEGIN** X‘FE’
  Called once when FNMTAP is started.

- **END** X‘FF’
  Called once when FNMTAP is ended.

- **SESSION** X‘19’
  Called for each PIU with TRANSIT=DR.

- **SESSION** X‘1A’
  Called for each PIU with TRANSIT=DFC.

- **ANALYSIS** X‘18’
  Called for each session record before being written.

- **TAPLOG** X‘16’
  Called for each session record before being written. This is the same function for the FNMLOG records.

To use FNMTAP to test your PIU analysis exit, code a SYS control statement in the SYSIN data set in the MVS JCL or VM EXEC that invokes FNMTAP. Sample SYS statements are provided in NPM.V2R4M0.SFNMJCL1(FNMTAP) for MVS and FMMTAPV EXEC for VM.

```
PARM=’’
EXIT=NONE
```

The following list describes the keywords and operands you can include in the SYS statement:
Specifies the 16-byte character string passed to the NPM installation-wide exit for the BEGIN function. If the string contains special characters or blanks, enclose the string in single quotation marks. If PARM is not coded, the string is set to blanks.

Specifies the name of the NPM installation-wide exit routine. This routine is loaded and called as specific events occur during NPM operation.

See *NPM Reference* for more information about FNMTAP.
Glossary

This glossary includes terms and definitions from:

- The American National Standard Dictionary for Information Systems, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies can be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The ANSI/EIA Standard — 440-A, Fiber Optic Terminology. Copies can be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006. Definitions are identified by the symbol (E) after the definition.
- The Information Technology Vocabulary, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.


The following cross-references are used in this glossary:

Contrast with: This refers to a term that has an opposed or substantively different meaning.

Synonym for: This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.

Synonymous with: This is a backward reference from a defined term to all other terms that have the same last word.

See: This refers to multiple-word terms that have the same meaning.

See also: This refers to terms that have a related, but not synonymous, meaning.

Deprecated term for: This indicates that the term should not be used. It refers to a preferred term, which is defined in its proper place in the glossary.

ACF/VTAM. Advanced Communications Function for the Virtual Telecommunications Access Method. Synonym for VTAM.

acquire. (1) In VTAM, to take over resources that were formerly controlled by an access method in another domain or to resume control of resources that were controlled by that domain but released. Contrast with release. See also resource takeover. (2) In a VTAM application program, to initiate and establish a session with another logical unit (LU). The acquire process begins when the application program issues a macroinstruction. See also accept.

active application. The application subsystem currently in an extended recovery facility (XRF) session with a terminal user. See also alternate application.

active collection. In NPM, the process of collecting data at the current time about a resource.

ACTPU. Activate physical unit. In SNA, a command used to start a session on a physical unit.

adapted control block (ACB). In NCP, a control block that contains line control information and the states of I/O operations for BSC lines, SS lines, or SDLC links.

adaptive pacing. Synonym for adaptive session-level pacing.

adapted session pacing. Synonym for adaptive session-level pacing.

adaptive session-level pacing. A form of session-level pacing in which session components exchange pacing windows that may vary in size during the course of a session. This allows transmission within a network to adapt dynamically to variations in availability and demand of buffers on a session-by-session basis. Session-level pacing occurs within independent stages along the session path according to local congestion at the intermediate nodes. Synonymous with adaptive pacing and adaptive session pacing. See pacing, session-level pacing, and virtual route pacing.

address space. A set of addresses used to uniquely identify network accessible units, sessions, adjacent link stations, and links in a node for each network in which the node participates. A type 2.1 node has one address space for intranode routing and one for each transmission group on which it can send message units.

address space control block (ASCB). A control block that contains the dispatching priority of an address space. There is an ASCB for...
each address space. The dispatcher, an MVS supervisor routine, selects the first ready ASCB on the chain of ASCBs.

**address space identifier (ASID).** A unique, system-assigned identifier for an address space.

**address space manager (ASM).** A component in a type 2.1 node that assigns and frees session addresses.

**address translation.** See network address translation.

**addressing mode (AMODE).** In MVS, a program attribute that refers to the address length that a program is prepared to handle upon entry. In the MVS/XA program, addresses can be 24 bits or 31 bits in length.

**adjacent link station (ALS).** (1) In SNA, a link station directly connected to a given node by a link connection over which network traffic can be carried. **Note:** Several secondary link stations that share a link connection do not exchange data with each other and therefore are not adjacent to each other. (2) With respect to a specific node, a link station partner in an adjacent node.

**adjacent nodes.** Two nodes connected together by at least one path that connects no other node. (1)

**ADP.** Automatic data processing. (A)

**Advanced Communications Function (ACF).** A group of IBM-licensed programs, principally VTAM, TCA M, NCP, and SSP, that use the concepts of Systems Network Architecture (SNA), including distribution of function and resource sharing.

**advanced peer-to-peer networking (APPN) end node.** A type 2.1 node that provides full SNA end-user services and supports sessions between its local control point (CP) and the CP in an adjacent network node, to dynamically register its resources with the adjacent CP (its network node server), to send and receive directory search requests, and to obtain management services; it can also attach to a subarea network as a peripheral node.

**advanced peer-to-peer networking (APPN) network.** An SNA network consisting of APPN network nodes, APPN end nodes, and, optionally, LEN end nodes.

**advanced peer-to-peer networking (APPN) network node.** A type 2.1 node that, besides offering full SNA end-user services, provides intermediate routing services within a type 2.1 network and network services to its local LUs and attached type 2.1 end nodes in its domain; it can also attach to a subarea network as a peripheral node.

**advanced peer-to-peer networking (APPN) node.** An APPN network node or an APPN end node.

**advanced peer-to-peer networking (APPN).** An extension to SNA that provides greater distributed network control that avoids critical hierarchical dependencies, thereby isolating the effects of single points of failure, (b) dynamic exchange of network topology information to foster ease of connection and reconfiguration, adaptive route selections, and simplified network definition, and (c) automated resource registration and directory lookup. APPN extends the LU 6.2 peer orientation for end-user services to network control; APPN also uses LU 6.2 protocols on its own control point sessions that provide the network control.

**advanced program-to-program communication (APPC).** An implementation of the SNA/SDLC LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs.

**agent.** Reports to the managing process, such as a network control station, on the status of managed network elements and performs actions on these elements as directed by the managing process. Examples of network elements are devices such as hosts, routers, and terminal servers.

**aggregate.** In programming languages, a structured collection of data objects that form a data type. (1)
application transaction program. A program written for or by a user to process the user's application; in an SNA network, an end user of a type 6.2 logical unit. Contrast with service transaction program.

APPLID. Application ID.

apply. An SMP process that moves distributed code and MVS-type programs to the system libraries.

APPN. Advanced peer-to-peer networking.

APPN end node. See advanced peer-to-peer networking (APPN) end node.

APPN intermediate routing. The capability of an APPN network node to accept traffic from one adjacent node and pass it on to another, with awareness of session affinities in controlling traffic flow and outage notifications.

APPN intermediate routing network. The portion of an APPN network consisting of the network nodes and their interconnections.

APPN network. See advanced peer-to-peer networking network.

APPN network node. See advanced peer-to-peer networking (APPN) network node.

APPN node. See advanced peer-to-peer networking (APPN) node.

AS/400. IBM Application System/400.

ASA. American Standards Association.

ASCB. Address space control block.

ASCII (American National Standard Code for Information Interchange). The standard code, using a coded character set consisting of 7-bit coded characters (8-bit including parity check), that is used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters. (A)

Note: IBM has defined an extension to ASCII code (characters 128-255).

ASID. Address space identifier.

ASM. (1) Auxiliary storage manager. (2) Address space manager.

asynchronous operation. Simultaneous operations of software or hardware. In software, an operation, such as a request for session establishment or data transfer, in which the application program is allowed to continue execution while the operation is performed. The access method informs the application program after the operation is completed. Contrast with synchronous operation.

ATCVT. VTAM communication vector table.

attaching device. Any device that is physically connected to a network and can communicate over the network.

authorized program analysis report (APAR). A request for correction of a problem caused by a defect in a current unaltered release of a program.

authorized program facility. A facility that permits identification of programs that are authorized to use restricted functions.

automated console operations (ACO). The use of automated procedures to replace or simplify the actions that an operator takes from a console in response to system or network events.

automatic activation. In VTAM, the activation of links and link stations in adjacent subarea nodes as a result of channel device name or RNAME specifications related to an activation command that names a subarea node. See also direct activation.

autotask. An unattended NetView operator station task that does not require a terminal or a logged-on user. Autotasks can run independently of VTAM and are typically used for automated console operations. Contrast with logged-on operator.

available. In VTAM, pertaining to a logical unit that is active, connected, enabled, and not at its session limit.

B

backup session. The session that replaces the failing primary extended recovery facility (XRF) session between a terminal user and the active subsystem.

BAF. batch archive facility.

base collection interval. The primary unit, in seconds or minutes, which NPM uses to calculate the span of time in which data is collected.

BASE disk. The virtual disk that contains the text decks and macroinstructions for VTAM, NetView, and VM/SNA console support (VSCS). It also contains control files and sample files used when running VTAM on the VM operating system. See also DELTA disk, MERGE disk, RUN disk, and ZAP disk.

basic information unit (BIU). In SNA, the unit of data and control information passed between half-sessions. It consists of a request/response header (RH) followed by a request/response unit (RU).

basic sequential access method (BSAM). In NPM, the method by which all PIUs collected for selected LUs can be logged into a sequential data set as they pass through VTAM.

basic transmission unit (BTU). In SNA, the unit of data and control information passed between path control components. A BTU can consist of one or more path information units (PIUs). See also blocking of PIUs.

BCU. Batch configuration utility.

begin bracket. In SNA, the value (binary 1) of the begin-bracket indicator in the request header (RH) of the first request in the first chain of a bracket; the value denotes the start of a bracket. Contrast with end bracket. See also bracket.

bidder. Synonym for bidder session.

bidder session. The half-session defined at session activation as having to request and receive permission from the other half-session to begin a bracket. Contrast with first-speaker session. Synonym for contentor-loser session. Synonymous with bidder.

binary synchronous communication (BSC). A form of telecommunication line control that uses a standard set of transmission control characters and control character sequences, for binary synchronous transmission of binary-coded data between stations. Contrast with Synchronous Data Link Control (SDLC).

binary synchronous transmission. Data transmission in which synchronization of characters is controlled by timing signals generated at the sending and receiving stations. See also start-stop transmission and Synchronous Data Link Control (SDLC).

BIND. In SNA, a request to activate a session between two logical units (LUs). See also session activation request. Contrast with UNBIND.

bit map. A representation of an image by an array of bits.

BIU. Basic information unit.

BIU segment. In SNA, the portion of a basic information unit (BIU) that is contained within a path information unit (PIU). It consists of either a request/response header (RH) followed by all or part of a request/response unit (RU), or of only a part of an RU.

BLKSIZE. block size.
block size.  (1) The number of data elements in a block.  (T)  A measure of the size of a block, usually specified in units, such as records, words, computer words, or characters.  (A)

blocking of PIUs.  In SNA, an optional function of path control that combines multiple path information units (PIUs) in a single basic transmission unit (BTU).

Note:  When blocking is not done, a BTU consists of one PIU.

BMAP.  Bit map.

border node.  Network nodes attach to one another natively; i.e., two attached network nodes have the same network ID.  APPN networks having different network IDs are said to be nonnative with respect to each other.  One way nonnative APPN networks can be interconnected is by a peripheral border node.  Topology information is not passed between the subnetworks.  Similarly a border node can also connect to another border node.  Two types of border node are defined in the APPN architecture:

- Peripheral border node
- Extended border node

BN.  Boundary node.

boundary function.  (1) In SNA, a capability of a subarea node to provide protocol support for attached peripheral nodes, such as: (a) interconnecting subarea path control and peripheral path control elements, (b) performing session sequence numbering for low-function peripheral nodes, and (c) providing session-level pacing support.  (2) In SNA, the component that provides these capabilities.  See also boundary node, network addressable unit (NAU), peripheral path control, subarea node, and subarea path control.

boundary node (BN).  In SNA, a subarea node with boundary function.

Note:  A subarea node may be a boundary node, an intermediate routing node, both, or neither, depending on how it is used in the network.

BPS.  Bits per second.

bracket.  In SNA, one or more chains of request units and their responses that are exchanged between two session partners and that represent a transaction between them.  A bracket must be completed before another bracket can be started.  Examples of brackets are database inquiries/replies, update transactions, and remote job entry output sequences to workstations.

bracket protocol.  In SNA, a data flow control protocol in which exchanges between two session partners are achieved through the use of brackets, with one partner designated as session activation as the first speaker and the other as the bidder.  The bracket protocol involves bracket initiation and termination rules.

broadcast search.  The propagation of a search request, when the location of a resource is unknown to the requester, to all network nodes in an APPN network.

A broadcast search does not use database information about the location of a requested LU to propagate the search, because a directed search using database information has failed or there is no database information for the requested resource.  Each network node receiving the search request sends the search to each of its adjacent NN; this allows the whole network to be searched.  When the search reaches the NN serving the destination resource, that node sends back a positive reply to the first search request it receives.  All subsequent search requests received by that NN as part of the same search are answered with a negative reply.  A broadcast search is initiated only by a NN server for the LUs it serves if there is no central directory server (CDS) defined in the network.  If there is a CDS defined, only the CDS can issue a broadcast search.  If a CDS is active in the network, a NN server sends a directed search to the CDS instead of using a broadcast search.  The NN server can issue its own broadcast search only if the directed search is unable to reach the CDS.  A broadcast search is needed when there is no directory information giving the location of a requested LU or a directed search using existing directory information fails.  Contrast with directed Locate search.

BSAM.  Basic sequential access method.

BSC.  Binary synchronous communication.

BTU.  Basic transmission unit.

buffer pool.  (1) An area of storage in which all buffers of a program are kept.  (2) In ACF/TCAM, a group of buffers having the same size.  A buffer pool is established at initialization time in the message control program.  The buffers are built in extents chained together.

cache.  (1) An optional part of the directory database in network nodes where frequently used directory information may be stored to speed directory searches.  (2) To place, hide, or store in a cache.

call-accepted packet.  A call supervision packet that a called data terminal equipment (DTE) transmits to indicate to the data circuit-terminating equipment (DCE) that it accepts the incoming call.

call-connected packet.  A call supervision packet that a data circuit-terminating equipment (DCE) transmits to indicate to a calling data terminal equipment (DTE) that the connection for the call has been completely established.

call-request packet.  A call supervision packet that a data terminal equipment (DTE) transmits to ask that a connection for a call be established throughout the network.

call-supervision packet.  A packet used to establish or clear a call at the interface between the data terminal equipment (DTE) and the data circuit-terminating equipment (DCE).

callout.  The logical channel type on which the data terminal equipment (DTE) can send a call, but cannot receive one.

carrier.  (1) An electric or electromagnetic wave or pulse train that may be varied by a signal bearing information to be transmitted over a communication system.  (T)  (2) In data communication, a continuous frequency that can be modulated or impressed with an information carrying signal.

carrier sense.  In a local area network, an ongoing activity of a data station to detect whether another station is transmitting.  (T)

carrier sense multiple access with collision detection (CSMA/CD).  A protocol that requires carrier sense and in which a transmitting data station that detects another signal while transmitting, stops sending, sends a jam signal, and then waits for a variable time before trying again.  (T)  (A)

CAT.  Common address table.

CBS.  Communication-based system.

CCA.  (1) Common communication adapter.  (2) Console communication area.

CCITT.  Comité Consultatif International Télégraphique et Téléphonique.  The International Telegraph and Telephone Consultative Committee.

CCP.  (1) Communication control program.  (2) Configuration control program.

CCU.  (1) Communication control unit.  (2) Central control unit.

CCW.  (1) Channel control word.  (2) Channel command word.

CD-ROM.  High-capacity read-only memory in the form of an optically read compact disk.

CDI.  (1) Command and data interface.  (2) Change direction indicator.
Glossary

CDRM. Cross-domain resource manager.

CDRSC. Cross-domain resource.

CDS. (1) Central directory server (2) Control data set.

central directory server. An APPN network node that provides a repository for network resource locations; it also reduces the number of network searches by providing a focal point for queries and by caching the results of network node searches so that later searches do not have to be broadcast.

CE. An IBM service representative (customer engineer) who performs maintenance services for IBM software or IBM hardware.

central control unit (CCU). The communication controller hardware unit that contains the circuits and data flow paths needed to execute instructions and to control controller storage and the attached adapters.

central processing unit (CPU). (1) Synonym for processing unit. (T) (2) The part of a computer that includes the circuits that control the interpretation and execution of instructions. See processing unit. See also processor.

Note: A CPU is the circuitry and storage that executes instructions. Traditionally, the complete processing unit was often regarded as the CPU, whereas today the CPU is often a microchip. In either case, the centrality of a processor or processing unit depends on the configuration of the system or network in which it is used.

CfgView. Configuration window.

chain. (1) A group of logically linked user data records processed by LU 6.2. (2) A group of request units delimited by begin-chain and end-chain. Responses are always single-unit chains. See RU chain.

channel-attached. (1) Pertaining to the attachment of devices directly by input/output channels to a host processor. (2) Pertaining to devices attached to a controlling unit by cables, rather than by telecommunication lines. Contrast with link-attached. Synonymous with local.

channel-to-channel (CTC). A method of connecting two computing devices.

character-coded. Synonym for unformatted.

CICS. Customer Information Control System.

CID. (1) Communication identifier. (2) Connection identifier.

CINIT. A network services request sent from a system services control point (SSCP) to a logical unit (LU) asking that LU to establish a session with another LU and to act as the primary end of the session.

circuit switching. (1) A process that, on demand, connects two or more data terminal equipment (DTEs) and permits the exclusive use of a data circuit between them until the connection is released. (I) (A) (2) Synonymous with line switching. (3) See also message switching and packet switching.

CKD. Count key data.

class of service (COS). A designation of the transport network characteristics, such as route security, transmission priority, and bandwidth, needed for a particular session. The class of service is derived from a mode name specified by the initiator of a session.

cleanup. In SNA products, a network services request, sent by a system services control point (SSCP) to a logical unit (LU), that causes a particular LU-LU session with that LU to be ended immediately without requiring the participation of either the other LU or its SSCP.

clear-indication packet. A call supervision packet that a data circuit-terminating equipment (DCE) transmits to inform a data terminal equipment (DTE) that a call has been cleared.

clear-request packet. A call supervision packet transmitted by a data terminal equipment (DTE) to ask that a call be cleared.

CLI. Command line interface.

CLIST. Command list.

closed user group (CUG). In data communication, a group of users who can communicate with other users in the group, but not with users outside the group.

Note: A data terminal equipment (DTE) can belong to more than one closed user group.

CLPA. Create link pack area.

CLSDST. Close destination.

cluster. (1) A station that consists of a control unit (a cluster controller) and the terminals attached to it. (2) In systems with VSAM, a named structure consisting of a group of related components, for example, a data component with its index component.

cluster controller. A device that can control the input/output operations of more than one device connected to it. A cluster controller may be controlled by a program stored and executed in the unit; for example, the IBM 3601 Finance Communication Controller. Or, it may be entirely controlled by hardware; for example, the IBM 3272 Control Unit.

CM/2. Communication Manager/2.

CMC. (1) Common module connection. (2) Communication management configuration.

CMS. Conversational monitor system.

CNM. (1) Communication network management. (2) Communication Network Manager.

collection interval. In NPM, a user-specified value that controls the time at which performance data is transmitted.

collection point block (CPB). In NPM, a control block used to coordinate the collection of network and session data.

collision. An unwanted condition that results from concurrent transmissions on a channel. See also carrier sense multiple access with collision detection (CSMA/CD).

command list (CLIST). A list of commands and statements designed to perform a specific function for the user.

command procedure (PROC). In the NetView program, either a command list or a command processor.

command processor. A user-written module designed to perform a specific function. Command processors, which can be written in assembler language or in a

common address table (CAT). In OS/VS2, a table that lists the data contained in the CSA.

common service area (CSA). In OS/VS2, a part of the common area that contains data areas addressable by all address spaces. These data areas are protected during use by the key of the requester.

common storage area (CSA). An area of storage common to all address spaces.

communication adapter. A mechanism that enables communication facilities to be attached to host processors.

Communication Control Program (CCP). A portion of the network control program communication interrupt control program (CICP) that initiates and ends I/O line operations, handles first-level line error recovery and recording, and administers commands issued by background programs.

communication control unit (CCU). A communication device that controls transmission of data over lines in a network. Communication control units include transmission control units (such as the 2702...
transmission control unit) and communication controllers (such as the 3720 or 3725).

communication controller. A type of communication control unit whose operations are controlled by one or more programs stored and executed in the unit. It manages the details of control and the routing of data through a network.

communication identifier (CID). In VTAM, a key for locating the control blocks that represent a session. The key is created during session establishment and deleted when the session ends.

communication management configuration (CMC). (1) In VTAM, a technique for configuring a network that allows for the consolidation of many network management functions for the entire network in a single host processor. (2) A multiple-domain network configuration in which one of the hosts, called the communication management configuration host, performs most of the controlling functions for the network, thus allowing the other hosts, called data hosts, to process applications. This is accomplished by configuring the network so that the communication management host owns most of the resources in the network that are not application programs. The resources that are not owned by the communication management host are the resources that are channel-attached stations of data hosts.

communication management configuration host node. The type 5 host processor in a communication management configuration that does all network-control functions in the network except for the control of devices channel-attached to data hosts. Synonymous with communication management host. Contrast with data host node.

communication management host. Synonym for communication management configuration host node. Contrast with data host.

communication network management (CNM). The process of designing, installing, operating, and managing distribution of information and control among users of communication systems.

communication network management (CNM) application program. A VTAM application program that issues and receives formatted management services request units for physical units. The NetView program is an example of a CNM application program.

communication network management (CNM) interface. The interface that the access method provides to an application program for handling data and commands associated with communication system management. CNM data and commands are handled across this interface.

communication network management (CNM) processor. A program that manages one of the functions of a communications system. A CNM processor is executed under control of the NetView program.

Communications storage manager (CMS). A component of VTAM that enables host applications to share data with VTAM and other CSM users without having to physically copy the data.

configuration. (1) The manner in which the hardware and software of an information processing system are organized and interconnected. (T) (2) The devices and programs that make up a system, subsystem, or network. (3) In CCP, the arrangement of controllers, lines, and terminals attached to an IBM 3710 Network Controller. Also, the collective set of item definitions that describe such a configuration.

Configuration Control Program (CCP). An IBM-licensed program used interactively to define, display, and alter configurations that contain network controllers.

configuration services. One of the types of network services in a control point (SSCP, NNC, ENCP, or EPeU). Configuration services activate, deactivate, and records the status of physical units, links, and link stations.

configuration window. A window in NPM Desk/2 in which a hierarchy of objects is created.

connected. In VTAM, the state of a physical unit (PU) or a logical unit (LU) that has an active physical path to the host processor containing the system services control point (SSCP) that controls the respective PU or LU.

connection. In VTAM, synonym for physical connection.

connection network. A representation within an APPN network of a shared-access transport facility, such as a token ring, that reduces the system-definition burden on each APPN end node attached to the facility. Each such node may represent its connectivity to the other, real nodes on the facility generically, by a single, virtual routing node, which each reports to its network node server in the APPN network. The report includes local signaling information needed by any partner wanting to contact it over the facility. The transport facility represented this way, and the assemblage of nodes using the same virtual routing node representation, are collectively referred to as a connection network. By matching references to the same virtual routing node during its route selection, the network node server passes the required DLC signaling information in the search reply to the node originating a session over the connection network.

contention. In a session, a situation in which both NAUs attempt to initiate the same action at the same time, such as when both attempt to send data in a half-duplex protocol (half-duplex contention), or both attempt to start a bracket (bracket contention). At session initiation, one NAU is defined to be the contention winner; its action will take precedence when contention occurs. The contention loser must get explicit or implicit permission from the contention winner to begin its action.

contention-loser session. To an NAU, a session for which it was defined during session initiation to be the contention loser.

contention-winner session. To an NAU, a session for which it was defined during session initiation to be the contention winner.

control block. (1) A storage area used by a computer program to hold control information. (2) In the IBM Token-Ring Network, a specifically formatted block of information provided from the application program to the Adapter Support Interface to request an operation.

control data set (CDS). In NPM, an SMP data set used in the NPM installation process.

control operator. For logical unit (LU) 6.2, a service transaction program that describes and controls the availability of certain resources. For example, it describes network resources accessed by the local LU, and it controls session limits between the LU and its partners. See also contention.

control point (CP). A collection of tasks that provide directory and route selection functions for advanced peer-to-peer networking (APPN). An end node control point provides its own configuration, session, and management services with assistance from the control point in its serving network node. A network node control point provides session and routing services.

control program (CP). (1) A computer program designed to schedule and to supervise the execution of programs of a computer system. (2) (A) (2) The component of VM/370 that manages the resources of a single computer with the result that multiple computing systems appear to exist.

control section (CSECT). The part of a program specified by the programmer to be a relocatable unit, all elements of which are to be loaded into adjoining main storage locations.

control statement. In the NetView program, a statement in a command list that controls the processing sequence of the command list or allows the command list to send messages to the operator and receive input from the operator.

controller. A device that coordinates and controls the operation of one or more input/output devices, such as workstations, and synchronizes the operation of such devices with the operation of the system as a whole.

controlling application program. In VTAM, an application program with which a secondary logical unit (other than an application program)
Glossary

controlling logical unit. In VTAM, a logical unit with which a secondary logical unit (other than an application program) is automatically put in session whenever the secondary logical unit is available. A controlling logical unit can be either an application program or a device-type logical unit. See also automatic logon and controlling application program.

conversation. A logical connection between two transaction programs using an LU 6.2 session. Conversations are delimited by brackets to gain exclusive use of a session.

conversational monitor system (CMS). A virtual machine operating system that provides general interactive time sharing, problem solving, and program development capabilities, and operates only under control of the VM/370 control program.

converted command. An intermediate form of a character-coded command produced by VTAM through use of an unformatted system services definition table. The format of a converted command is fixed; the unformatted system services definition table must be constructed in such a manner that the character-coded command (as entered by a logical unit) is converted into the predefined, converted command format. See also unformatted.

COS. Class of service.

coupler. A device that connects a modem to a telephone network.

CP. (1) Control point. (2) VM/370 control program. (3) VM/XA migration aid control program.

CP name. A network-qualified name of a control point (CP), consisting of a network ID qualifier identifying the network (or name space) to which the CP’s node belongs, and a unique name within the scope of that network ID identifying the CP. Each type 2.1 node has one CP name, assigned to it at system-definition time. Within an APPN network, all network nodes share a common network ID. End nodes may have distinct network IDs; this allows them to connect into separate APPN networks and to manage their own name spaces independently of the rest of the network.

CP-CP sessions. The parallel sessions between two control points, using LU 6.2 protocols and a mode name of CPSVCMG, on which network services requests and replies are exchanged. Each CP of a given pair has one contention-winner session and one contention-loser session with the other.

CPB. Collection point block.

CPMT. CSA performance measurement table.

CPU. Central processing unit.

create link pack area (CLPA). An option used during IPL to initialize the link pack pageable area.

cross-domain. In SNA, pertaining to control or resources involving more than one domain.

cross-domain resource manager (CDRM). (1) In VTAM, the function in the system services control point (SSCP) that controls initiation and termination of cross-domain sessions. (2) In SNA, a CDRM is a host with VTAM running on it that manages a portion of an SNA network.

cross-network. In SNA, pertaining to control or resources involving more than one network.

cross-network session. An LU-LU or SSCP-SSCP session whose path traverses more than one SNA network.

cross-domain resource (CDRSC). In VTAM, synonym for other-domain resource.

CRPL. Copied request parameter list control block.

customer information control system (CICS). An IBM-licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

CVT. Communication vector table.

data circuit. (1) A pair of associated transmit and receive channels that provide a means of two-way data communication. (2) See also physical circuit and virtual circuit.

Notes:

1. Between data switching exchanges, the data circuit may include data circuit-terminating equipment (DCE), depending on the type of interface used at the data switching exchange.
2. Between a data station and a data switching exchange or data concentrator, the data circuit includes the data circuit-terminating equipment at the data station end, and may include equipment similar to a DCE at the data switching exchange or data concentrator location.

data circuit-terminating equipment (DCE). In a data station, the equipment that provides the signal conversion and coding between the data terminal equipment (DTE) and the line. (1)

Notes:

1. The DCE may be separate equipment or an integral part of the DTE or of the intermediate equipment.
2. A DCE may perform other functions that are usually performed at the network end of the line.

data control block (DCB). A control block used by access method routines in storing and retrieving data.

data flow control (DFC). In SNA, a request/response unit (RU) category used for requests and responses exchanged between the data flow control layer in one half-session and the data flow control layer in the session partner.

Notes:

1. In SNA, the sequencing rules for requests and responses by which network addressable units (NAUs) in a session coordinate and control data transfer and other operations; for example, bracket protocol. See also data link control protocol.
**Glossary**

**data forwarding.** The transmission of data from the NPM host to a workstation.

The number of data forwarding requests indicates the number of workstations requesting real-time data from the NPM host about a particular resource.

**data host.** Synonym for data host node. Contrast with communication management configuration host.

**data host node.** In a communication management configuration, a type 5 host node that is dedicated to processing applications and does not control network resources, except for its channel-attached or communication adapter-attached devices. Synonymous with data host. Contrast with communication management configuration host node.

**data link.** In SNA, synonym for link.

**data link control (DLC).** A set of rules used by nodes on a data link (such as an SDLC link or a token ring) to accompany an orderly exchange of information.

**data link control (DLC) layer.** In SNA, the layer that consists of the link stations that schedule data transfer over a link between two nodes and perform error control for the link. Examples of data link control are SDLC for serial-by-bit link connection and data link control for the System/370 channel.

**Note:** The DLC layer is usually independent of the physical transport mechanism and ensures the integrity of data that reaches the higher layers.

**data link control (DLC) protocol.** In SNA, a set of rules used by two nodes on a data link to accomplish an orderly exchange of information. Synonymous with line control.

**data link level.** In the hierarchical structure of a data station, the conceptual level of control or processing logic between high level logic and the data link that maintains control of the data link. The data link level performs such functions as inserting transmit bits and deleting receive bits; interpreting address and control fields; generating, transmitting, and interpreting commands and responses; and computing and interpreting frame check sequences. See also higher level, packet level, and physical level.

**data packet.** In X.25 communications, a packet used for the transmission of user data on a virtual circuit at the DTE/DCE interface.

**data service request block (DSRB).** The control block in the NetView program that contains information that a data services command processor (DSCP) needs to communicate with the data services task (DST).

**data services task (DST).** The NetView program subtask that gathers, records, and manages data in a VSAM file, or a network device that contains network management information.

**data set.** The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

**data set members.** Members of partitioned data sets that are individually named elements of a larger file that can be retrieved by name.

**data terminal equipment (DTE).** That part of a data station that serves as a data source, data sink, or both. (1) (A)

**data types.** In the NetView program, a description of the organization of panels. Data types are alerts, events, and statistics. Data types are combined with resource types and display types to describe the NetView program's display organization. See also display types and resource types.

**DataView.** DataView window.

**DataView window.** A window in NPM Desk/2 that contains real-time or historical data sent from an NPM host.

**DBCS.** Double-byte character set.

**DC.** Data chaining.

**DCB.** (1) Data control block. (2) Device control block.

**DCE.** (1) Data circuit-terminating equipment. (2) Data communication equipment.

**DCE clear-confirmation packet.** A call supervision packet that a data circuit-terminating equipment (DCE) transmits to confirm that a call has been cleared.

**DCS.** Dynamic configuration support.

**DCSS.** Discontiguous shared segment.

**DD.** Data definition.

**ddname.** Data definition name.

**DDR.** (1) Dynamic device reconfiguration. (2) Dynamic definite response. (3) Dynamic data reconfiguration.

**dealocate.** A logical unit (LU) 6.2 application program interface (API) verb that terminates a conversation, thereby freeing the session for a future conversation. Contrast with allocate.

**DEC.** Decimal.

**DEF.** Destination element field
defered collection. In NPM, the process of collecting data at a later time about a resource.

**definite response (DR).** In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request to return a response unconditionally, whether positive or negative, to that request chain. Contrast with exception response and no response. See also negative response and positive response.

**definition statement.** (1) In VTAM, the statement that describes an element of the network. (2) In NCP, a type of instruction that defines a resource to the NCP.

**definition statement identifier.** A specific character string that identifies the purpose of a definition statement.

**DELTA disk.** The virtual disk in a VM operating system that contains program temporary fixes (PTFs) that have been installed but not merged. See BASE disk, MERGE disk, RUN disk, and ZAP disk.

**Desk/2.** NetView Performance Monitor Desk/2.

**destination subarea.** In SNA, the location to which information is to be sent.

**device-type logical unit.** In VTAM, a logical unit that has a session limit of 1 and usually acts as the secondary end of a session. It is typically a logical unit (LU) in an SNA terminal, such as a 3270. It can be the primary end of a session; for example, the logical unit representing the Network Routing Facility (NRF) logical unit.

**diagnostic trace header (DTH).** Data, in hexadecimal format, that identifies the source of information to which it belongs.

**diat-out.** Pertaining to the direction in which a switched connection is requested by a host or an NCP.

**direct access storage device (DASD).** A device in which access time is effectively independent of the location of the data.

**direct activation.** In VTAM, the activation of a resource as a result of an activation command specifically naming the resource. See automatic activation. Contrast with indirect activation.

**directed Locate search.** A search request sent to a specific destination node known to contain a resource, such as a logical unit, to verify the continued presence of the resource at the destination node.
and to obtain the node's connectivity information for route calculation. Contrast with broadcast search.

A directed search uses information stored in the directory service database of a network node server to direct the search to the location of the requested LU. A directed search is sent to the node recorded as the owner of the requested LU to verify the information. A directed search can be sent to a network node server from an end node or from a network node server. A directed search can also be sent from a network node to a central directory server when the network node server does not have information on the location of the destination logical unit or following a failed search. A directed search is originated by a network node server for the LUs it serves, by a central directory server as a result of a search request from a network node, or by an end node to its network node server. A directed search is needed when a network node server or a central directory server receives a request for an LU for which it has a location stored in its database. The search is used to verify the information. A directed search is also used by a network server to send a search request to a central directory server if the network node server has no information on the location of a requested resource. A directed search is used by an end node to request location information about a resource from the end node's network node server.

directory. (1) In VM/SP, a control program (CP) disk file that defines each virtual machine's normal configuration: the user ID, password, normal and maximum allowable virtual storage, CP command privilege classes allowed, dispatching priority, logical editing symbols to be used, account number, and CP options desired. (2) A database in an APPN node that lists names of resources (in particular, logical units) and records the CP name of the node where each resource is located. See distributed directory database and local directory database.

directory services (DS). A control point component of an APPN node that maintains a directory of the location of network resources and manages searches of that directory.

discarded packet. A packet that is intentionally destroyed.

discontiguous shared segment (DCSS). An area of virtual storage outside the address range of a virtual machine. It can contain read-only data or reentrant code. It connects discontiguous segments to a virtual machine's address space so programs can be fetched.

disk operating system (DOS). An operating system for computer systems that use disks and diskettes for auxiliary storage of programs and data.

display levels. Synonym for display types.

display types. In the NetView program, a concept to describe the organization of panels. Display types are defined as total, most recent, user action, and detail. Display types are combined with resource types and data types to describe NetView's panel organization. See also data types and resource types. Synonymous with display levels.

distributed directory database. The complete listing of all the resources in the network as maintained in the individual directories scattered throughout an APPN network. Each node has a piece of the complete directory, but it is not necessary for any one node to have the entire list. Entries are created, modified, and deleted through system definition, operator action, automatic registration, and ongoing network search procedures. Synonymous with distributed network directory and network directory database.

distributed network directory. Synonym for distributed directory database.

Distributed Processing Control Executive (DPCX). An IBM licensed program designed to control the IBM 8100 Information System. DPCX manages an 8100 system resources (deviceremaining processing time) in such a way as to provide central control of a distributed processing system and application programming capability.

Distributed Processing Programming Executive (DPPX). A comprehensive collection of licensed programs that make up an operating system for 8100 Information System hardware.

dump. (1) To record, at a particular instant, the contents of all or part of one storage device in another storage device. Dumping is usually for the purpose of debugging. (T) (2) Data that has been dumped. (T) (3) To copy data in a readable format from main or auxiliary storage onto an external medium such as tape, diskette, or printer. (4) To copy the contents of all or part of virtual storage for the purpose of collecting error information.

duplex. Pertaining to communication in which data can be sent and received at the same time. Synonymous with full duplex. Contrast with half duplex.

dLBB. In VM CMS, a command to define and identify VSAM catalogs, clusters, and data spaces; to identify VSAM files used for program input/output; and to identify input/output files for AMSERV.

dLC. Data link control.

dM. Distribution Manager.

dNc. Dynamic network collection.

domain. (1) That part of a computer network in which the data processing resources are under common control. (T) (2) In SNA, see end node domain, network node domain, and system services control point domain.

domain operator. In a multiple-domain network, the person or program that controls operation of resources controlled by one system services control point (SSCP). See also network operator.

dOS. Disk operating system.

double-byte character set (DBCS). A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires 2 bytes, the typing, display, and printing of DBCS characters requires hardware and programs that support DBCS. Contrast with single-byte character set.

DPCX. Distributed Processing Control Executive.

DPN. Destination program name.

DPPX. Distributed Processing Programming Executive.

DR. (1) In VTAM, NCP, and CCP, dynamic reconfiguration. (2) In SNA, definite response.

DRN. Dynamic reconfiguration notification.

dS. (1) Desired state. (2) Directory services.

dSA. Destination subarea field.

dSECT. Dummy control section.

dSF. Data service facility.

dSN. Data set name.

dSNAME. Data set name.

DSRB. Data service request block.

DSRB. Data service request block output.

DSRB. Data services request block unsolicited.

dST. Data services task.

dTE. Data terminal equipment. (A)

DTE/DCE interface. The physical interface and link access procedures between a data terminal equipment (DTE) and a data circuit-terminating equipment (DCE).

DTH. Diagnostic trace header.
Glossary

**DWCT.** Down-count.

**dynamic configuration support (DCS).** In NPM, a function which allows users to change NCP definitions.

**dynamic data display.** Indicates that the REAL-TIME DataView window is refreshed automatically as soon as new data arrives.

**dynamic definite response (DDR).** A facility that allows a demountable volume to be moved and repositioned if necessary without abnormally terminating the job or repeating the initial program load procedure.

**dynamic network collection (DNC).** The NPM subsystem that collects, monitors, and displays data from within the network control program and has an automatic start and stop.

**dynamic reconfiguration (DR).** In VTAM, the process of changing the network configuration (peripheral PUs and LUs) without regenerating complete configuration tables.

**dynamic reconfiguration notification (DRN).** In VTAM and NCP, a notification to NPM from the NCP that DR changes have occurred.

**E.**

**E/T.** Error-to-traffic.

**EA.** Element address.

**EAS.** Extended area service.

**EBCDIC.** Extended binary-coded decimal interchange code. A coded character set consisting of 8-bit coded characters. (A)

**EC.** (1) Engineering change. (2) Extended control.

**ECB.** Event control block.

**ECMODE.** Extended control mode.

**ECSA.** (1) Extended common storage area. (2) Extended common service area.

**EIA.** Electronics Industries Association.

**EID.** Event identifier.

**element.** (1) A field in the network address. (2) In SNA, the particular resource within a subarea that is identified by an element address. See **subarea**.

**element address (EA).** In SNA, a value in the element address field of the network address identifying a specific resource within a subarea. See **subarea address**.

**Emulation Program (EP).** An IBM control program that allows a channel-attached 3705 or 3725 communication controller to emulate the functions of an IBM 2701 Data Adapter Unit, an IBM 2702 Transmission Control, or an IBM 2703 Transmission Control. See also network control program.

**EN.** End node.

**ENCP.** End-node control point.

**end bracket.** In SNA, the value (binary 1) of the end bracket indicator in the request header (RH) of the first request of the last chain of a bracket; the value denotes the end of the bracket. Contrast with **begin bracket**. See also bracket.

**end node (EN).** A node that provides full SNA end-user services and supports sessions between its local control point (CP) and the CP in an adjacent network node, to dynamically register its resources with the adjacent CP (its network node server), to send and receive directory search requests, and to obtain management services; it can also attach to a subarea network as a peripheral node or to other end nodes.

**See type 2.1 end node.**

**end node domain.** An end node control point, its attached links, and its local LUs.

**endpoint TG vector.** A pair of control vectors representing a transmission group (TG) available at an end node for use by sessions that terminate in the node. Such a TG is not in the network topology database, and one of the purposes of the Locate/CD-Initiate search is to obtain the endpoint TG vectors for the nodes containing the origin and destination LUs (when one or both resides at an end node) of a particular session. Endpoint TG vectors are carried in the CD-Initiate GDS variables sent to the origin LU's network node server, which combines these endpoint TG vectors with the node and TG data in the topology database to compute a preferred session route between the origin and destination LUs.

**Enterprise Systems Connection (ESCON).** A set of IBM products and services that provides a dynamically connected environment within an enterprise.

**entry point.** In SNA, a type 2.0, type 2.1, type 4, or type 5 node that provides distributed network management support. It sends network management data about itself and the resources it controls to a focal point for centralized processing, and it receives and executes focal-point initiated commands to manage and control its resources.

**EOF.** End of file.

**EP.** Emulation Program.

**EPL.** (1) Exit program link. (2) European program library.

**ER.** (1) Explicit route. (2) Exception response.

**ERI.** Exception response indicator.

**error-to-traffic (E/T).** The number of temporary errors compared to the traffic associated with a resource.

**ESCON.** Enterprise Systems Connection.

**ESS.** (1) Ethernet transmission subsystem. (2) Ethernet subsystem.

**ESTAE.** Extended specify task abnormal exit.

**Ethernet.** A 10-megabit baseband local area network that allows multiple stations to access the transmission medium at will without prior coordination, avoids contention by using carrier sense and deference, and resolves contention by using collision detection and transmission. Ethernet uses carrier sense multiple access with collision detection (CSMA/CD).

**EUI.** End-user interface.

**event.** In the NetView program, a record indicating irregularities of operation in physical elements of a network.

**event control block (ECB).** A control block used to represent the status of an event.

**event message.** A message created in response to an event.

**EVL.** Environment vector list.

**exception response (ER).** In SNA, a protocol requested in the form-of-response-requested field of a request header that directs the receiver to return a response only if the request is unacceptable as received or cannot be processed; that is, a negative response, but not a positive response, may be returned. Contrast with **definite response** and **no response**. See also **negative response** and **positive response**.

**EXEC.** (1) In a VM operating system, a user-written command file that contains CMS commands, other user-written commands, and execution control statements, such as branches. (2) In NPM, a user-written command file that contains NPM commands that can be run at NPM startup, online, or through the console. Also called NPM EXEC.

**exit program.** Synonym for **exit routine**.

---

274 NPM Installation and Customization
exit routine. Either of two types of routines: installation exit routines or user exit routines. Synonymous with exit program. See installation exit routine and user exit routine.

expedited flow. In SNA, a data flow designated in the transmission header (TH) that is used to carry network control, session control, and various data flow control request/response units (RUs); the expedited flow is separate from the normal flow (which carries primarily end-user data) and can be used for commands that affect the normal flow. Contrast with normal flow.

Note: The normal and expedited flows move in both the primary-to-secondary and secondary-to-primary directions. Requests and responses on a given flow, whether normal or expedited, usually are processed sequentially within the path, but the expedited flow traffic may be moved ahead of the normal-flow traffic within the path at queuing points in the half-sessions and for half-session support in boundary functions.

explicit command. In the NetView program, a command that is used to start an operation or to request information. The use of an explicit command prevents the user from having to step through the panel hierarchy to do these tasks.

explicit route (ER). In SNA, a series of one or more transmission groups that connect two subarea nodes. An explicit route is identified by an origin subarea address, a destination subarea address, an explicit route number, and a reverse explicit route number. Contrast with virtual route (VR).

EXT. External trace file.

extended architecture (XA). An extension to System/370 architecture that takes advantage of continuing high performance enhancements to computer system hardware.

extended border node. The extended border node allows the connection of network nodes with different net ID subnetworks that need not to be adjacent.

extended control mode (ECMODE). A mode in which all features of a System/370 computing system, including dynamic address translation, are operational.

extended recovery facility (XRF). A facility that minimizes the effect of failures in MVS, VTAM, the host processor, or high availability applications during sessions between high availability applications and designated terminals. This facility provides an alternate subsystem to take over sessions from the failing subsystem.

extended specify task abnormal exit (ESTAE). An MVS macroinstruction that provides recovery capability and gives control to the user-specified exit routine for processing, diagnosing an abend, or specifying a retry address.

F

fast path. In SAA Basic Common User Access architecture, a method of doing something more directly and quickly than the usual way; for example, pressing a function key is faster than typing a command.

FAT. File allocation table.

FBA. Fixed-block architecture.

FDB. File description block.

FDX. Full duplex.

fetch protection. A feature that determines right-of-access to main storage. A protection key that is associated with a fetch reference to main storage must match a storage key associated with each block of main storage.

FF. Frames forwarded.

FFST. First Failure Support Technology.

FIC. First-in-chain.

FID. Format identification.

field-formatted. Pertaining to a request or response that is encoded into fields, each having a specified format such as binary codes, bit-significant flags, and symbolic names. Contrast with character-coded.

field-formatted request. In SNA, a request that is encoded into fields, each having a specified format such as binary codes, binary counts, bit-significant flags, and symbolic names; a format indicator in the request/response header (RH) for the request is set to zero.

file allocation table. An index to one or more disk allocation blocks in which a file is located so that the file can be read in a random or sequential manner.

filter. In the NetView program, a function that limits the data that is to be recorded on the database and displayed at the terminal. See recording filter and viewing filter.

First Failure Support Technology (FFST). A software support architecture that provides automatic reporting, recording, and notification of software incidents.

First Failure Support Technology/2 (FFST/2). An OS/2 software support tool distributed in the Extended Services Package and Communication Manager/2 which provides automatic reporting, recording, and notification of software incidents.

first speaker. See first-speaker session.

first-in-chain (FIC). A request unit (RU) whose request header (RH) begin chain indicator is on and whose RH end chain indicator is off. See also RU chain.

first-speaker session. The half-session defined at session activation as: (a) able to begin a bracket without requesting permission from the other half-session to do so, and (b) winning contention if both half-sessions attempt to begin a bracket simultaneously. Synonym for contention-winner session. Contrast with bidder session.

flow control. In SNA, the process of managing the rate at which data traffic passes between components of the network. The purpose of flow control is to optimize the rate of flow of message units with minimum congestion in the network; that is, to neither overflow the buffers at the receiver or at intermediate routing nodes, nor leave the receiver waiting for more message units. See also adaptive session-level pacing, pacing, and session-level pacing.

FMD. Function management data.

FMH. Function management header.

FMID. Function modification identifier.

FNA. Free network address.

FNF. (1) Frames not forwarded. (2) Fill non-fill.

focal point. (1) See management services focal point. (2) In the NetView program, the focal point domain is the central host domain. It is the central control point for any management services element containing control of the network management data.

format identification (FID) field. In SNA, a field in each transmission header (TH) that indicates the format of the TH; that is, the presence or absence of certain fields. TH formats differ in accordance with the types of nodes between which they pass. There are six FID types:

1. FID0, used for traffic involving non-SNA devices between adjacent subarea nodes when either or both nodes do not support explicit route and virtual route protocols
2. FID1, used for traffic involving SNA devices between adjacent subarea nodes when either or both nodes do not support explicit route and virtual route protocols
Glossary

3. FID2, used for traffic between a subarea node and an adjacent type 2 peripheral node
4. FID3, used for traffic between a subarea node and an adjacent type 1 peripheral node
5. FID4, used for traffic between adjacent subarea nodes when both nodes support explicit route and virtual route protocols
6. FIDF, used for certain commands (for example, for transmission group control) sent between adjacent subarea nodes when both nodes support explicit route and virtual route protocols.

formatted system services. A portion of VTAM that provides certain system services as a result of receiving a field-formatted command, such as an Initiate or Terminate command. Contrast with unformatted system services (USS). See also field-formatted.

frame. (1) The unit of transmission in some local area networks, including the IBM Token-Ring Network. It includes delimiters, control characters, information, and checking characters. (2) In SDLC, the vehicle for every command, every response, and all information that is transmitted using SDLC procedures.

FRR. Functional recovery routine.

FSB. File service buffer.

FST. File service task.

full duplex (FDX). Synonym for duplex.

function management data (FMD). An RU category used for end-user data exchanged between logical units (LUs) and for requests and responses exchanged between network services components of LUs, PUs, and control points.

function management header (FMH). One or more headers, optionally present in the leading request units (RUs) of an RU chain, that allow one LU to: (a) select a transaction program or device at the session partner and control the way in which the end-user data it sends is handled at the destination, (b) change the destination or the session partner and control the way in which the end-user data it sends is handled at the destination, (c) transmit between session partners status or user information about the destination (for example, a program or device). Function management headers can be used with LU type 1, 4, and 6.2 protocols.

gateway. (1) A functional unit that interconnects two computer networks with different network architectures. A gateway connects networks or systems of different architectures. A bridge interconnects networks or systems with the same or similar architectures. (T) (2) In the AIX operating environment, an entity that operates above the link layer and translates, when required, the interface and protocol used by one network into those used by another distinct network. (3) In TCP/IP, a device used to connect two systems that use either the same or different communications protocols. (4) The combination of machines and programs that provide address translation, name translation, and system services control point (SSCP) rerouting between independent SNA networks to allow those networks to communicate. A gateway consists of one gateway NCP and at least one gateway VTAM. (5) In the IBM Token-Ring Network, a device and its associated software that connect a local area network to another local area network or a host that uses different logical link protocols.

gateway NCP. An NCP that performs address translation to allow cross-network session traffic. The gateway NCP connects two or more independent SNA networks. Synonymous with gateway node.

gateway node. Synonym for gateway NCP.

gateway SSCP. Synonym for gateway VTAM.

gateway VTAM. An SSCP that is capable of cross-network session initiation, termination, takedown, and session outage notification. A gateway VTAM is in session with the gateway NCP; it provides network name translation and assists the gateway NCP in setting up alias network addresses for cross-network sessions. Synonymous with gateway SSCP.

GBL. VTAM global (resource type).

GCB. Graphics control block.

GCS. Group control system.

GDDM. Graphical Data Display Manager.

GDS variable. General data stream variable.

general data stream (GDS) variable. A type of RU substructure that is preceded by an identifier and a length field and includes either application data, user control data, or SNA-defined control data.

generalized path information unit trace (GPT). A record of the flow of path information units (PIUs) exchanged between the network control program and its attached resources. PIU trace records consist of up to 44 bytes of transmission header (TH), request/response header (RH), and request/response unit (RU) data.

generalized trace facility (GTF). An optional OS/VS service program that records significant system events, such as supervisor calls and start I/O operations, for the purpose of problem determination.

generation. The process of assembling and linking code. Definition statements so that resources can be identified to all the necessary programs in a network.

generation definition. The definition statement of a resource used in generating a program.

generic alert. A product-independent method of encoding alert data by means of both of one code points indexing short units of stored text and (2) textual data.

generic unbind. Synonym for session deactivation request.

global character. Synonym for pattern-matching character.

GMF. Graphical Monitor Facility.

GMT. Greenwich mean time.

GPT. Generalized path information unit trace.

Graphical Data Display Manager (GDDM). In the NetView Performance Monitor (NPM), an IBM-licensed program used in conjunction with the Presentation Graphics Feature (PGF) to generate online graphs in the NPM Graphic Subsystem.

group. In the NetView/PC program, to identify a set of application programs that are to run concurrently.

group control system (GCS). A component of VM that provides multiprogramming and shared memory support to virtual machines. It is a saved system intended for use with SNA products.

GSA. (1) General subsystem attachment. (2) General save area.

GTF. Generalized trace facility.

H

half-duplex (HD, HDX). In data communication, pertaining to transmission in only one direction at a time. Contrast with duplex. See also half-duplex operation and half-duplex transmission.

half-duplex operation. A mode of operation of a data link in which data can be transmitted in both directions, one way at a time. (T)
half-duplex transmission. Data transmission in either direction, one direction at a time. (1) (A)

half-session. A session-layer component consisting of the combination of data flow control and transmission control components comprising one end of a session. See also session connector.

hard error. (1) An error condition on a network that requires that the network be reconfigured or that the source of the error be removed before the network can resume reliable operation. Contrast with soft error. (2) Synonym for hard failure. (T)

hard failure. An error condition on a network that requires that the network be reconfigured or that the source of the error be removed before the network can resume reliable operation. Synonymous with hard error. (T)

hardware monitor. The component of the NetView program that helps identify network problems, such as hardware, software, and microcode, from a central control point using interactive display techniques.

HEX. Hexadecimal.

hierarchy. The resource types, display types, and data types that make up the organization, or levels, in a network.

high-level data link control (HDLC). In data communication, the use of a specified series of bits to control data links in accordance with the International Standards for HDLC: ISO 3309 Frame Structure and ISO 4335 Elements of Procedures.

high-level language (HLL). A programming language that does not reflect the structure of any particular computer or operating system. For the NetView program, the high-level languages are PL/I and C.

HIPER. High-impact or pervasive APAR.

historical data. Indicates that the data displayed in the HISTORICAL DataView window was requested within a specified time span in the past from the NPM host VSAM data files. The performance data collection is not currently active. No data forwarding is required.

HLDQ. Hold queue.

HLQ. High-level qualifier.

hop. In APPN, a portion of a route that has no intermediate nodes. It consists of only a single transmission group connecting adjacent nodes.

host node. (1) A node at which a host computer is located. (T) (2) A node that provides an application program interface (API) and a common application interface. See boundary node, node, peripheral node, subarea host node, and subarea node. See also boundary function and node type.

host NPM. A host to which a user is connected that is running NPM.

host processor. (1) A processor that controls all or part of a user application network. (T) (2) A network in the processing unit in which the data communication access method resides.

host transit time. In the NetView Performance Monitor (NPM), the average time (in seconds) that all transactions spend in the host. It includes both VTAM and application time. It is also reported as an average for the transactions originating at the logical unit for which data collection is occurring. See operator transit time and network transit time.

I format. Information format.

I frame. Information frame.

icon. A graphic symbol, displayed on a screen, that a user can point to with a device such as a mouse in order to select a particular function or software application.
interactive chart utility (ICU). A utility provided by the graphical data display manager (GDDM) to allow basic graphic handling capability and a menu-driven generation of different forms of graphs. ICU is a part of the presentation graphics feature.

Interactive Problem Control System (IPCS). A component of VM that permits online problem management, interactive problem diagnosis, online debugging for disk-resident CP abend dumps, problem tracking, and problem reporting.

Interactive System Productivity Facility (ISPF). An IBM-licensed program that serves as a full-screen editor and dialogue manager. Used for writing application programs, it provides a means of generating standard screen panels and interactive dialogues between the application programmer and terminal user.

interchange node (ICN). A node that acts both as an APPN network node and a subarea type 5 node to transform APPN protocols to subarea protocols and vice versa. For example, one LU may be in a subarea network and its partner LU in an APPN network and an LU-LU session can be established.

interconnected networks. SNA networks connected by gateways.

interconnection. See SNA network interconnection (SNI).

intermediate routing function (IRF). A capability within a node that allows it to receive and route path information units (PIUs) that neither originate from, nor are destined for, network accessible units (NAUs) in that node.

intermediate routing network. See APPN intermediate routing network.

intermediate routing node (IRN). A node containing intermediate routing function.

intermediate session routing (ISR). A type of intermediate routing function provided by an APPN network node that provides session-level outage reporting and flow control for all routes passing through it.

Internet Protocol (IP). A protocol used to route data from its source to its destination in an Internet environment.

INTQ. Intermediate queue.

intranode routing. The capability of path control to route PIUs for sessions between NAUs that reside in the same node.

IP. Internet Protocol.

IPCS. Interactive Problem Control System.

IPF. Information Presentation Facility

IPL. (1) Initial program loader. (A) (2) Initial program load.

IPX. Internetwork package exchange

IRN. Intermediate routing node.

ISD. IBM software distribution.

ISPF. Interactive System Productivity Facility.

ISTATUS. In VTAM and NCP, a definition specification method for indicating the initial status of resources. See also indirect activation.

IUCV. Inter-user communication vehicle.

J

JCL. Job control language.

JES. Job entry subsystem (MVS).

job control language (JCL). A control language used to identify a job to an operating system and to describe the job's requirements.

K

key-encrypting key. In computer security, a key used for encryption and decryption of other keys.

key-sequenced data set (KSDS). A VSAM file or data set whose records are loaded in key sequence and controlled by an index. The NetView Performance Monitor (NPM) uses this type of file for the session statistics file and the network review file.

keyword. (1) A name or symbol that identifies a parameter. (2) The part of a command operand that consists of a specific character string (such as DSNAME=). See also definition statement and keyword operand. Contrast with positional operand.

keyword operand. An operand that consists of a keyword followed by one or more values (such as DSNAME=HELLO). See also definition statement. Contrast with positional operand.

keyword parameter. A parameter that consists of a keyword followed by one or more values.

KSDS. Key-sequenced data set.

L

LAN. Local area network.

LAN Manager. An IBM program that manages LAN networks.

LAN network manager. An IBM program that manages LAN networks.

LAP. Link access procedure.

LAPB. Link access protocol-balanced.

last-in-chain (LC). A request unit (RU) whose request header (RH) end chain indicator is on and whose RH begin chain indicator is off. See also RU chain.

layer. In SNA, a grouping of related functions that are logically separate from the functions in other groups. Implementation of the functions in one layer can be changed without affecting functions in other layers.

LDLC. LAN data link control.

LEN. Low-entry networking.

LEN end node. See low-entry networking (LEN) end node.

LEN node. A type 2.1 node that supports independent LU protocols but does not support CP-CP sessions. It may be a peripheral node attached to a boundary node in a subarea network, an end node attached to an APPN network node in an APPN network, or a peer-connected node directly attached to another LEN node or APPN end node. See also LEN end node.

LC. (1) Last-in-chain. (2) In NCP, line interface coupler.

line. The portion of a data circuit, external to data circuit-terminating equipment (DCE), that connects the DCE to a data switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DCE to another DSE. (1) Synonymous with channel and circuit. See also telecommunication line.
line control. Synonym for data link control protocol.

line control discipline. Synonym for link protocol.

line discipline. Synonym for link protocol.

line speed. The number of binary digits that can be sent over a telecommunication line in one second, expressed in bits per second (bps).

line switching. Synonym for circuit switching.

link. In SNA, the combination of the link connection and the link stations joining network nodes, for example a System/370 channel and its associated protocols or a serial-by-bit connection under the control of Synchronous Data Link Control (SDL). A link connection is the physical medium of transmission. A link, however, is both logical and physical. Synonymous with data link.

link access procedures (LAP). The link level elements used for data interchange between data circuit-terminating equipment (DCE) and data terminal equipment (DTE) operating in user classes of service 8 to 11, as specified in CCITT Recommendation X.1.

link access protocol-balanced (LAPB). A protocol used for accessing an X.25 network at the link level. LAPB is a duplex, asynchronous, symmetric protocol, used in point-to-point communication.

link connection. The physical equipment providing two-way communication between one link station and one or more other link stations; for example, a telecommunication line and data circuit-terminating equipment (DCE). Synonymous with data circuit.

link connection segment. A portion of the configuration that is located between two resources listed consecutively in the service point command service (SPCS) query link configuration request list.

link level. A part of Recommendation X.25 that defines the link protocol used to get data into and out of the network across the full-duplex link connecting the subscriber's machine to the network node. LAP and LAPB are the access protocols recommended by the CCITT. See data link level.

link protocol. (1) The rules for sending and receiving data at the link level. (2) See protocol. (3) See also link level.

link station. (1) The hardware and software components within a node representing a connection to an adjacent node over a specific link. For example, if node A is the primary end of a multipoint line that connects to three adjacent nodes, node A will have three link stations representing the connections to the adjacent nodes. See also adjacent link station. (2) In VTAM, a named resource within a subarea node that represents another subarea node that is attached by a subarea link. In the resource hierarchy, the link station is subordinate to the subarea link.

link-attached. Pertaining to devices that are connected to a controlling unit by a data link. Contrast with channel-attached. Synonymous with remote.

LLC. Logical link control.

LMI. Local management interface.

load module. All or part of a computer program in a form suitable for loading into main storage for execution. A load module is usually the output of a linkage editor. (T)

local. Pertaining to a device accessed directly without use of a telecommunication line. Synonym for channel-attached.

local area network (LAN). (1) A computer network located on a user's premises within a limited geographical area. Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation. (T) See also wide area network. (2) A network in which a set of devices are connected to one another for communication and that can be connected to a larger network. See also token ring.

local directory database. That set of resources (LUs) in the network known at a particular node. The resources included are all those in the node's domain as well as any cache entries.

locate. Synonym for Locate/CD-Initiate.

locate search. The means directory services in a node uses to find a resource that is not in that node. The Locate search enables directory services to ask the directory services components in other APPN nodes for information on the target resource. See also broadcast search and directed Locate search.

locate/CD-Initiate. An abbreviated term for an inter-node message that contains one of the following sets of GDS Variables:

- A Locate (X'12C4'), a Find Resource (X'12CA'), and a Cross-Domain Initiate (X'12CA') GDS variable used for a network search request
- A Locate (X'12C4'), a Found Resource (X'12CB'), and a Cross-Domain Initiate (X'12CA') GDS variable used for a search reply when a network resource has been located

These message structures correspond to the CP components that perform the search of the distributed network directory and establish the session. The Locate GDS variable contains information used to control the delivery of the search messages in the network. Find and Found GDS variables contain information used in the directories: origin cache data (control point information) and search arguments (destination LU name), and located resource information, respectively. The Cross-Domain Initiate GDS variable contains endpoint TG vector information to be used in selecting the route for the session. The length of the Locate/CD-initiate message is limited to 1024 bytes.

logged-on operator. A NetView operator station task that requires a terminal and a logged-on user. Contrast with autotask.

logical channel. In packet mode operation, a sending channel and a receiving channel that together are used to send and receive data over a data link at the same time. Several logical channels can be established on the same data link by interleaving the transmission of packets.

logical link control (LLC) protocol. In a local area network, the protocol that governs the exchange of transmission frames between data stations independently of how the transmission medium is shared. (T)

Note: The LLC protocol was developed by the IEEE 802 committee and is common to all LAN standards.

logical record. (1) A set of related data or words considered to be a record from a logical viewpoint. (T) (2) In VSAM, a unit of information normally pertaining to a single subject; a logical record is the user record requested of or given to the data management function.

logical unit (LU). (1) A type of network accessible unit that enables end users to gain access to network resources and communicate with each other. (2) In SNA, a port through which an end user accesses the SNA network and the functions provided by system services control points (SSCPs). An LU can support at least two sessions—one with an SSCP and one with another LU—and may be capable of supporting many sessions with other LUs. See also network addressable unit (NAU), peripheral LU, physical unit (PU), system services control point (SSCP), primary logical unit (PLU), and secondary logical unit (SLU).

logical unit (LU) 6.2. A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface (API) consisting of structured verbs that are mapped into a product implementation.

logmode table. Synonym for logon mode table.

logon mode. In VTAM, a subset of session parameters specified in a logon mode table for communication with a logical unit. See also session parameters.
logon mode table. In VTAM, a set of entries for one or more logon modes. Each logon mode is identified by a logon mode name.

Synonymous with logmode table.

low-entry networking (LEN). A capability in type 2.1 nodes allowing them to directly attach to one another using peer-to-peer protocols and allowing them to support multiple and parallel sessions between logical units.

low-entry networking (LEN) end node. A type 2.1 end node that provides all SNA end-user services, can attach directly to other type 2.1 nodes using peer protocols, and derives network services implicitly from an adjacent network node when attached to an APPN network without a session between its local control point (CP) and another CP; it can also attach to a subarea network as a peripheral node.

low-entry networking (LEN) node. A type 2.1 node that supports independent LU protocols but does not support CP-CP sessions. It may be a peripheral node attached to a boundary node in a subarea network, an end node attached to an APPN network node in an APPN network, or a peer-connected node directly attached to another LEN node or APPN end node. See also low-entry networking (LEN) end node.

LPA. Link pack area.

LRECL. Logical record length.

LSAP. Local service access point.

LSR. Local shared resource.

LU. Logical unit.

LU group. In the NetView Performance Monitor (NPM), a file containing a list of related or unrelated logical units. The LU group is used to help simplify data collection and analysis.

LU type. The classification of an LU in terms of the specific subset of SNA protocols and options it supports for a given session, namely:

- The mandatory and optional values allowed in the session activation request
- The usage of data stream controls, function management headers (FMHs), request unit parameters, and sense data values
- Presentation services protocols such as those associated with FMH usage.

LU types 0, 1, 2, 3, 4, 6.1, 6.2, and 7 are defined.

LU type 6.2 (LU 6.2). A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface consisting of structured verbs that are mapped into a product implementation.

LU 6.2. Logical unit 6.2.

LU 6.2 session. A session that is initiated by VTAM on behalf of a logical unit (LU) 6.2 application program, or a session initiated by a remote LU in which the application program specifies that VTAM is to control the session by using the APPCCMD macroinstruction.

LU 6.2 verb. A syntactical unit in the LU 6.2 application program interface representing an operation.

LU-LU session type. Deprecated term for LU type.

LU-LU session. A session between two logical units (LUs) in an SNA network. It provides communication between two end users, or between an end user and an LU services component.

LUS. Logical unit services.

M

MAC. Medium access control.

MAC frame. A transmission frame that controls the operation of the IBM Token-Ring Network and any ring station operations that affect the ring.

machine check handler (MCH). A feature that analyzes errors and attempts recovery by retrying the failing instruction. If retry is unsuccessful, it attempts to correct the malfunction or to isolate the affected task.

macroinstruction. (1) An instruction in a source language that is to be replaced by a defined sequence of instructions in the same source language and that may also specify values for parameters in the replaced instructions. (T) (2) In assembler programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macro definition. The statements normally produced from the macro definition replace the macroinstruction in the program. See also definition statement.

maintenance services. In SNA, one of the types of network services in system services control points (SSCPs) and physical units (PUs). Maintenance services provide facilities for testing links and nodes and for collecting and recording error information. See also configuration services, management services, network services, and session services.

major node. In VTAM, a set of resources that can be activated and deactivated as a group. See minor node.

management services (MS). One of the types of network services in control points (CPs) and physical units (PUs). Management services are the services provided to assist in the management of SNA networks, such as problem management, performance and accounting management, configuration management, and change management.

management services focal point (MSFP). For any given management services discipline (for example, problem determination or response time monitoring), the control point that is responsible for that type of network management data for a sphere of control. This responsibility may include collecting, storing or displaying the data or all of these. (For example, a problem determination focal point is a control point that collects, stores, and displays problem determination data.)

MAT. Monitor address table.

MB1. Mouse button 1.

MB2. Mouse button 2.

Mbps. Megabits per second.

MCH. (1) Machine check handler. (2) Multichannel.

MCS. Multiple console support.

medium access control (MAC). For local area networks, the method of determining which device has access to the transmission medium at any time.

medium access control (MAC) protocol. In a local area network, the protocol that governs access to the transmission medium, taking into account the topological aspects of the network, in order to enable the exchange of data between data stations. (T) See also logical link control protocol.

menu bar. The part of the OS/2 main window display that contains the choices a user can make. For example, the choices on the NPM Desk/2 main window menu bar are Connection, Options, and Help.

MERGE disk. The virtual disk in the VM operating system that contains program temporary fixes (PTFs) after the VMF/MERGE EXEC is invoked. See BASE disk, DELTA disk, RUN disk, and ZAP disk.
**message generation.** In TPNS, the process of executing TPNS MSGTXT statements that generate messages from the resources being simulated by TPNS.

**message switching.** The process of receiving a message, storing it, and forwarding it to its destination unaltered. (T)

**message unit.** In SNA, the unit of data processed by any layer; for example, a basic information unit (BIU), a path information unit (PIU), or a request/response unit (RU).

**MI.** Manual intervention.

**MIC.** Middle-in-chain.

**middle-in-chain (MIC).** A request unit (RU) whose request header (RH) begin chain indicator and RH end chain indicator are both off. See also RU chain.

**migration.** The installation of a new version or release of a program to replace an earlier version or release.

**minidisk.** Synonym for virtual disk.

**minor node.** In VTAM, a uniquely defined resource within a major node. See major node and node.

**MNPS.** Multinode persistent sessions.

**mode name.** The name used by the initiator of a session to designate the characteristics desired for the session, such as traffic pacing values, message-length limits, sync point and cryptography options, and the class of service within the transport network.

**modem (modulator/demodulator).** (1) A functional unit that modulates and demodulates signals. One of the functions of a modem is to enable digital data to be transmitted over analog transmission facilities. (T) (A) (2) A device that converts digital data from a computer to an analog signal that can be transmitted on a telecommunication line, and converts the analog signal received to data for the computer.

**module.** A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to or output from an assembler, compiler, linkage editor, or executive routine. (A)

**monitor.** In the IBM Token-Ring Network, the function required to initiate the transmission of a token on the ring and to provide soft-error recovery in case of lost tokens, circulating frames, or other difficulties. The capability is present in all ring stations.

**monitor event message.** A message created in response to a value that has exceeded a user-defined threshold.

**monitor resolution message.** A message created when a value that has exceeded a user-defined threshold falls into normal range.

**mouse.** A commonly used pointing device with one or more buttons that a user presses to interact a computer system.

**MPC.** Multipath-channel.

**MRI.** Machine-readable-information.

**MS.** (1) Management services. (2) Major subvector.

**MSG.** Console messages.

**MTRC.** Message trace table.

**multipart node persistent sessions (MNPS).** MNPS provides for the recovery of hardware, VTAM, MVS, or other software failures with minimal impact to the users.

**Multiple Virtual Storage (MVS).** See MVS.

**Multiple Virtual Storage/Extended Architecture (MVS/XA).** See MVS/XA product.

**multiple-domain network.** (1) A network with more than one system services control point. (2) An APPN network with more than one network node.

**multipoint connection.** A connection established for data transmission among more than two data stations. (I) (A)

**Note:** The connection may include switching facilities.

**MVS.** (1) Multiple virtual storage. Implies MVS/370, the MVS/XA product, and the MVS/ESA product. (2) Multiple Virtual Storage, consisting of MVS/System Product Version 1 and the MVS/370 Data Facility Product operating on a System/370 processor. See also MVS/XA product.

**MVS/ESA product.** Multiple Virtual Storage/Enterprise Systems Architecture.

**MVS/XA product.** Multiple Virtual Storage/Extended Architecture product, consisting of MVS/System Product Version 2 and the MVS/XA Data Facility Product, operating on a System/370 processor in the System/370 extended architecture mode. See also MVS.

**MWA.** Message work area.

**N**

**name translation.** In SNA network interconnection, the conversion of logical unit names, logon mode table names, and class-of-service names used in one network to equivalent names for use in another network. This function can be provided through the NetView program and invoked by a gateway system services control point (SSCP) when necessary. See also alias name.

**NAU.** (1) Network accessible unit. (2) Network addressable unit.

**NC.** Network control.

**NCB.** Node control block.

**NCP.** Network Control Program.

**NCP/token ring interconnect (NTRI).** An NCP function that allows a communication controller to attach to the IBM Token-Ring Network and that provides both subarea and peripheral node DLC services in the SNA network.

**NCP/EP definition facility (NDF).** A program that is part of System Support Programs (SSP) and is used to generate a partitioned emulation program (PEP) load module or a load module for a Network Control Program (NCP) or for an Emulation Program (EP).

**NCP/Token Ring interconnect (NTRI).** An NCP function that allows a communication controller to attach to the IBM Token-Ring Network and that provides both subarea and peripheral node data link control (DLC) services in the SNA network.

**NDF.** NCP/EP definition facility.

**negative poll.** A negative response to polling in a binary synchronous terminal.

**negative response (NR).** In SNA, a response indicating that a request did not arrive successfully or was not processed successfully by the receiver. Contrast with positive response.

**NEO.** Network extension option.

**NEOPU.** Network extension option physical unit.

**NETID.** Network identifier.

**NetView command list language.** An interpretive language unique to the NetView program that is used to write command lists.

**NetView Graphic Monitor Facility.** The NetView Graphic Monitor Facility (NGMF) interface is used to monitor the performance of resources running under the NetView program.
Glossary

The NGMF uses interactive graphics to display views that represent a network, a portion of a network, or a group of networks. The NGMF runs on NetView V2R3 or a later release.

NPM Desk/2 supplies an interface to the NGMF, which enables you to invoke NPM commands from the NGMF through a context menu. These choices allow you to enter NPM Desk/2 and use the performance monitoring functions on resources shown on the NGMF views.

NetView Performance Monitor (NPM). An IBM-licensed program that collects, monitors, analyzes, and displays data relevant to the performance of a VTAM telecommunication network. It runs as an online VTAM application program.

NetView Performance Monitor Desk/2. NPM Desk/2 is the interface between an OS/2 workstation and an NPM host. NPM Desk/2 provides functions to collect and display performance data using Configuration and DataView windows.

NetView Program. An IBM-licensed program used to monitor a network, manage it, and diagnose its problems.

NetView Synergy Interface (NSI). An interface for collecting data from VTAM in the host or from other session managers. NSI is only available for MVS.

NetView/PC. A PC-based IBM-licensed program through which application programs can be used to monitor, manage, and diagnose problems in IBM Token-Ring networks, non-SNA communication devices, and voice networks.

NetView-NetView task (NNT). The task under which a cross-domain NetView operator session runs. See operator station task.

NetWare loadable module. A program that can be loaded and unloaded from server memory while the server is running.

network. A group of nodes and the links interconnecting them. See advanced peer-to-peer network, connection network, public network, SNA network, subarea network, subnet, transport network, type 2.1 intermediate routing network, type 2.1 network, and user-application network.

network accessible unit (NAU). A logical unit (LU), physical unit (PU), control point (CP), or system services control point (SSCP). It is the origin or the destination of information transmitted by the path control network. Synonymous with network addressable unit.

network address. In a subarea network, an address, consisting of subarea and element fields, that identifies a link, link station, physical unit, logical unit, or system services control point. Subarea nodes use network addresses; peripheral nodes use local addresses or local-form session identifiers (LFSIDs). The boundary function in the subarea node to which a peripheral node is attached transforms local addresses or LFSIDs to network addresses and vice versa. Contrast with network name.

network address translation. In SNA network interconnection, the conversion of the network address assigned to a logical unit in one network into an address in an adjacent network. This function is provided by the gateway NCP that joins the two networks. See also alias address and real address.

network addressable unit (NAU). Synonym for network accessible unit.

network control (NC). In SNA, a request/response unit (RU) category used for requests and responses exchanged between physical units (PUs) for such purposes as activating and deactivating explicit and virtual routes and sending load modules to adjust peripheral nodes. See also data flow control, function management data, and session control.

network control program. A program, generated by the user from a library of IBM-supplied modules, that controls the operation of a communication controller.

Network Control Program (NCP). An IBM-licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability.

network controller. A concentrator and protocol converter used with SDLC links. By converting protocols, which manage the way data is sent and received, the IBM 3710 Network Controller allows the use of non-SNA devices with an SNA host processor.

network definition facility (NDF). The facility that defines the identities and characteristics of each node in the network and the arrangement of the nodes in that system.

network directory database. Synonym for distributed directory database.

network extension option (NEO). A function provided by NCP that allows the user to define user line control resources and programmed resources.

network gateway accounting (NGA). The NetView Performance Monitor (NPM) subsystem that receives traffic information from the gateway NCP for sessions that flow throughout a network.

network identifier (NETID). A 1- to 8-byte customer-selected name or an 8-byte registered name that uniquely identifies a specific subnetwork. See also registered network ID.

network log. A file containing messages, commands, and command procedures that have been processed by the NetView program. In addition, output resulting from command procedure traces, command echoes, and other activity occurring within the NetView program appears in the network log.

network management vector transport (NMVT). A management services request/response unit (RU) that flows over an active session between physical unit management services and control point management services (SSCP-PU session).

network manager. A program or group of programs that is used to monitor, manage, and diagnose the problems of a network.

network name. (1) The symbolic identifier by which end users refer to a network accessible unit, a link, or a link station within a given network. In APPN networks, network names are also used for routing purposes. Contrast with network address. (2) In a multiple-domain network, the name of the APPL statement defining a VTAM application program. The network name must be unique across domains. Contrast with ACB name. See uninterpreted name.

network node (NN). Synonym for APPN network node. A node that offers full SNA end-user services and that can provide the following to its local LUs and client end nodes:
- Distributed directory services
- Intermediate routing services within an APPN network
- Network services

The APPN network node cooperates with other network nodes to maintain a network topology database, which is used to select optimal routes for LU-LU sessions based on requested classes of service. An APPN network node can also attach to a subarea network as a peripheral node or to other end nodes.

network node server. An APPN network node that provides network services for its local LUs and client end nodes.

network operator. (1) A person who or program that controls the operation of all or part of a network. (2) In a multiple-domain network, a person who or program that controls all the domains. Contrast with domain operator.

Network Performance Analysis Logical Unit (NPALU). In NPM, the virtual logical unit generated in an NCP with which the network subsystem communicates.

network performance analyzer (NPA). A function of NCP that collects performance data about devices. The data is recorded by NPM.
network product support (NPS). The function of the NetView program that provides operations control for the IBM 3710 Network Controller, the 5860 family of modems, and the NCP and provides configuration of 3710s and the 5860 family of modems. NPS provides operator commands for running diagnostics for link problems, determination and for changing product operating parameters.

network services. (1) The services within network accessible units that control network operation through SSCP-SSCP, SSCP-PU, SSCP-LU, and CP-CP sessions. (2) The session services directory and route-selection functions) and management services provided by an APPN network-node control point to its domain.

network session accounting (NSA). The NetView Performance Monitor (NPM) subsystem that receives session accounting information from the NCP for sessions that flow throughout a network.

Network Terminal Option (NTO). An IBM-licensed program, used in conjunction with NCP, that allows certain non-SNA devices to participate in sessions with SNA application programs in the host processor. When data is sent from a non-SNA device to the host processor, NTO converts non-SNA protocol to SNA protocol; and when data is sent from the host processor to the non-SNA device, NTO converts SNA protocol to non-SNA protocol.

network topology database. The representation of the current connectivity between the network nodes within an APPN network. It includes:

- Entries for all network nodes and the transmission groups interconnecting them
- Entries for all virtual routing nodes to which network nodes are attached.

network transit time. In the NetView Performance Monitor (NPM), the average time (in seconds) that all transactions spend in the network. See operator transit time and host transit time.

network-node domain. An APPN network-node control point, its attached links, the network resources for which it answers directory search requests (namely, its local LLUs and adjacent LEN end nodes), the adjacent APPN end nodes with which it exchanges directory search requests and replies, and other resources (such as a local storage device) associated with its own node or an adjacent end node for which it provides management services.

NGA. Network gateway accounting.

NGMF. NetView Graphic Monitor Facility.

NLM. NetWare loadable module.

NLS. National language support.

NMA. Network management application.

NMVT. Network management vector transport.

NNCP. Network node control point.

NNT. NetView-NetView task.

no response. In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request not to return any response, regardless of whether or not the request is received and processed successfully. Contrast with definite response and exception response.

node. (1) An endpoint of a link or junction common to two or more links in a network. Nodes can be processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities. See APPN end node, APPN network node, boundary node, host node, LEN end node, peripheral node, and subarea node. (2) In VTAM, a point in a network defined by a symbolic name. See major node and minor node.

node name. In VTAM, the symbolic name assigned to a specific major or minor node during network definition.

node type. A designation of a node according to the protocols it supports and the network accessible units that it can contain. Five types are defined: 1, 2.0, 2.1, 4, and 5. Within a subarea network, type 1, type 2.0, and type 2.1 nodes are peripheral nodes, while type 4 and type 5 nodes are subarea nodes.

non-BC. Non-broadcast.

Non-SNA Interconnection (NSI). An IBM-licensed program that provides format identification (FID) support for selected non-SNA facilities. Thus, it allows SNA and non-SNA facilities to share SDLC links. It also allows the remote concentration of selected non-SNA devices along with SNA devices.

nonswitched line. A telecommunication line on which connections do not have to be established by dialing. Contrast with switched line.

normal flow. In SNA, a data flow designated in the transmission header (TH) that is used primarily to carry end-user data. The rate at which requests flow on the normal flow can be regulated by session-level pacing. Normal and expedited flows move in both the primary-to-secondary and secondary-to-primary directions. Contrast with expedited flow.

notebook. A graphical representation that resembles a bound notebook containing pages separated into sections by tabbed divider pages. To move from one section to another, a user can turn the pages of a notebook by clicking on an arrow push button.

NOTIFY. A network services request that is sent by a system services control point (SSCP) to a logical unit (LU) to inform the LU of the status of a procedure requested by the LU.

NPA. Network performance analyzer.

NPALU. Network performance analysis logical unit.

NPM. NetView Performance Monitor.


NPM log. In NPM, a data set in which NPM data can be stored.

NPM security profile. In NPM, a data set that can be created to restrict operator access to data and functions.

NPS. Network product support.

NPSI. X.25 NCP Packet Switching Interface.

NSA. (1) Network session accounting. (2) Nonsequenced acknowledgment.

NSC. (1) Network session accounting block. (2) Native subchannel.

NSCX. NSC extensions.

NSI. (1) NetView Synergy Interface. (2) Non-SNA Interconnection.

NSS. Named saved system.

NTO. Network Terminal Option.

NTRI. NCP/Token Ring interconnect.

NVAS. NetView access services.

NW. NetWare.

OCB. Operator control block.

octet. (1) A byte that consists of 8 bits. (2) A byte composed of 8 binary elements. Synonymous with eight-bit byte.

ODAI. Origin destination assignment indicator.
operator transit time. In the NetView Performance Monitor (NPM), the sum of host and network transit times. See host transit time and network transit time.

online. (1) Pertaining to the operation of a functional unit when under the direct control of the computer. (T) (2) Pertaining to a user's ability to interact with a computer. (A) (3) Pertaining to a user's access to a computer via a terminal. (A) (4) Controlled by, or communicating with, a computer. (S) Contrast with offline.

Online Message Facility. In NPM, the facility that enables you to view help information online. It can be used as a message help tool to display the descriptions of host messages, operator actions, and system responses.

only-in-chain (OIC). A request unit for which the request header (RH) begin chain indicator and RH end chain indicator are both on. See also RU chain.

operand. (1) An entity on which an operation is performed. (I) (2) That which is operated upon. An operand is usually identified by an address part of an instruction. (A) (3) Information entered with a command name to define the data on which a command processor operates and to control the execution of the command processor. (A) (4) An expression to whose value an operator is applied. See also definition statement, keyword, keyword parameter, and parameter.

Operating System/Virtual Storage (OS/VS). A family of operating systems that control IBM System/360 and System/370 computing systems. OS/VS includes VS1, VS2, MVS/370, and MVS/XA.

operator profile. (1) In the NetView program, the resources and activities over which a network operator has control. The statements defining these resources and activities are stored in a file that is activated when the operator logs on. (2) In NPM, the setup definitions defining these resources and activities are stored in a file that is activated when the operator logs on. (2) In NPM, the setup definitions for an operator's environment. This includes security access to various parts of NPM.

operator station task (OST). The NetView task that establishes and maintains the online session with the network operator. There is one operator station task for each network operator who logs on to the NetView program. See NetView-NetView task.

operator transit time. In the NetView Performance Monitor (NPM), the sum of host and network transit times. See host transit time and network transit time.

OPT. Option.

OS. Operating system.

OS/VS. Operating System/Virtual Storage.

OS/2. Operating System/2.

OSA. Origin subarea field.

OSS. Operating system services.

OST. Operator station task.

other-domain resource. A representation for a logical unit that is owned by another domain and is referenced by a symbolic name, which can be qualified by a network identifier.

outboard data link control (ODLC) resources. SDLC and serial optical channel resources that are connected to the 3746 Model 900 (expansion unit of the 3745 CCU).
that node. (3) In the NetView/PC program, a complete line in a configuration that contains all of the resources in the service point command service (SPCS) query link configuration request list.

**path control (PC).** The function that routes message units between network accessible units in the network and provides the paths between them. It converts the BIUs from transmission control (possibly segmenting them) into path information units (PIUs) and exchanges basic transmission units containing one or more PIUs with data link control. Path control differs by node type: some nodes (T2.0 and T2.1, for example) use local addresses or session identifiers for routing, and others (subarea nodes) use network addresses for routing. See **peripheral path control** and **subarea path control**. See also link, **peripheral node**, and **subarea node**.

**path control network.** Synonym for transport network.

**path information unit (PIU).** A message unit consisting of a transmission header (TH) alone, or a TH followed by a basic information unit (BIU) or a BIU segment. See also transmission header.

**pattern-matching character.** A special character such as an asterisk (*) or a question mark (?) that can be used to represent one or more characters. Any character or set of characters can replace a pattern-matching character. Synonymous with global character and wildcard character.

**PC.** (1) Personal computer. (2) Path control. (3) Program call.

**PCB.** Presentation control block.

**PCID.** Procedure-correlation identifier.

**PCNET.** PC network.

**PCT.** Presentation control table.

**PD.** Problem determination.

**PDF.** Parallel data field.

**PDI.** Problem determination information.

**PDS.** Partitioned data set.

**PDU.** Protocol data unit.

**pending active session.** In VTAM, the state of an LU-LU session recorded by the system services control point (SSCP) when it finds both logical units (LUs) available and has sent a CINIT request to the primary logical unit (PLU) of the requested session.

**PEP.** Partitioned emulation programming.

**PerfDesk facility.** The PerfDesk facility enables you to produce, save, and reuse NPM Desk/2 windows containing Configuration and DataView windows. When you create a PerfDesk file and save it, all the actions required to produce the windows are saved. The NPM Desk/2 PerfDesk facility allows you to predefine parameters to solve common problems in the network.

**performance class.** In the NetView program, a description of an objective or commitment of performance. It consists of a performance class name, boundary definitions, response time definition, response time ranges, and response time percentage objectives. Sessions may be assigned performance classes.

**peripheral border node.** The peripheral border node enables the connection of network nodes with different net IDs and session establishment between LUs in different, adjacent net ID subnetworks.

**peripheral host node.** A node that provides an application program interface (API) for running application programs but does not provide SSCP functions and is not aware of the network configuration. The peripheral host node does not provide subarea node services. It has boundary function provided by its adjacent subarea. See **boundary node**, host node, node, **peripheral node**, subarea host node, and **subarea node**. See also boundary function and node type.

**peripheral link.** A link that connects a peripheral node to a subarea node.

**peripheral logical unit (LU).** In SNA, a logical unit in a peripheral node.

**peripheral LU.** Peripheral logical unit.

**peripheral node.** A node that uses local addresses for routing and therefore is not affected by changes in network addresses. A peripheral node requires boundary-function assistance from an adjacent subarea node. A peripheral node is a type 1, 2, 0, or 2.1 node connected to a subarea node with boundary function within a subarea. See **boundary node**, host node, node, **peripheral host node**, subarea host node, and **subarea node**. See also boundary function and node type.

**peripheral path control.** The function in a peripheral node that routes message units between units with local addresses and provides the paths between them. See **path control** and **subarea path control**. See also boundary function, peripheral node, and **subarea node**.

**peripheral physical unit (PU).** In SNA, a physical unit in a peripheral node.

**peripheral PU.** Peripheral physical unit.

**permit packet.** At the interface between a data terminal equipment (DTE) and a data circuit-terminating equipment (DCE), a packet used to transmit permits over a virtual circuit.

**PGF.** Presentation graphics feature.

**physical circuit.** A circuit established without multiplexing. See also data circuit. Contrast with virtual circuit. See also data circuit.

**physical connection.** In VTAM, a point-to-point connection or multipoint connection.

**physical level.** In X.25, the mechanical, electrical, functional, and procedural media used to activate, maintain, and deactivate the physical link between the data terminal equipment (DTE) and the data circuit-terminating equipment (DCE). See **data link level** and **packet level**.

**physical unit (PU).** The component that manages and monitors the resources (such as attached links and adjacent link stations) associated with a node, as requested by an SSCP via an SSCP-PU session. An SSCP activates a session with the physical unit in order to indirectly manage, through the PU, resources of the node such as attached links. This term applies to type 2.0, type 4, and type 5 nodes only. See also **peripheral PU** and **subarea PU**.

**physical unit (PU) services.** In SNA, the components within a physical unit (PU) that provide configuration services and maintenance services for SSCP-PU sessions. See also **logical unit (LU) services**.

**PIU.** Path information unit.

**PI.** Programming language one.

**PLPA.** Pageable link pack area.

**PLU.** Primary logical unit.

**PM.** Presentation Manager.

**PMT.** Performance measurement table.

**POI.** Program operator interface.

**point and click.** A method for selecting a graphical item or positioning a cursor that is performed with a mouse or other pointing device.

**point-to-point connection.** A connection established between two data stations for data transmission. (I) (A)

**Note:** The connection may include switching facilities.
poll.  (1) To determine whether any remote device on a telecommunication line is ready to transmit data.  (2) To execute a polling sequence.  See also positive poll and negative poll.

polling.  (1) On a multipoint connection or a point-to-point connection, the process whereby data stations are invited, one at a time, to transmit.  (I) (2) Interrogation of devices for such purposes as to avoid contention, to determine operational status, or to determine readiness to send or receive data.  (A)

port.  The representation of a physical connection to the link hardware.  A port is sometimes referred to as an adapter.  There may be one or more ports controlled by a single DLC process.

positional operand.  An operand in a language statement that has a fixed position.  See also definition statement.  Contrast with keyword operand.

positive poll.  A positive response to polling in a binary synchronous terminal.

positive response.  In SNA, a response indicating that a request was received and processed.  Contrast with negative response.

POST.  Power-on self test.

PP.  Parallel print.

PPO.  Primary program operator application program.

PPT.  (1) Primary program operator interface task.  (2) Primary POI task.  (3) Program properties table.

PR.  Print error.

Prepare.  A PS header that flows as part of commit processing, indicating the partner has begun the first phase of the two-phase commit process.

presentation graphics feature (PGF).  In the NetView Performance Monitor (NPM), a feature used in conjunction with the graphical data display manager (GDDM) to generate online graphs in the NPM graphic subsystem.

presentation services.  (1) A part of the DPPX/Base that adapts the data and control conventions of one end user of a session to the requirements of the other end user of the session.  (2) In DPCX, a part of program services that provides the methods for programs to exchange data both with displays and with printers.

primary half-session.  In SNA, the half-session that sends the session activation request.  See also primary logical unit.  Contrast with secondary half-session.

primary logical unit (PLU).  In SNA, the logical unit (LU) that sends the BIND to activate a session with its partner LU.  Contrast with secondary logical unit.

primary POI task (PPT).  The NetView subtask that processes all unsolicited messages received from the VTAM program operator interface (POI) and delivers them to the controlling operator or to the command processor.  The PPT also processes the initial command specified to execute when the NetView program is initialized and timer request commands scheduled to execute under the PPT.

primary program operator application program (PPO).  A program operator application program that is authorized to receive unsolicited messages.  When the PPO is active, all unsolicited messages go to the PPO.  Conversely, when the PPO is inactive, unsolicited messages go to the system console.  There can be only one PPO in any domain.

primary session.  An extended recovery facility (XRF) session between the active application subsystem and a terminal user.

PROC.  Command procedure.

procedure-correlation identifier (PCID).  In SNA, a value used by a control point to correlate requests and replies.

PROFILE EXEC.  In VM, a special EXEC procedure with a file name of PROFILE.  The procedure is normally executed immediately after CMS is loaded into a virtual machine.  It contains CP and CMS commands that are to be issued at the start of every terminal session.

program operator.  A VTAM application program that is authorized to issue VTAM operator commands and receive VTAM operator awareness messages.  See also solicited message and unsolicited message.

program operator interface (POI).  A VTAM function that allows programs to perform VTAM operator functions.

Program status word (PSW).  An area in storage used to indicate the order in which instructions are executed, and to hold and indicate the status of the computer system.

program temporary fix (PTF).  A temporary solution or bypass of a problem diagnosed by IBM in a current unaltered release of the program.

program-to-program interface (PPI).  In the NetView program, a facility that allows user programs to send data buffers to or receive data buffers from other user programs.  It also allows system and application programs to send alerts to the NetView hardware monitor.

prompttable table.  A character you assign to represent a resource or a date when defining a PerfDesk.

protocol.  (1) A set of semantic and syntactic rules that determine the behavior of functional units in achieving communication.  (I) (2) In Open Systems Interconnection architecture, a set of semantic and syntactic rules that determine the behavior of entities in the same layer in performing communication functions.  (T) (3) In SNA, the meanings of, and the sequencing rules for, requests and responses used for managing the network, transferring data, and synchronizing the states of network components.  Synonymous with line control discipline and line discipline.  See bracket protocol and link protocol.

protocol data unit (PDU).  A unit of data in a network.  For SDLC protocols, a PDU is a path information unit.  For token-ring protocols, a PDU is an I-frame.

PS.  Presentation services.

PSH.  Presentation services header.

PSP.  Preventative service planning.

PSRB.  Presentation services request block.

PST.  (1) Process scheduling table.  (2) Program scheduling table.

PSW.  program status word.

PTF.  Program temporary fix.

PU.  Physical unit.

PU type.  (1) Deprecated term for node type.  (2) The type of physical unit in a node.

PU-PU flow.  In SNA, the exchange between physical units (PUs) of network control requests and responses.

public network.  A network established and operated by a telecommunication Administration or by a Recognized Private Operating Agency (RPOA) for the specific purpose of providing circuit-switched, packet-switched, and leased-circuit services to the public.  Contrast with user-application network.

pull-down menu.  A list of choices extending from a selected menu-bar choice that gives access to actions.  For example, if you choose the
Options choice from the NPM Desk/2 main window menu bar, the pull-down menu contains the Settings choice.

**PUT.** Program update tape.

**PWS.** Programmable workstation.

**Q**

**QBRG.** Query bridge.

**QCB.** Queue control block.

**QNET.** Query network.

**QRB.** Query resource block.

**QRI.** Queued response indicator.

**QSAM.** Queued sequential access method.

**QSEG.** Query segment.

**queued sequential access method (QSAM).** An extended version of the basic sequential access method (BSAM). When this method is used, a queue is formed of input data blocks that are awaiting processing or of output data blocks that have been processed and are awaiting transfer to auxiliary storage or to an output device.

**quiessce.** (1) To end a process by allowing operations to complete normally. (2) In a VTAM application program, for one node to stop another node from sending synchronous-flow messages.

**R**

**RACF.** Resource Access Control Facility.

**rapid transit protocol (RTP).** An efficient mechanism used by High Performance Routing (HPR) to perform end-to-end error recoveries (selective retries) and flow control.

**RC.** Return code.

**RDT.** Resource definition table.

**real address.** The address by which a logical unit (LU) is known within the SNA network in which it resides.

**real name.** The name by which a logical unit (LU), logon mode table, or class-of-service (COS) table is known within the SNA network in which it resides.

**real time.** (1) In Open Systems Interconnection architecture, pertaining to the processing of data by a computer in connection with another process outside the computer according to time requirements imposed by the outside process. This term is also used to describe systems operating in conversational mode and processes that can be influenced by human intervention while they are in progress. (1) (A) (2) In Open Systems Interconnection architecture, pertaining to an application such as a process control system or a computer-assisted instruction system in which response to input is fast enough to affect subsequent input.

**real-time data.** Indicates an active data forwarding collection. The performance information is current, and the data is displayed online in a REAL-TIME DataView window. Real-time performance data is collected at regular intervals and the DataView is refreshed automatically, as soon as the data arrives.

**receive pacing.** In SNA, the pacing of message units being received by a component. See also send pacing.

**receive ready (RR).** In communications, a data link command that indicates that a station is ready to receive protocol data units. Receive ready also acknowledges receipt of protocol data units.

**RECFM.** Record format.

**Recommendation X.25.** An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for the interface between data terminal equipment and packet-switched data networks. See also packet switching.

**Recommendation X.3.** An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for packet assembly/disassembly (PAD) in a public data network.

**record.** (1) In programming languages, an aggregate that consists of data objects, possibly with different attributes, that usually have identifiers attached to them. In some programming languages, records are called structures. (1) (2) A set of data treated as a unit. (1) (3) A set of one or more related data items grouped for processing. (4) In VTAM, the unit of data transmission for record mode. A record represents whatever amount of data the transmitting node chooses to send.

**recording filter.** In the NetView program, the function that determines which events, statistics, and alerts are stored on a database.

**RECSZ.** Record size.

**reentrant.** The attribute of a program or routine that allows the same copy of the program or routine to be used concurrently by two or more tasks.

**refresh interval.** The interval at which NPM displays the most recent data collected from VTAM. See sampling interval.

**registered network ID.** An 8-byte name included in an IBM-maintained worldwide registry that has a structured format and is assigned to a particular IBM customer to uniquely identify a specific network.

**regular command.** In the NetView program, any VTAM or NetView command that is not an immediate command and is processed by a regular command processor. Contrast with immediate command.

**release.** (1) A distribution of a new product or new function and APAR fixes for an existing product. Normally, programming support for the prior release is discontinued after some specified period of time following availability of a new release. The first version of a product is announced as Release 1, Modification Level 0. (2) In VTAM, to relinquish control of resources (communication controllers or physical units). See also resource takeover. Contrast with acquire.

**remote.** Pertaining to a system, program, or device that is accessed through a telecommunication line. Contrast with local. Synonym for link-attached.

**remote data source.** In NPM Desk/2, indicates an NPM host that is not locally connected to the network.

**remove.** In the IBM Token-Ring Network, to take an attaching device off the ring.

**request header (RH).** The control information that precedes a request unit (RU). See also request/response header (RH).

**request unit (RU).** A message unit that contains control information, end-user data, or both.

**request/response unit (RU).** A generic term for a request unit or a response unit. See request unit (RU) and response unit (RU).

**request/response header (RH).** Control information associated with a particular RU. The RH precedes the request/response unit (RU) and specifies the type of RU (request unit or response unit).

**reset.** On a virtual circuit, reinitialization of data flow control. At reset, all data in transit are eliminated.

**reset packet.** A packet used to reset a virtual circuit at the interface between the data terminal equipment (DTE) and the data circuit-terminating equipment (DCE).

**resolution message.** A message created when a value that has exceeded a user-defined threshold falls into normal range.
resource. (1) Any facility of a computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs. (2) In the NetView program, any hardware or software that provides function to the network.

Resource Access Control Facility (RACF). An IBM-licensed program that provides for access control by identifying and verifying the users of the system, by authorizing access to protected resources, by logging the detected unauthorized attempts to enter the system, and by logging the detected accesses to protected resources.

resource definition table (RDT). In VTAM, a table that describes the characteristics of each node available to VTAM and associates each node with a network address. This is the main VTAM network configuration table.

resource hierarchy. In VTAM, the relationship among network resources in which some resources are subordinate to others as a result of their position in the network structure and architecture; for example, the logical units (LUs) of a peripheral physical unit (PU) are subordinate to that PU, which, in turn, is subordinate to the link attaching it to its subarea node.

resource registration. Resource registration places information about the location of resources in a directory services database. This registration reduces broadcast searches by ensuring that a resource will be found in the directory services database. Resources can be registered to a directory database on a NN server and/or to a CDS. By registering resources to a NN server, you ensure that the resources will be found during a search. By registering resource to a CDS, you ensure that the resources will be found without requiring a broadcast search. The defaults for registration change depending on the type of resource and whether the resource is located on an end node or a network node. Independent LUs are not registered by default; VTAM end nodes register their dependent LUs with their NN server by default.

resource resolution table (RRT). In NPM, this table contains the names of network resources for which data is to be collected. The NPM RRT corresponds with an NCP and is built by NPMGEN from an NCP Stage I and an NCP RRT.

resource table. In ACF/TCAM extended networking, a main-storage table that associates each resource identifier with an external logical unit (LU) or application program.

resource takeover. In VTAM, an action initiated by a network operator to transfer control of resources from one domain to another. See takeover. See also acquire and release.

resource types. In the NetView program, a concept to describe the organization of panels. Resource types are defined as central processing unit, channel, control unit, and I/O device for one category; and communication controller, adapter, link, cluster controller, and terminal for another category. Resource types are combined with data types and display types to describe display organization. See also data types and display types.

response. In data communication, a reply represented in the control field of a response frame. It advises the primary or combined station of the action taken by the secondary or other combined station to one or more commands. See also command.

response frame. A frame transmitted by a secondary station or a frame transmitted by a combined station that contains the address of the transmitting combined station.

response header (RH). A header, optionally followed by a response unit (RU), that indicates whether the response is positive or negative and that may contain a pacing response. See also negative response, pacing response, and positive response.

response time. (1) The elapsed time between the end of an inquiry or demand on a computer system and the beginning of the response; for example, the length of time between an indication of the end of an inquiry and the display of the first character of the response at a user terminal. (1) (A) (2) For response time monitoring, the time from the activation of a transaction until a response is received, according to the response time definition coded in the performance class.

response time monitor (RTM). A feature available with certain hardware devices to allow measurement of response times, which may be collected and displayed by the NetView program.

RTP. Rapid transit protocol

response unit (RU). A message unit that acknowledges a request unit. It may contain prefix information received in a request unit. If positive, the response unit may contain additional information (such as session parameters in response to BIND SESSION). If negative, the response unit contains sense data defining the exception condition.

Restructured Extended Executor (REXX). A general-purpose, procedural language for end-user personal programming, designed for ease by both casual general users and computer professionals. It is also useful for application macros. REXX includes the capability of issuing commands to the underlying operating system from these macros and procedures. Features include powerful character-string manipulation, automatic data typing, manipulation of objects familiar to people, such as words, numbers, and names, and built-in interactive debugging.

REX. Route extension.

REXX. Restructured Extended Executor.

RH. Request/response header.

ring. See ring network.

ring network. (1) A network in which every node has exactly two branches connected to it and in which there are exactly two paths between any two nodes. (T) (2) A network configuration in which devices are connected by unidirectional transmission links to form a closed path.

RMODE. Residency mode.

RNAA. Request network address assignment.

RNR. Receive not ready.

route. An ordered sequence of nodes and transmission groups (TGs) that represent a path from an origin node to a destination node traversed by the traffic exchanged between them.

route extension (REX). In SNA, the path control network components, including a peripheral link, that make up the portion of a path between a subarea node and a network addressable unit (NAU) in an adjacent peripheral node. See also explicit route (ER), path, and virtual route (VR).

routing. (1) The process of determining the path to be used for transmission of a message over a network. (T) (2) The assignment of the path by which a message will reach its destination.

RPL. Request parameter list.

RPOA. Recognized private operating agency.

RPQ. Request for price quotation.

RR. Receive ready.

RSCV. A control vector that describes a route within an APPN network. The RSCV consists of an ordered sequence of control vectors that
identify the TGs and nodes that make up the path from an origin node to a destination node.

**RRT.** Resource resolution table.

**RTM.** Response time monitor.

**RTP.** Resource table prefix.

**RU.** Request/response unit.

**RU chain.** In SNA, a set of related request/response units (RUs) that are consecutively transmitted on a particular normal or expedited data flow. The request RU chain is the unit of recovery; if one of the RUs in the chain cannot be processed, the entire chain is discarded. Each RU belongs to only one chain, which has a beginning and an end indicated by means of control bits in request/response headers within the RU chain. Each RU can be designated as first-in-chain (FIC), last-in-chain (LIC), middle-in-chain (MIC), or only-in-chain (OIC). Response units and expedited-flow request units are always sent as only-in-chain.

**RUN disk.** The virtual disk that contains the VTAM, NetView, and VM/SNA console support (VSCS) load libraries, program temporary fixes (PTFs), and user-written modifications from the ZAP disk. See BASE disk, DELTA disk, MERGE disk, and ZAP disk.

**S**

**SA.** (1) Subarea. (2) System area.

**SAA.** Systems Application Architecture.

**SAF.** (1) System Authorization Facility. (2) Source address field.

**same-domain.** Pertaining to communication between entities in the same SNA domain. Contrast with cross-domain. See also single-domain network.

**sampling interval.** The interval at which NPM updates its own counters. This updated data is not displayed until the next refresh interval. If a sampling interval is set higher than a refresh interval, the sampling interval is automatically reset to the refresh interval. See refresh interval.

**SAP.** Service access point.

**SAW.** Session awareness.

**SC.** Session control.

**SCA.** Session communication area.

**SCB.** (1) Session control block. (2) String control byte.

**SCT.** Section control table.

**SDB.** Session data buffer.

**SDH.** Session data header.

**SDLC.** Synchronous Data Link Control.

**SDM.** Software distribution manager.

**SDSF.** System display and search facility.

**SDT.** (1) Start data traffic. (2) Session data table. See also session data.

**SDWA.** System diagnostic work area.

**secondary half-session.** In SNA, the half-session that receives the session-activation request. See also secondary logical unit (SLU). Contrast with primary half-session.

**secondary logical unit (SLU).** In SNA, the logical unit (LU) that contains the secondary half-session for a particular LU-LU session. An LU may contain secondary and primary half-sessions for different active LU-LU sessions. Contrast with primary logical unit (PLU).

**secondary program operator application program (SPO).** A program operator application program that is not authorized to receive unsolicited messages. An SPO can receive only the messages generated by commands it issues. There can be more than one SPO in a domain, in addition to a primary program operator application program (PPO). Contrast with primary program operator application program.

**segment.** (1) Synonym for BIU segment. (2) In the IBM Token-Ring Network, a section of cable between components or devices. A segment may consist of a single patch cable, several patch cables that are connected, or a combination of building cable and patch cables that are connected. (3) See link connection segment.

**segmentation.** A process by which path control (PC) divides basic information units (BIUs) into smaller units, called BIU segments, to accommodate smaller buffer sizes in adjacent nodes. Both segmentation and segment assembly are optional PC features. The support for either or both is indicated in the BIND request and response.

**SEMEA.** Southern Europe Middle East Africa.

**send pacing.** In SNA, pacing of message units that a component is sending. See also receive pacing.

**sequencing.** (1) The chronological display of the real-time graphic monitor. (2) Ordering in a series or according to rank or time.

**server.** (1) A functional unit that provides shared services to workstations over a network; for example, a file server, a print server, a mail server. (T) (2) In a network, a data station that provides facilities to other stations; for example, a file server, a print server, a mail server. (A)

**service access point (SAP).** A logical point made available by an adapter where information can be received and transmitted. A single service access point can have many links terminating in it.

**Service Level Reporter (SLR).** A licensed program that generates management reports from data sets such as System Management Facility (SMF) files.

**service point (SP).** An entry point that supports applications that provide network management for resources not under the direct control of itself as an entry point. Each resource is either under the direct control of another entry point or not under the direct control of any entry point. A service point accessing these resources is not required to use SNA sessions (unlike a focal point). A service point is needed when entry point support is not yet available for some network management function.

**service point command service (SPCS).** An extension of the command facility in the NetView program that allows the host processor to communicate with a service point by using the communication network management (CNM) interface.

**service reminder (SR).** In the NetView/PC program, a notification set by the operator that is displayed on a panel and logs a specified message.

**service transaction program.** Any IBM-supplied transaction program running in a network accessible unit. Contrast with application transaction program.

**session.** (1) In network architecture, for the purpose of data communication between functional units, all the activities that take place during the establishment, maintenance, and release of the connection. (2) In SNA, a logical connection between two network accessible units (NAUs) that can be activated, tailored to provide various protocols, and deactivated, as requested. Each session is uniquely identified in a transmission header (TH) accompanying any transmissions exchanged during the session.
session activation request. In SNA, a request that activates a session between two network accessible units (NAUs) and specifies session parameters that control various protocols during session activity; for example, BIND and ACTPTU. Contrast with session deactivation request.

session awareness (SAW) data. Data collected by the NetView program about a session that includes the session type, the names of session partners, and information about the session activation status. It is collected for LU-LU, SSCP-LU, SSCP-PU, and SSCP-SSCP sessions and for non-SNA terminals not supported by NTO. It can be displayed in various forms, such as most recent sessions lists.

session collection. The NPM subsystem that collects, monitors, and displays data collected in the host for analysis.

session connector. A session-layer component in an APPN network node or in a subarea node boundary or gateway function that connects two stages of a session. Session connectors swap addresses from one address space to another for session-level intermediate routing, segment session message units as needed, and (except for gateway function session connectors) adaptively pace the session traffic in each direction. See also half-session.

session control (SC). In SNA, either of the following:
- One of the components of transmission control. Session control is used to purge data flowing in a session after an unrecoverable error occurs, to resynchronize the data flow after such an error, and to perform cryptographic verification.
- A request unit (RU) category used for requests and responses exchanged between the session control components of a session and for session activation and deactivation requests and responses.

session control block (SCB). In NPM, control blocks in common storage area for session collection.

session cryptography key. In SNA, a data encrypting key used to encipher and decipher function management data (FMD) requests transmitted in an LU-LU session that uses cryptography.

session data. Data about sessions collected by NPM.

session deactivation request. In SNA, a request that deactivates a session between two network accessible units (NAUs); for example, UNBIND and DACTPTU. Synonymous with generic unbind. Contrast with session activation request.

session information retrieval (SIR). The function that allows an operator to enable or disable session information retrieval for a particular gateway or for all gateway sessions. When a gateway session ends, trace information about the most recent sequence or FID0 numbers to cross the gateway is passed back to all system services control points (SSCPs) that have enabled SIR for that session or for all sessions. This information can also be passed back to the requesting host.

session initiation request. In SNA, an Initiate or logon request from a logical unit (LU) to a system services control point (SSCP) that an LU-LU session be activated.

session layer. The composite layer consisting of the data flow control and transmission control layers forming the half-sessions and session connectors in the network.

session limit. (1) The maximum number of concurrently active LU-LU sessions that a particular logical unit (LU) can support. (2) In NCP, the maximum number of concurrent line-scheduling sessions on a non-SDLC, multipoint line.

session manager (SM). A product, such as NetView Access Services, that allows a user at a terminal to log on to multiple applications concurrently.

session monitor. The component of the NetView program that collects and correlates session-related data and provides online access to this information.

session parameters. In SNA, the parameters that specify or constrain the protocols (such as bracket protocol and pacing) for a session between two network accessible units. See also logon mode.

session partner. In SNA, one of the two network accessible units (NAUs) having an active session.

session path. The half-sessions delimiting a given session and their interconnection (including any intermediate session connectors).

session services. One of the types of network services in the control point (CP) and in the logical unit (LU). These services provide facilities for an LU or a network operator to request that a control point (an ENCP, NNCP, or SSCP) assist with initiating or terminating sessions between logical units. Assistance with session termination is needed only by SSCP-dependent LUs. See configuration services, maintenance services, and management services.

session statistics file. In NPM, an online VSAM key-sequenced data set (KSDS) used for storing session data.

session trace. In the NetView program, the function that collects session trace data for sessions involving specified resource types or involving a specific resource.

session trace data. Data, relating to sessions, that is collected by the NetView program whenever a session trace is started and that consists of session activation parameters, VTAM path information unit (PIU) data, and NCP data.

session-layer component. A half-session or session connector.

session-level pacing. A flow control technique that permits a receiving half-session or session connector to control the data transfer rate (the rate at which it receives request units) on the normal flow. It is used to prevent overloading a receiver with unprocessed requests when the sender can generate requests faster than the receiver can process them. See also pacing and virtual route pacing.

session-termination request. In VTAM, a request that an LU-LU session be terminated.

shared. Pertaining to the availability of a resource for more than one use at the same time.

shared-access transport facility (SATF). A transmission facility, such as a multipoint link connection, a public switched network, or a token ring, on which multiple pairs of nodes can form concurrently active links.

shortcut keys. Keys that provide keyboard options that can be used instead of a mouse. Shortcut keys are available for every NPM Desk/2 option.

shutdown. The process of ending operation of a system or a subsystem, following a defined procedure.

single-byte character set (SBCS). A character set in which each character is represented by a one-byte code. Contrast with double-byte character set (DBCS).

single-domain network. In SNA, a network with one system services control point (SSCP). Contrast with multiple-domain network.

SIO. Start I/O.

SIR. Session information retrieval.

SLR. Service Level Reporter.

SLU. Secondary logical unit.

SMF. System management facility.

SMP. System Modification Program.

SMP/E. System Modification Program Extended.

SNA. Systems Network Architecture.
SNA network. The part of a user-application network that conforms to the formats and protocols of Systems Network Architecture. It enables reliable transfer of data among end users and provides protocols for controlling the resources of various network configurations. The SNA network consists of network accessible units (NAUs), boundary function, gateway function, and intermediate session routing function components; and the transport network.

SNA network interconnection (SNI). The connection, by gateways, of two or more independent SNA networks to allow communication between logical units in those networks. The individual SNA networks retain their independence.

SNAP. System Network Analysis Program.

SNI. SNA network interconnection.

SNT. Subarea number table.

SOC. Serial optical channel.

soft error. (1) An error that occurs sporadically and that may not appear on successive attempts to read data. Synonymous with transient error. (T) (2) An intermittent error on a network that requires retransmission. Contrast with hard error.

Note: A soft error by itself does not affect overall reliability of a network, but reliability may be affected if the number of soft errors reaches the ring error limit.

solicited message. A response from VTAM to a command entered by a program operator. Contrast with unsolicited message.

SP. Service point.

span. In the NetView program, a user-defined group of network resources within a single domain. Each major or minor node is defined as belonging to one or more spans. See also span of control.

span of control. The total network resources over which a particular network operator has control. All the network resources listed in spans associated through profile definition with a particular network operator are within that operator's span of control.

SPC. Software and Publications Center (IBM Copenhagen).

SPCS. Service point command service.

SPLU. Session partner logical unit.

SPO. Secondary program operator application program.

SPX. Sequenced packet exchange.

SQA. System queue area.

SR. Service reminder.

SRB. Service request block.

SRT. System reference table.

SS. (1) Start-stop. (2) Session services.

SSB. System statistics buffer.

SSCP. System services control point.

SSCP rerouting. In SNA network interconnection, the technique used by the gateway system services control point (SSCP) to send session-initiation request units (RUs), by way of a series of SSCP-SSCP sessions, from one SSCP to another, until the owning SSCP is reached.

SSCP-independent LU. An LU that is able to activate an LU-LU session (that is, send a BIND request) without assistance from an SSCP. It does not have an SSCP-LU session. Currently, only an LU 6,2 can be an independent LU.

SSCP-LU session. In SNA, a session between a system services control point (SSCP) and a logical unit (LU). The session enables the LU to request the SSCP to help initiate LU-LU sessions.

SSCP-PU session. In SNA, a session between a system services control point (SSCP) and a physical unit (PU); SSCP-PU sessions allow SSCPs to send requests to and receive status information from individual nodes in order to control the network configuration.

SSCP-SSCP session. In SNA, a session between the system services control point (SSCP) in one domain and the SSCP in another domain. An SSCP-SSCP session is used to initiate and terminate cross-domain LU-LU sessions.

SSI. Subsystem interface.

SSP. (1) System Support Program product. (2) System service program.

SSRT. System symbol resource table.

start option. In VTAM, a user-specified or IBM-supplied option that determines certain conditions that are to exist during the time a VTAM system is operating. Start options can be predefined or specified when VTAM is started.

start-stop (SS) transmission. (1) Asynchronous transmission such that each group of signals representing a character is preceded by a start signal and is followed by a stop signal; (T) (A) (2) Asynchronous transmission in which a group of bits is (a) preceded by a start bit that prepares the receiving mechanism for the reception and registration of a character, and (b) followed by at least one stop bit that enables the receiving mechanism to come to an idle condition pending reception of the next character. See also binary synchronous transmission and synchronous data link control.

statement identifier. The lexical entity in a language statement that indicates the purpose of the statement, such as the action to be performed or the resource being defined. See also definition statement, definition statement identifier, and operator.

static. (1) In programming languages, pertaining to properties that can be established before execution of a program; for example, the length of a fixed length variable is static. (l) (2) Pertaining to an operation that occurs at a predetermined or fixed time. (3) Contrast with dynamic.

station. An input or output point of a system that uses telecommunication facilities; for example, one or more systems, computers, terminals, devices, and associated programs at a particular location that can send or receive data over a telecommunication line.

status code. In VTAM, information on the status of a resource as shown in a 10-character state code; for example, STATEACTIV for active.

status monitor. A component of the NetView program that collects and summarizes information on the status of resources defined in a VTAM domain.

STC. Started task.

STCK. Store-clock (time).

subarea. A portion of the SNA network consisting of a subarea node, attached peripheral nodes, and associated resources. Within a subarea node, all network accessible units (NAUs), links, and adjacent link stations (in attached peripheral or subarea nodes) that are addressable within the subarea share a common subarea address and have distinct element addresses.

subarea address. A value in the subarea field of the network address that identifies a particular subarea. See also element address.

subarea host node. A host node that provides both subarea function and an application program interface (API) for running application programs. It provides system services control point (SSCP) functions and subarea node services, and it is aware of the network configuration. See boundary node, communication management configuration host...
node, data host node, host node, node, peripheral node, and subarea node. See also boundary function and node type.

**subarea network.** Interconnected subareas, their directly attached peripheral nodes, and the transmission groups that connect them.

**subarea node (SN).** A node that uses network addresses for routing and maintains routing tables that reflect the configuration of the network. Subarea nodes can provide gateway function to connect multiple subarea networks, intermediate routing function, and boundary function support for peripheral nodes. Type 4 and type 5 nodes are subarea nodes. See boundary node, host node, node, peripheral node, and subarea host node. See also boundary function and node type.

**subarea path control.** The function in a subarea node that routes message units between network accessible units (NAUs) and provides the paths between them. See path control and peripheral path control. See also boundary function, peripheral node, and subarea node.

**subarea PU.** In SNA, a physical unit (PU) in a subarea node.

**switched line.** (1) Any group of nodes that have a set of common characteristics, such as the same network ID. (2) In the AIX operating system, one of a group of multiple logical network divisions of another network, such as can be created by the Transmission Control Protocol/Internet Protocol (TCP/IP) interface program.

**suboperand.** One of multiple elements in a list comprising an operand. See also definition statement.

**subsystem.** A secondary or subordinate system, usually capable of operating independently of, or asynchronously with, a controlling system. (T)

**subvector.** A subcomponent of the NMVT major vector.

**supervisor.** The part of a control program that coordinates the use of resources and maintains the flow of processing unit operations.

**supervisor call (SVC).** A request that serves as the interface into operating system functions, such as allocating storage. The SVC protects the operating system from inappropriate user entry. All operating system requests must be handled by SVCs.

**SVGA.** Super Video Graphics Array/Adapter.

**SVC.** (1) Supervisor call. (2) Switched virtual circuit.

**SVT.** System services vector table.

**switched line.** A telecommunication line in which the connection is established by dialing. Contrast with nonswitched line.

**switched major node.** In VTAM, a major node whose minor nodes are physical units and logical units attached by switched SDLC links.

**switched virtual circuit (SVC).** An X.25 circuit that is dynamically established when needed. The X.25 equivalent of a switched line.

**SYNAD.** Synchronous error routine address.

**SYNAD exit routine.** A synchronous EXLST exit routine that is entered when a physical error is detected.

**sync point.** An intermediate or end point during processing of a transaction at which an update or modification to one or more of the transaction’s protected resources is logically complete and error free. Synonymous with synchronization point.

**synchronization point.** Synonym for sync point.

**Synchronous Data Link Control (SDLC).** A discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-level Data Link Control (HDLC) of the International Organization for Standardization, for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges may be duplex or half-duplex over switched or nonswitched links. The configuration of the link connection may be point-to-point, multipoint, or loop. (I) See also binary synchronous communications.

**synchronous operation.** In VTAM, a communication, or other operation in which VTAM, after receiving the request for the operation, does not return control to the program until the operation is completed. Contrast with asynchronous operation.

**SYSMOD.** System modification.

**system definition.** The process, completed before a system is put into use, by which desired functions and operations of the system are selected from various available options. Synonymous with system generation.

**system generation.** Synonym for system definition.

**System Management Facility (SMF).** A standard feature of MVS that collects and records a variety of system and job-related information.

**System modification (SYSMOD).** The input data to SMP or SMP/E that defines the introduction, replacement, or update of elements in the operating system and associated distribution libraries to be installed under control of SMP or SMP/E. A system modification is defined by a set of modification control statements.

**System Modification Program (SMP).** A program used to install software and software changes on MVS systems.

**System Modification Program Extended (SMP/E).** An IBM-licensed program used to install software and software changes on MVS systems. In addition to providing the services of SMP, SMP/E consolidates installation data, allows more flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

**System queue area (SQA).** In OS/VS, an area of virtual storage reserved for system-related control blocks. It contains fixed pages and is assigned protection key zero.

**system services control point (SSCP).** A component within a subarea network for managing the configuration, coordinating network operator and problem determination requests, and providing directory services and other session services for end users of the network. Multiple SSCPs, cooperating as peers with one another, can divide the network into domains of control, with each SSCP having a hierarchical control relationship to the physical units and logical units within its own domain.

**system services control point (SSCP) domain.** The system services control point, the physical units (PUs), the logical units (LUs), the links, the link stations, and all the resources that the SSCP has the ability to control by means of activation and deactivation requests.

**System Support Program (SSP) product.** A group of IBM-licensed programs that manage the running of other programs and the operation of associated devices, such as the display station and printer. The SSP also contains utility programs that perform common tasks, such as copying information from diskette to disk. See Advanced Communication Function (ACF).

**Systems Application Architecture (SAA) solution.** A set of IBM software interfaces, conventions, and protocols that provide a framework for designing and developing applications that are consistent across systems.

**Systems Network Architecture (SNA).** The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks.
**T**

**TAB.** Terminal anchor block.

**takeover.** The process by which the failing active subsystem is released from its extended recovery facility (XRF) sessions with terminal users and replaced by an alternate subsystem. See resource takeover.

**TAP.** Transit analysis program.

**TCA.** Task control area.

**TCAM.** Telecommunications Access Method. Synonymous with ACF/TCAM.

**TCB.** Task control block.

**TCP.** Transmission Control Protocol.

**TDL.** Twin-axial data link control.

**telecommunication line.** (1) The portion of a data circuit external to a data circuit-terminating equipment (DCE) that connects the DCE to a data-switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DSE to another DSE. (T) (2) Any physical medium, such as a wire or microwave beam, that is used to transmit data. Synonymous with transmission line.

**Telecommunications Access Method (TCAM).** An access method used to transfer data between main storage and remote or local terminals.

**teleprocessing network simulator (TPNS).** A testing package that enables a user to test and evaluate teleprocessing systems before actual terminal installation.

**teleprocessing request block (TPRB).** A function used by NPM to control input/output requests to or from terminals to files.

**terminate.** In SNA, a request unit that is sent by a logical unit (LU) to its system services control point (SSCP) to cause the SSCP to start a procedure to end one or more designated LU-LU sessions.

**TG.** Transmission group.

**TG vector.** See transmission group vector.

**TH.** Transmission header.

**threshold.** (1) In the NetView program, a percentage value, set for a resource and compared to a calculated error-to-traffic ratio. (2) In NPM, high or low values supplied by the user to monitor data and statistics being collected.

**TIC.** Token-ring interface coupler.

**Time Sharing Option (TSO).** An operating system option; for the System/370 system, the option provides interactive time sharing from remote terminals.

**Time Sharing Option Extensions (TSO/E).** (1) The base for all TSO enhancements. It provides MVS users with additional functions, improved usability, and better performance. (2) In the MVS/ESA environment, TSO/E also provides virtual storage constraint relief.

**time-of-day clock.** A System/370 hardware feature that is incremented once every microsecond, and provides a consistent measure of elapsed time suitable for indicating date and time. The TOD clock runs regardless of whether the processing unit is in a running, wait, or stopped state.

**TOD.** Time-of-day. See time-of-day clock.

**token.** (1) In a local area network, the symbol of authority passed successively from one data station to another to indicate the station temporarily in control of the transmission medium. Each data station has an opportunity to acquire and use the token to control the medium.

A token is a particular message or bit pattern that signifies permission to transmit. (T) (2) A sequence of bits passed from one device to another along the token ring. When the token has data appended to it, it becomes a frame.

**token ring.** A network with a ring topology that passes tokens from one attaching device to another; for example, the IBM Token-Ring Network.

**token-ring interface coupler (TIC).** An adapter that can connect a 3720, 3725, or 3745 Communication Controller to an IBM Token-Ring Network.

**topology database update (TDU).** A message about a new or changed link or node that is broadcast among APPN network nodes to maintain the network topology database, which is fully replicated in each network node.

**TP.** (1) Transmission priority. (2) Transaction program.

**TPEX.** Teleprocessing executor.

**TPNS.** Teleprocessing network simulator.

**TPR.** Transaction processing routine.

**TPRB.** Teleprocessing request block.

**TPX.** A session manager interface developed by Legent Corporation.

**TR.** Trace.

**TR-TR.** Token ring to token ring.

**transaction program.** (1) A program that processes transactions in an SNA network. There are two kinds of transaction programs: application transaction programs and service transaction programs. See also conversation. (2) In VTAM, a program that performs services related to the processing of a transaction. One or more transaction programs may operate within a VTAM application program that is using the VTAM application program interface (API). In that situation, the transaction program would request services from the application program, using protocols defined by that application program. The application program, in turn, could request services from VTAM by issuing the APPCCMD macroinstruction.

**transit time.** In NPM, transit time is the same as response time. See response time, host time, network time, and operator time.

**Transmission Control Protocol (TCP).** A communications protocol used in Internet and in any network that follows the U.S. Department of Defense standards for inter-network protocol. TCP provides a reliable host-to-host protocol between hosts in packet-switched communications networks and in interconnected systems of such networks. It assumes that the Internet protocol is the underlying protocol.

**transmission control unit (TCU).** A communication control unit whose operations are controlled by programmed instructions from the computing system to which the unit is attached. No program is stored or executed in the unit. Examples are the IBM 2702 and 2703 Transmission Controls. Contrast with communication controller.

**transmission frame.** In data transmission, data transported from one node to another in a particular format that can be recognized by the receiving node. In addition to a data or information field, a frame has some kind of delimiter that marks its beginning and end and usually control fields, address information that identifies the source and destination, and one or more check bits that allow the receiver to detect errors that may occur after the sender has transmitted the frame.

**transmission group (TG).** A group of links between adjacent subarea nodes, appearing as a single logical link for routing of messages. A transmission group may consist of one or more SDLC links (parallel links) or of a single System/370 channel. In APPN, transmission group is synonymous with link.

**transmission group (TG) vector.** A representation of an endpoint TG in a T2.1 network, consisting of two control vectors: the TG Descriptor
Glossary

(X’46’) control vector and the TG Characteristics (X’47’) control vector.

transmission header (TH). Control information, optionally followed by a basic information unit (BIU) or a BIU segment, that is created and used by path control to route message units and to control their flow within the network. See also path information unit.

transmission line. Synonym for telecommunication line.

transmission priority. In SNA, a rank assigned to a path information unit (PIU) that determines its precedence for being selected by the transmission group control component of path control for forwarding to the next subarea along the route traversed by the PIU.

transmission subsystem component (TSC). The component of VTAM that comprises the transmission control, path control, and data link control layers of SNA.

transport network. The part of the SNA network that includes the data link control and path control layers. Synonymous with path control network.

TRC. Next trace entry.

TRCB. First trace entry.

TSC. (1) Telecommunications subsystem controller. (2) Transmission subsystem component.

TSCB. (1) Task set control block. (2) Transmission subsystem control block.

TSO. Time Sharing Option.

TSO/E. Time Sharing Option Extensions. A licensed program that provides enhancements to MVS/370 and MVS/XA users.

TXB. Transaction program block.

type 2.1 end node. A type 2.1 node that provides full SNA end-user services, but no intermediate routing or network services to any other node; it is configured only as an endpoint in a network. See also advanced peer-to-peer networking (APPN) end node and low-entry networking (LEN) end node.

type 2.1 intermediate routing network. Synonym for APPN intermediate routing network.

type 2.1 network. A collection of interconnected type 2.1 network nodes and type 2.1 end nodes. A type 2.1 network may consist of nodes of just one type, namely, all network nodes or all end nodes; a pair of directly attached end nodes is the simplest case of a type 2.1 network.

type 2.1 network node. Synonym for APPN network node.

type 2.1 node. An SNA node that can be configured as an endpoint or intermediate routing node in a type 2.1 network, or as a peripheral node attached to a subarea network. See also APPN end node, APPN network node, LEN end node, and type 2.1 end node.

type 4 node. A subarea node that contains a PUCP rather than an SSCP and that is controlled by one or more type 5 nodes.

type 5 node. A subarea node that contains an SSCP and that has hierarchical control of peripheral nodes and type 4 subarea nodes.

U

UIB. User identification block.

UNBIND. In SNA, a request to deactivate a session between two logical units (LUs). See also session deactivation request. Contrast with BIND.

unformatted system services (USS). In SNA products, a system services control point (SSCP) facility that translates a character-coded request, such as a logon or logoff request, into a field-formatted request for processing by formatted system services and that translates field-formatted replies and responses into character-coded requests for processing by a logical unit. Contrast with formatted system services. See also converted command.

uninterpreted name. In SNA, a character string that a system services control point (SSCP) can convert into the network name of a logical unit (LU). Typically, an uninterpreted name is used in a logon or Initiate request from a secondary logical unit (SLU) to identify the primary logical unit (PLU) with which the session is requested.

unsolicited message. A message, from VTAM to a program operator, that is unrelated to any command entered by the program operator. Contrast with solicited message.

UP. Unnumbered poll.

UPB. User profile block.

upper level qualifier. The upper level qualifier is a parameter that helps you to identify a resource by specifying information that is unique to that resource. It designates the name of the parent or predecessor of that resource, providing NPM with the exact coordinates of the object inside the configuration.

user exit. (1) A point in an IBM-supplied program at which a user exit routine may be given control. (2) A programming service provided by an IBM software product that may be requested during the execution of an application program for the service of transferring control back to the application program upon the later occurrence of a user-specified event.

user exit routine. A user-written routine that receives control at predefined user exit points. User exit routines can be written in assembler or a high-level language.

user-application network. A configuration of data processing products, such as processors, controllers, and terminals, established and operated by users for the purpose of data processing or information exchange, which may use services offered by communication common carriers or telecommunication Administrations. (T) Contrast with public network.

USS. Unformatted system services.

V

VB. Variable blocked.

VC. (1) Virtual connect. (2) Virtual circuit.

VC-type. Virtual connect-type.

VCS. Virtual circuit service.

VDT. (1) Video display terminal. (2) VTAM data table.

vector. The MAC frame information field.

verb. See LU 6.2 verb.

VGA. Video graphics adapter.

viewing filter. In the NetView program, the function that allows a user to select the alert data to be displayed on a terminal. All other stored data is blocked.

virtual circuit (VC). (1) In packet switching, the facilities provided by a network that give the appearance to the user of an actual connection. (T) See also data circuit. Contrast with physical circuit. (2) A logical connection established between two DTEs.

virtual disk. (1) In VM, a physical disk storage device, or a logical subdivision of a physical disk storage device, that has its own address, consecutive storage space for data, and index or description of stored data so that the data can be accessed. (2) Synonym for minidisk. (T)
virtual machine (VM). In VM, a functional equivalent of a computing system. On the 370 Feature of VM, a virtual machine operates in System/370 mode. On the ESA Feature of VM, a virtual machine operates in System/370, 370-XA, or ESA/370 mode. Each virtual machine is controlled by an operating system. VM controls the concurrent execution of multiple virtual machines on an actual processor complex.

Virtual Machine/Enterprise Systems Architecture (VM/ESA). An IBM-licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a real machine.

Virtual Machine/Extended Architecture (VM/XA). An operating system that facilitates conversion to MVS/XA by allowing several operating systems (a production system and one or more test systems) to run simultaneously on a single 370-XA processor. The VM/XA Migration Aid has three components: the control program (CP), the conversational monitor system (CMS), and the dump viewing facility.

Virtual Machine/System Product (VM/SP). An IBM-licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a real machine.

virtual route (VR). In SNA, either of the following:
- A logical connection between two subarea nodes that is physically realized as a particular explicit route
- A logical connection that is contained wholly within a subarea node for intranode sessions.

A virtual route between distinct subarea nodes imposes a transmission priority on the underlying explicit route, provides flow control through virtual route pacing, and provides data integrity through sequence numbering of path information units (PIUs). See also explicit route (ER), path, and route extension (REX).

virtual route (VR) pacing. In SNA, a flow control technique used by the virtual route control component of path control at each end of a virtual route to control the rate at which path information units (PIUs) flow over the virtual route. VR pacing can be adjusted according to traffic congestion in any of the nodes along the route. See also pacing and session-level pacing.

virtual route pacing response (VRPRS). A nonsequenced, supervisory path information unit (PIU) that flows at network priority. It may overtake VR-sequenced PIUs and consists of a transmission header with no basic information unit (BIU) data.

virtual routing node. A representation of a node's connectivity to a connection network defined on a shared-access transport facility, such as a token ring. Synonymous with connection network.

virtual storage (VS). The storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of auxiliary storage available, not by the actual number of main storage locations. (I) (A)

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 Storage Extended (VSE). An IBM-licensed program whose full name is the Virtual Storage Extended/Advanced Function. It is a software operating system controlling the execution of programs.

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.

VIT. VTAM internal trace.

W

WAN. Wide area network.

wide area network (WAN). (1) A network that provides communication services to a geographic area larger than that served by a local area network or a metropolitan area network, and that may use or provide public communication facilities. (T) (2) A data communications network designed to serve an area of hundreds or thousands of miles; for example, public and private packet-switching networks, and national telephone networks. Contrast with local area network (LAN).

wildcard character. Synonym for pattern-matching character.
window. (1) A portion of a display surface in which display images pertaining to a particular application can be presented. Different applications can be displayed simultaneously in different windows. (A) (2) In data communication, the number of data packets a data terminal equipment (DTE) or data circuit-terminating equipment (DCE) can send across a logical channel before waiting for authorization to send another data packet. The window is the main mechanism of pacing, or flow control, of packets. (3) See pacing window.

write-to-operator (WTO). An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that may need to be corrected.

deadline. A time-bound limit defining when a program must return control to the operating system. The time limit is calculated together with the amount of available resources when a job is scheduled.

write-to-operator with reply (WTOR). An optional user-coded service whereby a message can be written to the system console operator informing the operator of errors and unusual system conditions that may need correcting.

WS. Work station.

WTO. Write-to-operator.

WTOR. Write-to-operator with reply.

X


X.25 NCP Packet Switching Interface (NPSI). An IBM-licensed program that allows SNA users to communicate over packet switching data networks that have interfaces complying with CCITT Recommendation X.25. It allows SNA programs to communicate with SNA or non-SNA equipment over such networks.


XA. Extended architecture.

XDL. X.25 data link control.

XGA. Extended graphics array/adapter.

XI. X.25 SNA interconnection.

XRF. Extended recovery facility.

Z

ZAP disk. The virtual disk in the VM operating system that contains the user-written modifications to VTAM code. See BASE disk, DELTA disk, MERGE disk, and RUN disk.

31-bit storage addressing. The storage address structure available in an MVS/XA operating system.
Index

Numerics
3746 APPN NN data collection, installing 68
3746 APPN NN members, generating 68

A
access to NPM, controlling 9
accounting data collection
  installing 35
  tuning 221
alerts
  installing 33, 38, 43, 45, 47
  allocating data sets 18, 21, 28
ANPM statement
  coded 52
APF authorizing NPM load libraries 12, 23, 29
APPL statement
  coded 24, 39
  example 25
applications, defining to NPM
  manually 24
  using initialization program 18
attended installation 101
authorizing
  NPM for MVS 22, 29
  NPM load libraries 22, 29
  NSI for MVS 22, 29
autologon facility for NPALUs 67

B
backing up NPM data sets 3
base interval 145
before starting NPM 175
begin function (‘FE’) 238
blanks in statements 4
books about NPM
  See information about NPM
BUFFERS statement
  coded 38, 39, 48
  parameters
    BUFFNO 212
    BUFSEGS 212
    DDRBUF 212
    FSTBUFS 213
    MAXGSD 213
    QCBCCHN 213
    SESSNO 214
    SSBHASH 213
    SSRTSIZE 214
    TPRBCCHN 213
  tuning 212
BUFFNO parameter 212
BUFSEGS parameter
  on statement
    BUFFERS 212
BVAL parameter
  on command
    LU group list 167

C
CDRM parameter
  on command
    LU group list 168
CDRM statement
  coded 52
  tuning 214
CICS
  collecting session data 72
  installing send exit 72
CLOCK parameter
  general information 142
CNMNETV, loading 44
collection commands, executing at startup 141
collection intervals
  example 147, 148
  general information 145
in LAN bridge data collection 147
in NetWare resources data collection 220
in network data collection 146
in session data collection 147
synchronizing
  example 148
  general information 147
  LAN bridge data 148
  network data 147
  session data 147
  tuning 219, 221
collections
  executing at start up 141
  for NPM initialization program 14
  LU list statement 166
reading syntax diagrams xviii
rules for coding 4
configuration
  report 181
  summary 182
  symbol resource table 184
  symbols 184
console security 8
CONSOLE statement
  coded 34
  example 34
Index

console support
  installing 34
  with MVS/ESA Version 4 34
control blocks, cleaning up 207
control functions
  access to NPM 9
CONTROL statement
  tuning 214
creating
  installation directory 94
CSA
  cleaning up control blocks 207
  cutting usage 209
  estimating storage for 39
CSRTSIZE parameter
  on statement
    BUFFERS 212

D

data collection
  automating
    general information 18
  commands 141
  installing with initialization program 141
    dynamic network 62
    LAN bridge 44
    NetWare resources 46
    network gateway accounting 35, 59, 63, 65
    network session accounting 35, 59
    RTM 71
    session by application 62
  issuing commands at start up 141
  thresholds
    tuning 218
data destinations, specifying 159
data monitoring, automating 141
data sets
  allocating
    for NPM 21, 28
  allowed number of 21, 28
  backing up before installing NPM 3
  defining with initialization program 18
  effect on performance 211
  name in function parameter list 257
  organization, optimizing 211
  protecting 9
  use of term in book xviii
DD name in function parameter list 256
defaults
  for NPM 4
DEFAULTS statement
  parameters
    LWGINT 47
    VTAMINT 217
    tuning 217
defining applications 18, 24
defining initialization parameters
  for full installation 18
  for minimum installation 18
defining NPM as VTAM application 18
defining NPM data sets
  with initialization program 18
defining operators 18, 24
DESC parameter
  in FNMMMSG macro 227
descriptor code for WTO 232
Desk/2
  See NPM Desk/2
destinations for messages, changing 225
destinations for messages, determining
  NetView 235
  operator logged on 234
  operator not logged on 235
  operator with global authority 234
  SYSPRINT data set 233
  system console 232
DESTX parameter
  on command
    LWGCOLL 235
downloading product files 93
DSICMD data set 70
DSIDMN data set 70
DSIPARM data set 70, 220, 221
dynamic reconfiguration 67

E
end function (’FF’) 239
ending NPM 206
enhanced LU groups
  description 171
  example 172
  mask 171
environment vector list in function parameter list 249
exit
  applying maintenance 136
  CICS send 72
installing
  general purpose (FNMUEXIT) 259
  PIU analysis 259
  options in function parameter list 254
  routine function codes in function parameter list 252
  routine secondary function codes 260
  SMF 123
  tuning 221
  VTAM session manager 121

F
FILE statement
  coded 24
FILE statement (continued)
  example 25
  parameters
    NAME 24, 215
    tuning 215
file switch function ('09') 243
file user data in function parameter list 254
FNMACAAX1 121
FNMACAAX2 121
FNMACAAX3 122
FNMACAAX4 122
FNCLNUP, using 207
FNMDSTD data set 70
FNMXCAAA, using 126, 130
FNMXEXEC 3
FNMXEXITs, using 127
FNMINIT 3
FNMLCOLL data set
  general information 179
  writing data to 179
FNMLGD, using 21, 28
FNMLUGRP 3
FNMLUGRP data set 166
FNMLWGx data set
  general information 179
FNMMRS macro
  example 231
  nr parameter 231
  string parameter 231
  TYPE parameter 231
FNMMMSG macro
  example 223
  parameters
    DESC 227
    MCSFLAG 228
    MSG 224
    OPT 225
    RC 228
    ROUTCDE 226
    ROUTE 225
    TEXT 225
FNMOOPER 3
FNMPROF 3
FNPSMLK, using
FNPSMDS data set 160
FNPSDUMP data set 180
FNPSMF 3
FNPSNAP data set 180
FNSTRT 3
FNSTRT data set 141
FNMTAP transit analysis program 262
FNMU83x, using 126
FNMU83X1 123
FNMU83X2 123
  FNMACAAX1 121
  FNMACAAX2 121
FNMU84, using 126
Frame relay data collection, installing 60
FSTBUFS parameter on BUFFERS statement 213
full installation, performing
  manually 27
  using initialization program 17
function codes
  begin ('FE') 238
  DNC authorization ('07') 242
  end ('FF') 239
  file switch ('09') 243
  message ('0A') 244
  NetWare record ('16') 248
  NPALU connect ('03') 241
  NPALU disconnect ('04') 241
  NPM connect ('05') 242
  NPM disconnect ('06') 242
  NPMLOG record ('10') 244
  operator connect ('01') 239
  operator disconnect ('02') 240
  performance alert ('15') 247
  review record ('13') 246
  session interval analysis ('18') 248
  session record ('12') 245
  SMF record ('14') 246
  VTAMLOG record ('11') 245
functional security 7

G
  graphic subsystem security 8
  GTF parameter
    on command
      LU group list 168
  GWSESAC, keyword on NCP BUILD statement 65

H
  hardware requirements, NPM Desk/2
    recommended 92
    required 92
  header for NPM panels
    defining with initialization program 18
  help for initialization program 14
  hexadecimal representation of special characters 5
  HOST statement
    parameter
      MAXL 215
    tuning 215
  HOSTCOLL parameter
    on statement
      NPM 215
I

IEFU83 126
IEFU84 126

INACTVR parameter
on statement
NPM 216

information about NPM xxi

initialization parameters
defining 18
retrieving 19
saving 18

initialization statements
listed in configuration report 181

rules for coding 4

initialization statements, tuning
BUFFERS 212
CDRM 214
CONTROL 214
FILE 215
HOST 215
NGRAPHS 217
NPM 216
SYS 218
TASK 218

installation methods
attended 101
manual 27
unattended 99

installation requirements
recommended hardware 92
required hardware 92
required software 93

installation-wide exit (FNMUEXIT)
installing 258
register contents 237
testing with FNMTAP 238

installation-wide exit field in function parameter list 256

installation-wide exit functions 238

installation-wide exit parameters
data set name 257
DD name 256
environment vector list 249
exit options 254
exit routine function codes 252
file user data 254
installation-wide exit field 256
message parameter list 257
operator user data 254
session user data 255
SMF data areas 258
user data 254
user security area 256
volume serial 257
WTO descriptor and routing codes 257

installing NPM
allocating NPM data sets 21, 28
APF-authorizing NPM load libraries 12, 23, 29
authorizing NPM for MVS 22, 29
authorizing NSI for MVS 12, 23, 29
changes required for other products 58
code automated commands 54
defining initialization statements 32
defining NPM for MVS 22, 29
defining NPM operators 48
defining NPM to VTAM 12, 18, 23, 30
defining the NPM start procedure 49
general FNMINIT definitions 32
install NPM exits 58
LAN data collection 44
NetWare resource data collection 46, 54
network collection 34
network gateway accounting 35
network session accounting 35
NPM Desk/2 Interface 54, 89
NPM message table 232
NPM-to-NPM communication 51
RTM collection 48
session collection 36
VTAM statistics data collection 42

installing NPM Desk/2
creating installation directory 94
defining your operating system 97
generating the NPM Desk/2 profile file 98
loading and downloading product files 93
overview 92
receiving product files 94

installing NPM message table 232

INTERVAL parameter
general information 145

on command
LWGCOLL 145

on statement
DEFAULTS
NPM 216

ISPF log 18

ISPF, using to start initialization program 14

K

keyboards, non-US 5

keywords
allowed number of 4
in BUILD statement 59
in NCP command 54
in NPM statement 34, 46
in SYS statement 32
listing
ADDRESS 60
AGENT 56
ALERT 33
APPLID 51, 52
Index

---

**keywords (continued)**

<table>
<thead>
<tr>
<th>Listing (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFNO 38, 48, 178, 212</td>
</tr>
<tr>
<td>BUFSEGS 211, 212</td>
</tr>
<tr>
<td>BVAL 152, 167</td>
</tr>
<tr>
<td>CDRM 168, 210</td>
</tr>
<tr>
<td>CLOCK 142, 152, 205</td>
</tr>
<tr>
<td>CONNUTIL 150, 154</td>
</tr>
<tr>
<td>CSRTSZ 212</td>
</tr>
<tr>
<td>DAILY 150</td>
</tr>
<tr>
<td>DDRBUF 39</td>
</tr>
<tr>
<td>DESC 227</td>
</tr>
<tr>
<td>DESTX</td>
</tr>
<tr>
<td>DYNAMIC 154, 205</td>
</tr>
<tr>
<td>FSTBUFS 213</td>
</tr>
<tr>
<td>GROUP 154, 205</td>
</tr>
<tr>
<td>GTF 168</td>
</tr>
<tr>
<td>GWSAC 59, 65</td>
</tr>
<tr>
<td>HOST 51, 52, 56, 168</td>
</tr>
<tr>
<td>HOSTCOLL 36, 209, 213, 215, 218</td>
</tr>
<tr>
<td>INACTVR 216</td>
</tr>
<tr>
<td>INTERVAL 35, 36, 42, 145, 216</td>
</tr>
<tr>
<td>ISTATUS 68</td>
</tr>
<tr>
<td>L 56</td>
</tr>
<tr>
<td>LAN 44</td>
</tr>
<tr>
<td>LANCOLL 44</td>
</tr>
<tr>
<td>LANX 45</td>
</tr>
<tr>
<td>LBRGINT 44, 145, 147, 216</td>
</tr>
<tr>
<td>LOGAPPL 67</td>
</tr>
<tr>
<td>LSEGINT 44, 145, 147, 148, 216</td>
</tr>
<tr>
<td>LU 210</td>
</tr>
<tr>
<td>LWG 46</td>
</tr>
<tr>
<td>LWGIN 47, 146, 217</td>
</tr>
<tr>
<td>LWGX 46</td>
</tr>
<tr>
<td>MACRF 132, 211, 215</td>
</tr>
<tr>
<td>MAXCOLL 61</td>
</tr>
<tr>
<td>MAXGSD 213</td>
</tr>
<tr>
<td>MAXL 41, 209, 214</td>
</tr>
<tr>
<td>MAXMON 217</td>
</tr>
<tr>
<td>MAXNCP 216</td>
</tr>
<tr>
<td>MAXSUBA 216</td>
</tr>
<tr>
<td>MAXTP 59</td>
</tr>
<tr>
<td>MAXWS 92</td>
</tr>
<tr>
<td>MCSFLAG 228</td>
</tr>
<tr>
<td>MIN 168</td>
</tr>
<tr>
<td>NAME 39, 41, 52</td>
</tr>
<tr>
<td>NETWORK 34, 56, 169</td>
</tr>
<tr>
<td>NETWORKX 34</td>
</tr>
<tr>
<td>NEWNAME 59</td>
</tr>
<tr>
<td>NODEADDR 153</td>
</tr>
<tr>
<td>NPA 59</td>
</tr>
<tr>
<td>NPACOLL 61</td>
</tr>
<tr>
<td>NPM.CYCLE 62</td>
</tr>
<tr>
<td>NPM.DNC 62</td>
</tr>
<tr>
<td>NPM.MSSH 62</td>
</tr>
<tr>
<td>NPM.SPEED 62</td>
</tr>
<tr>
<td>NSA 35</td>
</tr>
</tbody>
</table>

---

**keywords (continued)**

<table>
<thead>
<tr>
<th>Listing (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSI 38</td>
</tr>
<tr>
<td>NWCOLL 46</td>
</tr>
<tr>
<td>OPER 169</td>
</tr>
<tr>
<td>OPT 225</td>
</tr>
<tr>
<td>OPTION 57, 143, 172</td>
</tr>
<tr>
<td>PASSWD 51</td>
</tr>
<tr>
<td>PRINTMBR 152, 169, 175</td>
</tr>
<tr>
<td>PRTCT 12, 23, 30</td>
</tr>
<tr>
<td>PUNAME 59</td>
</tr>
<tr>
<td>QCB 45, 47</td>
</tr>
<tr>
<td>QCBCCH 213</td>
</tr>
<tr>
<td>RC 228</td>
</tr>
<tr>
<td>RESOURCE 56</td>
</tr>
<tr>
<td>RETRY 216</td>
</tr>
<tr>
<td>ROUTCDE 226, 257</td>
</tr>
<tr>
<td>ROUTE 225</td>
</tr>
<tr>
<td>RPROFILE 51, 53</td>
</tr>
<tr>
<td>RSP 39</td>
</tr>
<tr>
<td>RTM 48</td>
</tr>
<tr>
<td>SAMPVHI 43, 217</td>
</tr>
<tr>
<td>SAMPVLW 43, 217</td>
</tr>
<tr>
<td>SAMPVMD 43, 217</td>
</tr>
<tr>
<td>SECURITY 119, 218</td>
</tr>
<tr>
<td>SERVER 56</td>
</tr>
<tr>
<td>SESSACC 59, 63</td>
</tr>
<tr>
<td>SESSSH 41, 151, 167, 169, 210</td>
</tr>
<tr>
<td>SESSINT 24, 37, 146, 149, 216</td>
</tr>
<tr>
<td>SESSION 37</td>
</tr>
<tr>
<td>SESSIONX 37</td>
</tr>
<tr>
<td>SESSNO 38, 178, 214</td>
</tr>
<tr>
<td>SNI 54</td>
</tr>
<tr>
<td>SP 55, 152, 153</td>
</tr>
<tr>
<td>SPEED 60</td>
</tr>
<tr>
<td>SRVUTIL 154</td>
</tr>
<tr>
<td>SSBHASH 213</td>
</tr>
<tr>
<td>SSRTSZ 214</td>
</tr>
<tr>
<td>SYN 39</td>
</tr>
<tr>
<td>SYNCH 147, 216</td>
</tr>
<tr>
<td>TELNET xviii, 36, 216</td>
</tr>
<tr>
<td>TEXT 225</td>
</tr>
<tr>
<td>TNSTAT 13, 31</td>
</tr>
<tr>
<td>TPRBCCH 213</td>
</tr>
<tr>
<td>TRANSIT 39</td>
</tr>
<tr>
<td>TYPE 40</td>
</tr>
<tr>
<td>UACCTNG 61</td>
</tr>
<tr>
<td>VLOG 151, 170</td>
</tr>
<tr>
<td>VOLUTIL 154</td>
</tr>
<tr>
<td>VTAM 42</td>
</tr>
<tr>
<td>VTAMCOLL 42, 155, 157, 196, 217</td>
</tr>
<tr>
<td>VTAMINT 44, 146, 217</td>
</tr>
<tr>
<td>VTAMX 43</td>
</tr>
</tbody>
</table>
LAN bridge data collection
installing 44
LAN data collection
tuning 220
large numbers of NCPs, tuning NPM for 221
LBRGCOLL command
coded 71
LBRGINT parameter
examples 148
on NPM statement 147, 216
load libraries, APF authorizing 12, 23, 29
loading product files 93
logging on to NPM 6
LU group list statement
coded 41
parameters
  BVAL 167
  CDRM 168
  GTF 168
  HOST 168
  LU 167
  MIN 168
  NETWORK 169
  OPER 169
  PRINTMBR 169
  SESSH 169
  VLOG 170
LU group name 167
LU groups
  contents 166
  example 170
  processing 166
LU parameter
on statement
  LU list 167
LWG parameter
on statement
  NPM 46
LWGCOLL command
coded 150, 154, 164
parameters
  AGENT 154
  CLOCK 142, 154
  CONNUTIL 154
  DAILY 154
  DESTX 235
  DYNAMIC 154
  GROUP 154
  INTERVAL 145
  OPTION 143, 154
  SP 154
  SRVUTIL 154
  VOLUTIL 154
LWGINT parameter
on statement
  DEFAULTS 47
LWGRES command
coded 55, 141, 150, 153, 164
parameters
  AGENT 56
  HOST 56
  L 56
  NETWORK 55
  NODEADDR 55
  OPTION 57
  RESOURCE 56
  SERVER 56
  SP 55
LWGSTOP command 206
LWGX parameter
on statement
  NPM 46
MACRF parameter
on statement
  FILE 215
maintenance, applying to exits 136
manuals about NPM
  See information about NPM
MAXL parameter
on statement
  CDRM 214
  HOST 215
MAXMON parameter
on statement
  NGRAPHS 217
MAXNCP parameter
on statement
  NPM 216
MAXSUBA parameter
on statement
  NPM 216
MAXWS parameter
on statement
  NPM 92
MCSFLAG parameter in FNMMSG macro 228
message
  for initialization program 14
  function ('0A') 244
  inserts 229
  parameter list in function parameter list 257
  replacement string 231
  table
    reinstalling 232
    replacing messages 231
    updating 223

302 NPM Installation and Customization
Index

migrating
  to NPM V2R4 from Version 1 releases 75, 85
  to NPM V2R4 from Version 2 releases 77, 87
MIN parameter
  on command 168
minimal security 6
minimum installation, performing
  manually 21
  using initialization program 17
MSG parameter in FNMMMSG macro 224
multiple destinations for parameters, specifying 5
MVS, authorizing NPM to 12, 23, 29

N
  names, allowed 4
  NCP
    defining to NPM 58
    resources, defining to NPM 58
    RRT, using 58
    tuning NPM for large numbers 221
  NCP command
    when coded 54
  NETCOLL command
    tuning 218
  NetView Access Services
    special processing for 12, 23, 29, 38, 40, 72
  NetView program-to-program interface 70
  NetView, sending alerts to 235
  NetWare record function ('16') 248
  NetWare resources data collection
    installing 117
    stopping 206
    tuning 220
  network data collection
    installing 34
    tuning 218
  network gateway accounting
    installing 35
  NETWORK parameter
    on command 169
  network session accounting
    installing 35
  NGRAPHS statement
    MAXMON parameter 217
    tuning 217
  non-US keyboards, special characters on 5
  normal security 7
  NPA, keyword on NCP BUILD statement 67
  NPALU
    autologon facility for 67
    connect function ('03') 241
    disconnect function ('04') 241
    initial status of 67
  NPM and other products
    CICS send exit 72
    SMF exits 121
    VTAM 121
  NPM books
    See information about NPM
  NPM configuration
    See configuration
  NPM connect function ('05') 242
  NPM Desk/2
    defining LU 6.2 component name in FNMINIT 33
    defining NPM to VTAM 89
    defining resource groups 173
    example of NPM Desk/2 resource group 173, 174
    host configuration example
      defining the logmode table 89
      workstation VTAM and NCP definitions 89
      installing 89
  NPM disconnect function ('06') 242
  NPM EXECs
    processing 160
  NPM initialization program
    general information 14
    getting started 14
    main menu 14
    panel flows 15
    retrieving initialization parameters 19
    saving initialization parameters 18
    using minimum installation option 17
    using normal installation option 17
  NPM manuals
    See information about NPM
  NPM NetWare Agent code, installing 117
  NPM samples 3
  NPM statement
    coded 24
    example 25
    parameters 36
      HOSTCOLL 215
      INACTVR 216
      INTERVAL 216
      LAN 44
      LANCOLL 216
      LWG 46
      LWGX 46
      MAXNCPS 216
      MAXWS 92
      NSA 35
      NWCOLL 46
      RETRY 216
      SESSINT 24, 216
      SYNCH 36, 216
      TELNET 216
      VTAM 42
      VTAMCOLL 42
      VTAMX 43
NPM statement (continued)
  tuning 215
NPM-to-NPM communication
  example definition 53
  installing 51
NPM, defining as VTAM application 18
NPM, tuning
  adjusting accounting collection 221
  adjusting initialization statements 211
  adjusting network collection 218
  general information 209
  installation-wide exits 221
  storage requirements 209
  using performance measurement table 211
  VSAM file processing 210
NPM.CYCLE keyword
  on NCP VTAMLST 62
NPM.DNC keyword
  on NCP VTAMLST 62
NPM.SESSION keyword
  on NCP VTAMLST 62
NPM.SPEED keyword
  on NCP VTAMLST 62
NPMLOG record function ('10') 244
  nr parameter in FNMMRSL macro 231
NSA parameter
  on statement
  NPM 35
NVAS
  See NetView Access Services
NWCOLL parameter
  on statement
  NPM 46

O
OPER parameter
  on statement
    LU group list 169
operator connect function ('01') 239
operator disconnect function ('02') 240
operator profile 6
OPERATOR statement
  coded 48, 52
  example 25
operator user data in function parameter list 254
operators
  defining manually 24
  sending messages to 234, 235
OPT parameter in FNMMMSG macro 225
OPTION parameter
  on commands
    LUGROUP 172
    LWGRES 57, 143
    NETCOLL 143
    VxxxCOLL 143
order of statements 4

P
parameters
  See keywords
performance alert function ('15') 247
PF keys
  using with initialization program 14
PIU analysis exit
  functions 262
  installing 262
  parameters
    exit routine secondary function codes 260
    PIU analysis parameter list 261
    PIU data area 260
    VTAM transmission header and request header 260
    register contents 259
PIU analysis parameter list 261
PIU data area 260
PMT (performance measurement table)
  general information 39
  using to tune NPM 211
PPT, updating 12, 22, 29
PRINTMBR parameter
  on command
    LU group list 169
PROFILE statement
  coded 24
  example 25
profiles
  defining 49
  operator 6
  remote 8
  security 6
program properties table, updating 12, 22, 29
protecting NPM data sets 9
PRTCT parameter
  on VTAM APPL statement 52

Q
QCBCHN parameter
  on statement
    BUFFERS 213

R
RACF
  coded
    SECURITY parameter, tuning 218
    user security area 256
RACF security 7
RC parameter in FNMSMMSG macro 228
Index

register contents
- installation-wide exit 237
- PIU analysis exit 259
remote profiles 8
- defining 51
RESOURCE parameter
- on command
  - LWGRES 56, 154
retrieving initialization parameters 19
RETRY parameter
- on statement
  - NPM 216
review file
- allocating 18
review record function (‘13’) 246
ROUTCDE parameter
- in FNMMMSG macro 226
ROUTE parameter
- in FNMMMSG macro 225
router programs
- general information 130
- using 126
router programs, sample 124
RTM data collection
- installing 32
rules for coding statements and commands 4

S
SAMP statement 43, 217
- parameters
  - SAMPVHI 43
  - SAMPVLW 43
  - SAMPVMD 43
sample router programs 124
samples
- working with 3, 21
saving initialization parameters 18
security 6
- console 8
- functional 7
- graphic subsystem 8
- minimal 6
- normal 7
- NPM 6
- NPM-to-NPM 8
- profile 6
- RACF 7
- user-defined 7
SECURITY parameter 218
- selecting SMF exit routines 123
- selecting VTAM session manager exit 122
SESSACC keyword
- on NCP BUILD statement 63
SESSCOLL command
- coded 24, 55
SESSH parameter
- on command
  - LU group list 169
SESSINT parameter
- examples 145, 147
- general information 148
- tuning 216
session collection
- by application, installing
  - benefits of 210
  - manually 21
  - using initialization program 17
  - tuning 219
session data collection and monitoring
- inactivating to reduce CSA usage 209
session file
- allocating 18
session interval analysis function (‘18’) 248
session record function (‘12’) 245
session user data in function parameter list 255
SESSNO parameter
- on statement
  - BUFFERS 214
SMF
- SMF exit 123
- SMF exit routines, selecting 123
- working with NPM 121
SMF data areas in function parameter list 258
SMF exit, installing in MVS
- by renaming and copying 126
- using sample router program 126
- using SMP/E 126
- using your own exit routine 129
SMF parameter
- on statement
  - CONTROL 214
SMF record function (‘14’) 246
SMP/E, using to install exits
- applying maintenance 136
- restoring USERMODs 135
- using USERMODs 134
software requirements, NPM Desk/2 93
special characters on non-US keyboards 5
SSBHASH parameter
- on statement
  - BUFFERS 213
SSRTSIZE parameter
- on statement 214
START command (to start NPM) 176
starting NPM
- in MVS 175
- statements, order of 4
STOP command
- in NPM 206
stopping NetWare resources data collection 206
stopping NPM 206
storage allocation, in configuration report 182
string parameter in FNMMRS macro 231
strings, using in statements 4
symbol resource table 184
SYNCH parameter
   examples 148
   general information 147
   on statement
      NPM 216
synonym chains 184
syntax diagrams, how to use xviii
syntax errors, handling 4
SYS statement
   parameters
      SECURITY 218
tuning 218
SYSDUMP data set 180
SYSPRINT log 233
system console, using 164
system symbol resource table 184

T

TASK statement
tuning 218
TELNET parameter in NPM statement 36, 216
TEXT parameter in FNMMSSG macro 225
thresholds for data collection
tuning 218, 220
timing of data collection, specifying 142
TNSTAT parameter, in VTAM 13, 31
TPRBCHN parameter
   on statement
      BUFFERS 213
transit analysis program 262
transmission priority data collection
   installing 59
tuning NetWare resources collection
   See NetWare resources data collection, tuning
tuning NPM
   See NPM, tuning
TYPE parameter
   in FNMMRS macro 231

U

unattended installation 99
user data in function parameter list 254
user security area in function parameter list 256
user-defined security 7
USERMODs
   using to install exits 134
   using to reinstall exits 135
Utilities feature, OS/2 ES 97

V

verifying installation
   reviewing NPM configuration 181
   verifying file definitions 187
   verifying LAN bridge collection 196
   verifying LAN segment collection 200
   verifying national language features 186
   verifying NetWare resources data collection 205
   verifying network collection 191
   verifying operating environment 187
   verifying profile definitions 186
   verifying session collection 188
   verifying storage use 188
   verifying VTAM statistics collection 195
VLOG parameter
   on command
      LU group list 170
volume serial in function parameter list 257
VSAM LSR
   file processing, tuning 210
   general information 132
   installing for MVS 132
   using to improve VSAM performance 132, 211
VTAM
   application, defining NPM as
      using initialization program 18
   defining NPM to 125
   password
      defining 18
   request header 260
   session manager exit 121
   session manager exit, installing in MVS
      by renaming and copying 126, 130
      using sample router program 126, 130, 131
      using SMP/E 126
      using your own exit routine 129, 131
   session manager exit, selecting 122
   transmission header 260
VTAM statistics data collection
   installing 42
tuning collection intervals 219
tuning data collection 219
tuning monitor thresholds 219
VTAMINT parameter
   on statement
      DEFAULTS 217
VTAMLLOG record function ('11') 245
VTAMLST 62
Vxxx COLL commands 143, 145, 146
   VADRCOLL
      coding example 150, 155
tuning 219
   VAPL COLL
      coding example 150, 157
tuning 219
VxxxCOLL commands (continued)

VAPNCOLL
  coding example 150, 156
  tuning 219
VBPLCOLL
  coding example 150, 157
  tuning 219
VCSMCOLL
  coding example 150, 156
  tuning 219
VDEVCOLL
  coding example 150, 157
  tuning 219
VGBLCOLL
  coding example 150, 158
  tuning 219
VMNPCOLL
  coding example 150, 155
  tuning 219
VRTPCOLL
  coding example 150, 156
  tuning 219
VVRCOLL
  coding example 150, 158
  tuning 219

W

WTO descriptor code 232
Communicating Your Comments to IBM

Tivoli NetView Performance Monitor
Installation and Customization
Version 2  Release 4
Publication No.  SH19-6964-03

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you prefer to send comments by mail, use the reader's comment form (RCF) at the back of this book. If you wish, you can give the RCF to the local branch office or IBM representative for postage-paid mailing.

If you prefer to send comments by FAX, use this number, which is in Italy:

(39.6) 5966.2077

If you prefer to send comments electronically, use this network ID:

ROMERCF at VNET.IBM.COM

Make sure to include the following in your note:

• Title and publication number of this book
• Page number or topic to which your comment applies
Help us help you!

Tivoli NetView Performance Monitor
Installation and Customization
Version 2  Release 4
Publication No. SH19-6964-03

We hope you find this publication useful, readable and technically accurate, but only you can tell us! Your comments and suggestions will help us improve our technical publications. Please take a few minutes to let us know what you think by completing this form.

<table>
<thead>
<tr>
<th>Overall, how satisfied are you with the information in this book?</th>
<th>Satisfied</th>
<th>Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How satisfied are you that the information in this book is:</th>
<th>Satisfied</th>
<th>Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accurate</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Complete</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Easy to find</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Easy to understand</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Well organized</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Applicable to your task</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

Specific Comments or Problems:

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

Please tell us how we can improve this book:

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

Thank you for your response. When you send information to IBM, you grant IBM the right to use or distribute the information without incurring any obligation to you. You of course retain the right to use the information in any way you choose.

Name _________________________________________________________________
Address _______________________________________________________________
Company or Organization ________________________________________________
Phone No. ____________________________________________________________
Program Number: 5655-043 (MVS/ESA)

Printed in Denmark by IBM Danmark A/S