Tuning Guide

Version 5 Release 1
Tivoli NetView for z/OS Tuning Guide

Copyright Notice

© Copyright IBM Corporation 1997, 2002. All rights reserved. May only be used pursuant to a Tivoli Systems Software License Agreement, an IBM Software License Agreement, or Addendum for Tivoli Products to IBM Customer or License Agreement. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without prior written permission of IBM Corporation. IBM Corporation grants you limited permission to make hardcopy or other reproductions of any machine-readable documentation for your own use, provided that each such reproduction shall carry the IBM Corporation copyright notice. No other rights under copyright are granted without prior written permission of IBM Corporation. The document is not intended for production and is furnished “as is” without warranty of any kind. All warranties on this document are hereby disclaimed, including the warranties of merchantability and fitness for a particular purpose.

U.S. Government Users Restricted Rights—Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation.

Trademarks

IBM, the IBM logo, Tivoli, the Tivoli logo, APPN, CICS, IBMLink, MVS, OS/390, RACF, RMF, VTAM, and z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Lotus is a registered trademark of Lotus Development Corporation.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States or other countries.

Other company, product, and service names may be trademarks or service marks of others.

Notices

References in this publication to Tivoli Systems or IBM products, programs, or services do not imply that they will be available in all countries in which Tivoli Systems or IBM operates. Any reference to these products, programs, or services is not intended to imply that only Tivoli Systems or IBM products, programs, or services can be used. Subject to valid intellectual property or other legally protectable right of Tivoli Systems or IBM, any functionally equivalent product, program, or service can be used instead of the referenced product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by Tivoli Systems or IBM, are the responsibility of the user. Tivoli Systems or 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 the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, New York 10504-1785, U.S.A.

Programming Interfaces

This publication documents no intended Programming Interfaces that allow the customer to write programs to obtain services of Tivoli Netview for z/OS.
Contents

Figures ........................................... v

Tables ............................................. vii

Preface ........................................... ix
Who Should Read This Document ............ ix
What This Document Contains ............... ix
Publications ..................................... ix
   Prerequisite and Related Documents ...... x
   Accessing Publications Online .......... x
   Ordering Publications ................. x
   Providing Feedback about Publications x
Contacting Customer Support ............... xi
Accessibility Information ..................... xi
   Keyboard Access ......................... xi
Conventions Used in This Document ....... x i
   Platform-specific Information ...... x i
   Terminology ......................... x ii
Reading Syntax Diagrams ...................... x iii
   Required Syntax ........................ x iii
   Optional Keywords and Variables ...... x iii
   Default Values ........................... x iv
   Long Syntax Diagrams .................. x iv
   Syntax Fragments ....................... x iv
   Commas and Parentheses .............. x v
   Highlighting, Brackets, and Braces .. x vi
   Abbreviations ......................... x vii

Chapter 1. Improving NetView Performance ........................................... 1
Achieving Performance Goals .................. 1
Using General Techniques ..................... 2

Chapter 2. Tuning for Automated Operations ........................................... 5
Tuning Techniques ............................... 5
Limiting System Messages ..................... 6
Using MVS Consoles ............................. 6
   Using Subsystem Allocatable Consoles .. 6
   Using EMCS Console Support ............ 7
NetView Subsystem Address Space .......... 7
NetView Automation Table ..................... 9
   Using BEGIN/END to Improve Efficiency 10
   Using Other Techniques to Improve Efficiency 10
   The AUTOCNT Command ................ 11
Automating Hardware Monitor Records ....... 16
Filtering Hardware Monitor Records ......... 16
Automation Tasks (Autotasks) ................ 17
Using Multiple Autotasks .................... 17
Using Full-Screen Automation .............. 17
Command and Message Forwarding ........... 19
Separation of the Automation Workload from Other NetView Workloads ......... 19
   Multiple NetView Programs ............ 19

Chapter 3. Tuning for AON ........................................... 23
Tuning Techniques ......................... 23
   The DSICMD Resident Option .......... 23
   DSICLD Library ......................... 23
   Automation Table ....................... 23
   DDF Tree and Panel ..................... 24
   Node Automation ....................... 24
   Operations ......................... 24
   TCP/IP Support for AON .......... 25

Chapter 4. Tuning for Command Procedures ........................................... 27
Tuning Techniques ......................... 27
   Command Lists ......................... 28
      Preloading Command Lists .......... 28
      Managing Command Lists with AUTODROP 29
Subroutines ......................... 30
REXX Command Lists ....................... 30
   How to Compile REXX Procedures ..... 31
   Compiled REXX/370 Command Lists .... 31
   REXX Function Packages .......... 32
   Estimating REXX Environments ....... 36
   Tuning REXX Environments .......... 37
Command Processors ....................... 39
   Command Processors Written in a High-Level Language .......... 39
   Running PL/I Programs in a Preinitialized Environment .......... 40
Global Variables ....................... 42
   Enhancing Performance .......... 42
   Save/Restore Processing .......... 42

Chapter 5. Tuning for the Hardware Monitor ........................................... 45
Tuning Techniques ......................... 45
   Hardware Monitor Filters .......... 45
   Alerts-Dynamic Panel ................. 47
   Using the ALCACHE Statement in CNMSTYLE .... 48
   Using the ALERTLOG Statement in CNMSTYLE .... 49
   Using the NPDA.ALERTFWD Statement .... 49
HMSTATS Command ....................... 50
   Using the DSRBO Parameter in CNMSTYLE .... 54
Wrap Counts ......................... 54
   SWRAP Command .................... 54
   Initialization Specifications ....... 54
   Error-to-Traffic (E/T) Ratio Thresholds ........... 55
   SRFILTER and SRATIO Commands .... 55
   RATIO Statement Initialization Specifications .... 56
   RATE Statement Initialization Specifications .... 56
Event/Automation Service .......... 57

Single NetView Program Using WLM Enclaves ........................................... 20

Chapter 3. Tuning for AON ........................................... 23
Tuning Techniques ......................... 23
   The DSICMD Resident Option .......... 23
   DSICLD Library ......................... 23
   Automation Table ....................... 23
   DDF Tree and Panel ..................... 24
   Node Automation ....................... 24
   Operations ......................... 24
   TCP/IP Support for AON .......... 25

Chapter 4. Tuning for Command Procedures ........................................... 27
Tuning Techniques ......................... 27
   Command Lists ......................... 28
      Preloading Command Lists .......... 28
      Managing Command Lists with AUTODROP 29
Subroutines ......................... 30
REXX Command Lists ....................... 30
   How to Compile REXX Procedures ..... 31
   Compiled REXX/370 Command Lists .... 31
   REXX Function Packages .......... 32
   Estimating REXX Environments ....... 36
   Tuning REXX Environments .......... 37
Command Processors ....................... 39
   Command Processors Written in a High-Level Language .......... 39
   Running PL/I Programs in a Preinitialized Environment .......... 40
Global Variables ....................... 42
   Enhancing Performance .......... 42
   Save/Restore Processing .......... 42

Chapter 5. Tuning for the Hardware Monitor ........................................... 45
Tuning Techniques ......................... 45
   Hardware Monitor Filters .......... 45
   Alerts-Dynamic Panel ................. 47
   Using the ALCACHE Statement in CNMSTYLE .... 48
   Using the ALERTLOG Statement in CNMSTYLE .... 49
   Using the NPDA.ALERTFWD Statement .... 49
HMSTATS Command ....................... 50
   Using the DSRBO Parameter in CNMSTYLE .... 54
Wrap Counts ......................... 54
   SWRAP Command .................... 54
   Initialization Specifications ....... 54
   Error-to-Traffic (E/T) Ratio Thresholds ........... 55
   SRFILTER and SRATIO Commands .... 55
   RATIO Statement Initialization Specifications .... 56
   RATE Statement Initialization Specifications .... 56
Event/Automation Service .......... 57
# Figures

1. Required Syntax Elements  
2. Syntax for Variables  
3. Optional Syntax Elements  
4. Sample of Defaults Syntax  
5. Sample Syntax Diagram with Fragments  
6. Sample Syntax Diagram with Commas  
7. Example: Using a REXX CLIST to Collect Performance Statistics  
8. Message Detail Report  
9. Message Summary Report  
10. CNMSJM11 before Modification to Switch Function Package Search Order  
11. CNMSJM11 After Modification with No User Function Package Defined  
12. CNMSJM11 After Modification with a User Function Package Defined  
13. Sample Output of the HLLENV LIST,STATS,RESET Command  
14. Hardware Monitor Database and Filters  
15. Example of Output from HMSTATS Command  
16. Example of KCLASS and MAPSESS Statements  
17. Example of KCLASS and MAPSESS Statements using the KEEPSESS and DGROUP options  
18. Session Monitor Session and Storage Information Panel  
19. RODM Log Record Type 8 for API Statistics  
20. RODM Log Record Type 8 for Segment and Window Statistics  
21. Histogram Data  
22. Sample BLDVRP Macros Defining VSAM Buffer Pools for CNMSJM01  
23. Sample LISTCAT Command Output Using 3390 DASD  
24. Sample Output from the VSAMPOOL Command Using 3390 DASD  
25. Sample Output from the DSRBS Command  
26. Sample DISPPI Command Output  
27. Sample Output from the RESOURCE Command  
28. Sample TASKMON Output  
29. TASKUTIL Command Output  
30. Output from TASKUTIL Command, 2 Hours Later  

v
## Tables

1. Resource Web sites ........................................ x
2. How the Position of Syntax Diagram Elements Is Used .......... xiii
3. Syntax Elements Examples .................................... xvi
4. NetView Subsystem Address Space Job Values .................. 7
5. Determining the Best ALCACHE Statement for Your System .... 48
6. NMC topology server/NMC topology console platforms .......... 81
7. Sample MACRF and CISIZE Values for NetView Components .. 98
8. Base NetView Formulas ....................................... 140
9. Hardware Monitor Formulas ................................... 141
10. Session Monitor Formulas .................................... 144
11. VSAM LSR Buffer Storage ................................... 145
12. NetView Subsystem Storage .................................. 145
13. Additional NetView Storage .................................. 148
14. GMFHS Address Space ........................................ 149
15. Event/Automation Service Address Space ....................... 149
16. RODM Address Space Storage ................................ 150
17. RODM Object Storage ....................................... 151
18. RODM Data and Address Space Storage for MSM-managed Resources ........................................... 153
19. Minimum Virtual Storage Requirements Summary ............... 154
20. DASD Storage Conversion Table .............................. 154
21. Minimum DASD Storage Requirements Summary ................. 155
Preface

This document provides tuning information and techniques for the Tivoli® NetView® for z/OS™ product. For the purpose of this document, tuning is any activity that helps the NetView program and its network environment achieve a certain performance goal in areas such as response time, resource utilization, and throughput.

Although this document does not provide a precise method of tuning the NetView program, it highlights those areas that have the most effect on the performance of the NetView program and explains what actions you can take in those areas to improve the performance. This document assumes that you have some problem diagnosis experience and have an idea of what is causing the performance goal to be missed.

Who Should Read This Document

This document is intended for system programmers and those persons whose jobs involve improving the performance of the NetView program. It assumes a thorough understanding of the NetView program.

What This Document Contains

Each chapter of this document describes a different NetView component or area of tuning concern. Start with the chapter that describes the component that you think is affecting performance the most. A list summarizing major tuning techniques for that component or area appears at the beginning of each chapter. If you are just getting started in tuning the NetView program, concentrate on these lists. The most important tuning recommendations are listed first. Only the most important tuning recommendations in this document are covered in the lists. Additional tuning recommendations are described in the chapters as follows.

Chapter 1, “Improving NetView Performance” on page 1 provides basic tuning information for storage and network growth, and should be read by those persons just beginning to tune their NetView programs.

Chapter 2, “Tuning for Automated Operations” on page 5 through Chapter 9, “Tuning for VSAM” on page 97 refer to specific NetView components and their tuning concerns.

Chapter 10, “Additional Tuning Considerations” on page 107 provides miscellaneous tuning information and operating system-specific tuning techniques.

Chapter 11, “Storage Considerations” on page 135 provides the formulas to estimate storage requirements and other storage considerations.

Publications

This section lists prerequisite and related documents. It also describes how to access Tivoli publications online, how to order Tivoli publications, and how to make comments on Tivoli publications.
Prerequisite and Related Documents

To read about the new functions offered in this release, refer to the Tivoli NetView for z/OS Installation: Migration Guide.

You can find additional product information on these Internet sites:

<table>
<thead>
<tr>
<th>Table 1. Resource Web sites</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM®</td>
</tr>
<tr>
<td>Tivoli Systems</td>
</tr>
<tr>
<td>Tivoli NetView for z/OS</td>
</tr>
</tbody>
</table>

The Tivoli NetView for z/OS Web site offers demonstrations of the NetView product, related products, and several free NetView applications you can download. These applications can help you with tasks such as:

- Getting statistics for your automation table and merging the statistics with a listing of the automation table
- Displaying the status of a JES job or cancelling a specified JES job
- Sending alerts to the NetView program using the program-to-program interface (PPI)
- Sending and receiving MVS™ commands using the PPI
- Sending TSO commands and receiving responses

Accessing Publications Online

You can access many Tivoli publications online using the Tivoli Information Center, which is available on the Tivoli Customer Support Web site:

http://www.tivoli.com/support/documents/

These publications are available in PDF format. Translated documents are also available for some products.

Ordering Publications

You can order many Tivoli publications online at the following Web site:

http://www.ibm.com/shop/publications/order

You can also order by telephone by calling one of these numbers:

- In the United States: 800-879-2755
- In Canada: 800-426-4968
- In other countries, for a list of telephone numbers, see the following Web site:
  http://www.tivoli.com/inside/store/lit_order.html

Providing Feedback about Publications

We are very interested in hearing about your experience with Tivoli products and documentation, and we welcome your suggestions for improvements. If you have comments or suggestions about our products and documentation, contact us in one of the following ways:

- Send an e-mail to pubs@tivoli.com.
- Complete our customer feedback survey at the following Web site:
Contacting Customer Support

If you have a problem with any Tivoli product, you can contact Tivoli Customer Support. See the Tivoli Customer Support Handbook at the following Web site:

http://www.tivoli.com/support/handbook/

The handbook provides information about how to contact Tivoli Customer Support, depending on the severity of your problem, and the following information:

- Registration and eligibility
- Telephone numbers and e-mail addresses, depending on the country you are in
- What information you should gather before contacting support

Note: Additional support for Tivoli NetView for z/OS is available at the NetView for z/OS Web site:

http://www.tivoli.com/nv390

Under Related Documents, select Other Online Sources.

The page displayed contains a list of news groups, forums, and bulletin boards.

Accessibility Information

Refer to Tivoli NetView for z/OS User's Guide for information about accessibility.

Keyboard Access

Standard shortcut and accelerator keys are used by the product and are documented by the operating system. Refer to the documentation provided by your operating system for more information.

Refer to Tivoli NetView for z/OS User's Guide for more information about keyboard access.

Conventions Used in This Document

The document uses several typeface conventions for special terms and actions. These conventions have the following meaning:

**Bold** Commands, keywords, flags, and other information that you must use literally appear like this, in bold.

*Italics* Variables and new terms appear like this, in italics. Words and phrases that are emphasized also appear like this, in italics.

**Monospace** Code examples, output, and system messages appear like this, in a monospace font.

**ALL CAPS** Tivoli NetView for z/OS commands are in ALL CAPITAL letters.
Platform-specific Information

For more information about the hardware and software requirements for NetView components, refer to the Tivoli NetView for z/OS Licensed Program Specification.

Terminology

For a list of Tivoli NetView for z/OS terms and definitions, refer to:

http://www.networking.ibm.com/nsg/nsgmain.htm

For brevity and readability, the following terms are used in this document:

NetView

- Tivoli NetView for z/OS Version 5 Release 1
- Tivoli NetView for OS/390® Version 1 Release 4
- Tivoli NetView for OS/390 Version 1 Release 3
- TME 10™ NetView for OS/390 Version 1 Release 2
- TME 10 NetView for OS/390 Version 1 Release 1
- IBM NetView for MVS Version 3
- IBM NetView for MVS Version 2 Release 4
- IBM NetView Version 2 Release 3

MVS

OS/390, or z/OS operating systems.

RACF®

RACF is a component of the SecureWay® Security Server for z/OS and OS/390, providing the functions of authentication and access control for OS/390 and z/OS resources and data, including the ability to control access to DB2® objects using RACF profiles. Refer to:


Tivoli Enterprise™ software

Tivoli software that manages large business networks.

Tivoli environment

The Tivoli applications, based upon the Tivoli Management Framework, that are installed at a specific customer location and that address network computing management issues across many platforms. In a Tivoli environment, a system administrator can distribute software, manage user configurations, change access privileges, automate operations, monitor resources, and schedule jobs. You may have used TME 10 environment in the past.

TME 10

In most product names, TME 10 has been changed to Tivoli.

V and R

Specifies the version and release.

VTAM® and TCP/IP

VTAM and TCP/IP are included in the IBM Communications Server element of the OS/390 and z/OS operating systems. Refer to

http://www.ibm.com/software/network/commserver/about/
Unless otherwise indicated, references to programs indicate the latest version and release of the programs. If only a version is indicated, the reference is to all releases within that version.

When a reference is made about using a personal computer or workstation, any programmable workstation can be used.

**Reading Syntax Diagrams**

Syntax diagrams start with double arrowheads on the left (▶️) and move along the main line until they end with two arrowheads facing each other (◀️).

As shown in the following table, syntax diagrams use position to indicate the required, optional, and default values for keywords, variables, and operands.

*Table 2. How the Position of Syntax Diagram Elements Is Used*

<table>
<thead>
<tr>
<th>Element Position</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the command line</td>
<td>Required</td>
</tr>
<tr>
<td>Above the command line</td>
<td>Default</td>
</tr>
<tr>
<td>Below the command line</td>
<td>Optional</td>
</tr>
</tbody>
</table>

**Required Syntax**

The command name, required keywords, variables, and operands are always on the main syntax line. Figure 1 specifies that the `resname` variable must be used for the `CCPLOADF` command.

```
Figure 1. Required Syntax Elements

CCPLOADF resname
```

Keywords and operands are written in uppercase letters. Lowercase letters indicate variables such as values or names that you supply. In Figure 2, `MEMBER` is an operand and `membername` is a variable that defines the name of the data set member for that operand.

```
TRANSMSG MEMBER=membername
```

**Optional Keywords and Variables**

Optional keywords, variables, and operands are below the main syntax line. Figure 3 on page xiv specifies that the ID operand can be used for the `DISPREG` command, but is not required.
Default Values

Default values are above the main syntax line. If the default is a keyword, it appears only above the main line. You can specify this keyword or allow it to default.

If an operand has a default value, the operand appears both above and below the main line. A value below the main line indicates that if you choose to specify the operand, you must also specify either the default value or another value shown. If you do not specify an operand, the default value above the main line is used.

Figure 4 shows the default keyword STEP above the main line and the rest of the optional keywords below the main line. It also shows the default values for operands MODNAME=* and OPTION=* above and below the main line.

RID

Figure 4. Sample of Defaults Syntax

Long Syntax Diagrams

When more than one line is needed for a syntax diagram, the continued lines end with a single arrowhead (►). The following lines begin with a single arrowhead (►), as shown in Figure 4.

Syntax Fragments

Commands that contain lengthy groups or a section that is used more than once in a command are shown as separate fragments following the main diagram. The fragment name is shown in mixed case. See Figure 5 on page xv for a syntax with the fragments ReMote and FromTo.
Commas and Parentheses

Required commas and parentheses are included in the syntax diagram. When an operand has more than one value, the values are typically enclosed in parentheses and separated by commas. In the OP operand, for example, contains commas to indicate that you can specify multiple values for the testop variable.

Figure 5. Sample Syntax Diagram with Fragments
If a command requires positional commas to separate keywords and variables, the commas are shown before the keyword or variable, as in Figure 4 on page xiv.

For example, to specify the BOSESS command with the `sessid` variable, enter:

```
NCCF BOSESS applid,,sessid
```

You do not need to specify the trailing positional commas. Positional and non-positional trailing commas either are ignored or cause the command to be rejected. Restrictions for each command state whether trailing commas cause the command to be rejected.

**Highlighting, Brackets, and Braces**

Syntax diagrams do not rely on highlighting, underscoring, brackets, or braces; variables are shown italicized in hardcopy or in a differentiating color for NetView help and BookManager® online books.

In parameter descriptions, the appearance of syntax elements in a diagram immediately tells you the type of element. See Table 3 for the appearance of syntax elements.

**Table 3. Syntax Elements Examples**

<table>
<thead>
<tr>
<th>This element...</th>
<th>Looks like this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyword</td>
<td>CCPLOADF</td>
</tr>
<tr>
<td>Variable</td>
<td>resname</td>
</tr>
<tr>
<td>Operand</td>
<td>MEMBER=membername</td>
</tr>
<tr>
<td>Default</td>
<td>today or INCL</td>
</tr>
</tbody>
</table>
Abbreviations

Command and keyword abbreviations are described in synonym tables after each command description.
Chapter 1. Improving NetView Performance

This chapter addresses many tuning-related questions asked by users of the NetView program. This chapter also provides basic tuning recommendations to improve general NetView performance. Review these topics and recommendations for relevancy to your environment.

Achieving Performance Goals

Tuning is any activity that helps the NetView program and its network environment achieve a certain performance goal in areas such as response time, resource utilization, and throughput. This goal or expectation can be as formal as a service level agreement or as informal as a person’s perception. Regardless of the formality of the performance goal, tuning is any activity performed to meet that goal.

The task of tuning is difficult to define. People have many different perceptions and approaches to tuning. Unlike the installation task, no set sequence of steps leads to a tuned system. This book describes the areas that affect the performance of the NetView program and shows methods for controlling and optimizing that performance. It explains how the following factors affect NetView program tuning:

- Filter use
- Automation activity
- Network size
- VSAM data set design, placement, and management
- Command list use, design, and data set placement
- View size
- Workstation CPU and RAM
- Link speed

An important part of tuning is setting the performance goal. However, that goal varies from installation to installation. Some installations have more message traffic or network activity than others. Many factors uniquely define the performance goal for each installation; therefore, these factors are not discussed in this book.

This book explains how to determine certain tuning values and suggests values you should use. Each environment needs to be tuned according to the following factors:

- Automation levels
- Logon and logoff rates
- Monitoring activities
- Network component failure rates
- Network transaction rates
- Problem determination activities
- System transaction rates
- Resource status change activities

All recommendations are based on experiences and measurements done in an IBM controlled environment. The use of these recommendations is a customer responsibility and depends on the customers’ ability to evaluate and integrate them into their operational environments.
Using General Techniques

If you are just getting started in tuning the NetView program, use the following general tuning techniques to improve overall NetView performance. You can handle the most frequently encountered performance and tuning considerations by using these techniques.

1. Use the RATE statement to stop rapidly recurring hardware monitor alerts from being recorded to the hardware monitor database. See “RATE Statement Initialization Specifications” on page 56 for more information.

2. Decide which data is critical to keep for the session monitor, and set the following session monitor options and parameters accordingly:
   - Sense code values in DSICTMOD for DASD filtering
   - KCLASS DASD option
   - KCLASS AVAIL option
   - KCLASS SAW option
   - KCLASS KEEPPIU option
   - KCLASS KEEPSESS option

   See Chapter 6, “Tuning for the Session Monitor” on page 59 for more information.

3. Use the DBAUTO command to reorganize or reset the VSAM databases regularly. See “VSAM Database Maintenance” on page 105 for more information.

4. Use the following utilities to monitor performance:

   **AUTOCNT**
   Generates automation table usage reports. See “The AUTOCNT Command” on page 11.

   **DSRBS**
   Displays statistics on data services request block (DSRB) use for NetView and user-written data services tasks (DSTs). See “Using the DSRBO Parameter in CNMSTYLE” on page 54.

   **DISPPI**
   Displays information about NetView program-to-program interface buffer queues, including buffer queue lengths, total buffers sent, and buffer storage usage. See “NetView Program-to-Program Interface” on page 116.

   **HLLENV**
   Displays information about usage of the regular and critical preinitialized environment pools by PL/I command processors. See “Running PL/I Programs in a Preinitialized Environment” on page 40.

   **HMSTATS**
   Displays hardware monitor event and alert workload counters, and statistics about the alert cache and alerts dynamic (ALD) screen update processing. See “HMSTATS Command” on page 50.

   **LISTCAT**
   Displays VSAM database definition and performance data for NetView DSTs that have open VSAM databases. See “LISTCAT Command” on page 101.

   **LOGTSTAT**
   Logs record type 38 subtype 2 to SMF at logoff/logon/termination.
MAPCL
Provides information about preloaded CLISTs. See “Preloading Command Lists” on page 28

QRYGLOBL
Displays information about NetView global variables, including the expected number of common or task global variables and the actual number of variables found. See “Global Variables” on page 42

RESOURCE
Provides CPU and storage statistics for the NetView address space. See “RESOURCE Command” on page 119

SESSMDIS
Displays session monitor session counts, storage use, and workload traffic information. See “SESSMDIS Command” on page 75

STATAPI
Generates RODM API statistics to analyze the content and activity of RODM. See “RODM API Statistics” on page 89

STATCELL
Collects information concerning the distribution of storage cells in windows and segments. See “RODM Cell Pool Statistics” on page 91

TASKMON
Shows CPU, storage, I/O, and message queue rate statistics for all tasks in NetView.

TASKURPT
A NetView sample that demonstrates how reports can be generated from the task resources data in the SMF log via REXX.

TASKUTIL
Displays task performance information, including CPU utilization, queue lengths, storage use, and active command lists. See “TASKUTIL Command” on page 127

TOPOSNA LISTPOOL
Displays SNA Topology manager storage pool statistics.

TOPOSNA LISTRODM
Displays RODM activity and object counts. See “SNA Topology Manager” on page 123

TOPOSNA LISTSTOR
Displays storage usage counts for SNA topology manager. See “SNA Topology Manager” on page 123

VSAMPOOL
Displays VSAM local shared resource (LSR) pool use statistics. See “VSAMPOOL Command” on page 103

You can add these utilities to a command list that can be executed periodically to collect performance statistics. See the example in Figure 7 on page 4
Recording performance statistics to the NetView log in this manner can provide useful historical information for trend analysis and can be invaluable in performance problem analysis.

```rexx
/* REXX clist to collect performance statistics. Can be invoked by an*/
/* EVERY timer to run under an autotask - for example, */
/* EXCMD AUTO2,EVERY 00:15,V5R1PERF */
/* Substitute your RODMNAME for X below! If you're not using RODM, */
/* delete the lines referring to X below. */
/* */
/* */
'MVS F X,STATAPI,CLEAR'
'AUTOCNT REPORT=BOTH,STATS=SUMMARY'
'DISPII'
'DSRBS AAUTSKLP'
'DSRBS BNJDSERV'
'DSRBS DSILOG'
'HMSTATS'
'LISTCAT AAUTSKLP'
'LISTCAT BNJDSERV'
'LISTCAT DSILOG'
'MAPCL'
'MVS F X,STATCELL'
'RESOURCE'
'SESSMDIS'
'TASKMON = *
/* 'TASKURPT (NOWINDOW' */
'TASKUTIL'
'TOPOSNA LISTPOOL'
'TOPOSNA LISTSTOR'
'TOPOSNA LISTSTOR'
'VSAMPOOL'
'say 'End of V5R1PERF CLIST'
exit
```

Figure 7. Example: Using a REXX CLIST to Collect Performance Statistics

Recording performance statistics to the NetView log in this manner can provide useful historical information for trend analysis and can be invaluable in performance problem analysis.
Chapter 2. Tuning for Automated Operations

The NetView program provides functions that you can use to automate operations. Many messages are sent to the NetView program for automation after being processed by the MVS message processing facility (MPF).

For a detailed description of NetView automation, refer to the Tivoli NetView for z/OS Automation Guide.

Tuning Techniques

Following are the major tuning techniques for automated operations, arranged in order of expected effect on performance, with the most important tuning considerations listed first. These recommendations are described in detail in this chapter.

1. Limit the number of system messages processed by the NetView program. On MVS systems, include a .NO_ENTRY statement specifying AUTO(NO) in the MPFLST.xx member of SYS1.PARMLIB. See “Limiting System Messages” on page 6.
2. Segment the automation table using BEGIN/END sections to minimize the average number of automation table statements evaluated for each message or MSU. See “Using BEGIN/END to Improve Efficiency” on page 10.
3. Order BEGIN/END sections and statements within sections in order of frequency to minimize the average number of statements evaluated for each message or MSU. See “Using BEGIN/END to Improve Efficiency” on page 10.
4. Use the AUTOCNT command to generate usage reports for the NetView automation table. See “The AUTOCNT Command” on page 11.
5. Eliminate simple command procedures when the condition processing can be performed directly in the automation table. See “Using Other Techniques to Improve Efficiency” on page 10.
6. Block the hardware monitor ESREC, AREC, and OPER filters where possible for alerts that are to be automated. See “Filtering Hardware Monitor Records” on page 16.
7. Minimize use of the CONTINUE automation table action if it causes the entire table to be scanned. See “Using Other Techniques to Improve Efficiency” on page 10.
8. Use the MSUSEG automation table condition if you plan to automate alerts. This automates the MSU directly, rather than generating the BNJ146I and BNJ030I messages using the hardware monitor OPER filter. See “Automating Hardware Monitor Records” on page 16.
9. Use multiple autotasks for MVS environments to distribute the automation workload across multiple processors. See “Automation Tasks (Autotasks)” on page 17.
10. Use the RMTCMD command, where possible, to forward commands and messages between a distributed system and a focal point. The alternative method uses OST-NNT sessions. See “Command and Message Forwarding” on page 19.
Limiting System Messages

To use NetView automation support with the NetView program, the first and most important step is to limit the number of system messages the NetView program processes. The NetView automation table is searched for most messages that pass through the NetView program. Therefore, it is important to limit the system message traffic sent to the NetView program to only those messages that should be considered for automation or those that are needed for display purposes.

On MVS systems, the MPFLSTxx member of SYS1.PARMLIB enables you to identify whether a message is eligible for automation processing. Eligible messages are passed to the NetView program over the subsystem interface or through extended multiple console support (EMCS) consoles (see “NetView Subsystem Address Space” on page 7).

When passing messages to the NetView program, note the following about MPFLSTxx:

- Specify AUTO(YES) or AUTO(NO) to identify whether a message is eligible for automation processing.
- Specify the defaults for groups of messages listed in the MPFLSTxx member with the .DEFAULT statement. The .DEFAULT statement uses AUTO(NO) unless you specify AUTO(YES).
- Specify the default processing you want for messages that are not identified in the MPFLSTxx member with the .NO_ENTRY statement.

**Note:** Ensure that your MPFLSTxx member has a .NO_ENTRY statement with AUTO(NO) specified. This helps limit the number of messages sent to the NetView program to only those needed for automation or display purposes.

The .NO_ENTRY statement uses AUTO(YES) unless you specify AUTO(NO). If you do not have a .NO_ENTRY statement in MPFLSTxx, the system uses a default of AUTO(YES).

For more information about MPF, refer to the appropriate MVS publication.

When you automate the operation of distributed hosts with a focal point host, handle as much of the traffic as possible at the distributed host to minimize the overhead of forwarding messages to the focal point host for automation.

Using MVS Consoles

The NetView program uses MVS consoles for receiving system messages and command responses from MVS. The NetView program can use one of the following types of MVS consoles:

- Subsystem allocatable consoles
- EMCS consoles

Define the type of MVS console used by the NetView program by specifying the system message delivery mechanism on the MVSPARM statement in CNMSTYLE. Refer to the Tivoli NetView for z/OS Administration Reference for more information.

Using Subsystem Allocatable Consoles

Subsystem allocatable consoles are virtual consoles reserved for use by subsystems such as JES2, JES3, and the NetView program. These consoles are defined in the
CONSOLxx member of SYS1.PARMLIB. They do not exist physically but are able to interact with subsystems as if they were actual consoles.

For information about defining subsystem allocatable consoles, see the appropriate MVS publication. To use the subsystem interface method of message routing, specify MVSPARM.MSGIFAC = USESSI in CNMSTYLE.

**Using EMCS Console Support**

The NetView program provides an option to use EMCS consoles. EMCS consoles use the task with the load module name CNMCSSIR to receive all messages marked AUTO(YES) or AUTO(token) in the MPF table.

You can assign attributes to the consoles and change the attributes of your EMCS consoles to ensure that messages with certain route codes are delivered to the consoles you specify. See the *Tivoli NetView for z/OS Security Reference* for a description of the EMCS console attributes.

To use EMCS consoles, specify MVSPARM.MSGIFAC=SYSTEM or MVSPARM.MSGIFAC=CMDONLY in CNMSTYLE and the subsystem interface procedure parameters.

**Note:** It is recommended that the CNMCSSIR task be the only task that obtains an AUTO(YES) console so that all DOM traffic can be processed properly.

**NetView Subsystem Address Space**

The NetView subsystem address space job must be running to enable you to:

- Receive NetView commands from sources outside of the NetView program
- Start the program-to-program interface
- Receive messages that are to be delivered to the NetView program through the subsystem

The sizes of the message and command buffers are specified on the EXEC statement of the NetView subsystem address space job. See the *Tivoli NetView for z/OS Installation: Getting Started* for a description of the subsystem startup job (CNMPSSI).

*Table 4* lists the minimum, maximum, and recommended values for use in the NetView subsystem address space job.

<table>
<thead>
<tr>
<th></th>
<th>REG</th>
<th>MBUF</th>
<th>CBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>When receiving messages through the subsystem interface:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum</td>
<td>250</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Maximum</td>
<td>15825</td>
<td>8000000</td>
<td>8000000</td>
</tr>
<tr>
<td>Recommendation</td>
<td>1250</td>
<td>4000</td>
<td>200</td>
</tr>
<tr>
<td><strong>When receiving messages using EMCS consoles:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum</td>
<td>250</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Maximum</td>
<td>15825</td>
<td>8000000</td>
<td>8000000</td>
</tr>
<tr>
<td>Recommendation</td>
<td>275</td>
<td>100</td>
<td>200</td>
</tr>
</tbody>
</table>

*where:*
REG  Specifies the region size. To calculate region size, multiply the sum of the number of message and command buffers by 256, divide that product by 1024, and add a program size of 200K. The region size (REG) is calculated using the following equation:

\[
\text{REG} = \frac{(\text{MBUF} + \text{CBUF}) \times 256}{1024} + 200K
\]

**Note:** If you are using the NetView program-to-program interface (PPI), see [“Estimating Storage Usage” on page 135](#) for estimating the virtual storage requirements of the NetView subsystem address space.

MBUF  Specifies the number of message buffers to be allocated in the NetView subsystem address space. The message buffer queue holds the WTO, WTOR, and DOM messages for processing by the NetView program. The size of each message buffer is 256 bytes.

**Note:** When you use EMCS consoles for message delivery, set the message buffer size to the minimum value, because no messages appear on the subsystem interface.

CBUF  Specifies the number of command buffers to be allocated in the NetView subsystem address space. The command buffer queue holds the NetView commands entered from the MVS console for processing by the NetView program.

MSGIFAC  Specifies when message buffering begins. When USESSI is specified, messages are buffered as soon as the Subsystem Interface Router task becomes active in the NetView application address space. This is the default. If QUESSI, QSSIAT, or SSIEET is specified, buffering occurs as soon as the SSI initializes. The Subsystem Interface Router task does not need to be active.

**Notes:**

1. Using MSGIFAC=QUESSI, QSSIAT, or SSIEET may require that the REG and MBUF keyword values be changed in the NetView SSI startup procedure. Buffered messages begin to accumulate as soon as the NetView SSI completes its initialization. Until the NetView Subsystem Interface Router (task with MOD=CNMCSSIR in the NetView Application address space) becomes active, the buffered messages will continue to accumulate. The rate at which the subsystem address space fills depends on the amount of message traffic that is generated. Therefore, the time it takes to fill the subsystem address space depends on message traffic and the number of message buffers defined (MBUF) when the subsystem was started. If MBUF is increased, the subsystem region size (REG) may also need to be increased.

Because the subsystem address space has the ability to queue messages while the router is inactive, there will be a backlog of messages to be processed. Depending on the number of messages that were queued, the router task may show increased activity until the backlog is relieved. Therefore, it is more important to have the MPF table analyzed thoroughly so that messages are not passed unnecessarily to the subsystem address space.
2. Regardless of which MSGIFAC value you use, consider using the MAXMQOUT setting to limit the rate that the CNMCSSIR task can send data. Use the TASKMON command, or review SMF record type 38, subtype 2, to determine the historical rates that CNMCSSIR produced. Then set the MAXMQOUT setting using the OVERRIDE TASK=CNMCSSIR,MAXMQOUT=nnnn command. This is important when you use MSGIFAC=QUESSI, QSSIAT, or SSIEXT which can create a backlog of messages while CNMCSSIR is inactive. The MAXMQOUT setting must be high enough to prevent the SSI address space buffers from running out of space, but low enough so that the NetView automation is able to process the traffic rate. Setting MAXMQOUT for CNMCSSIR can also reduce message flooding during system error recovery or other periods of high system message rates. Consider using automation of messages BNH161I, BNH162I, and BNH163I to anticipate storage depletion, and using the OVERRIDE command for CNMCSSIR MAXMQOUT value to slow down the rate at which MVS messages are being received.

---

**NetView Automation Table**

Automation of system and network messages and MSUs is accomplished with automation statements (IF-THEN, END, ALWAYS, SYN, and %INCLUDE) in the NetView automation table, which is activated by the AUTOTBL command. Messages and MSUs are passed through the active NetView automation table to sequentially search one or more of the following automation table statements that indicate actions to be automatically performed for that message or MSU.

- **IF-THEN** contains conditions to be tested and actions to be taken if the conditions are satisfied. Actions that can be taken include a direct response to a system message, a command or command procedure to be executed, or presentation modification. Multiple actions can be specified on a single IF-THEN statement.
- **END** indicates the end of a BEGIN/END section that can be used to logically segment automation statements.
- **ALWAYS** indicates that certain actions are always to be done if the sequential search through the active automation table reaches the ALWAYS statement.
- **SYN** defines table synonyms that represent values that can subsequently be substituted within other automation table statements.
- **%INCLUDE** physically segments the table into multiple members or files.

If an automation table is activated as a result of a NetView AUTOTBL command, a compact, preprocessed version of the automation table is loaded into NetView storage. This allows a quicker search of the active NetView automation table than if the actual automation members are scanned when a message or MSU is processed. SYN and %INCLUDE statements do not affect the processing time of messages and MSUs. These statements are processed when the AUTOTBL command is processed, not when each message or MSU is processed.

You can decrease the processing time for messages and MSUs by following specific automation table design guidelines. The guidelines in the rest of this section decrease the average search time for the NetView program to locate matching statements and to perform actions.

As you make changes to improve the efficiency of the automation table, you should monitor the performance of the table using the AUTOCNT command. See "The AUTOCNT Command" on page 11 for more information.
Using BEGIN/END to Improve Efficiency

- Because the NetView automation table is searched sequentially (top to bottom), logically segmenting the automation table using BEGIN/END sections provides a large performance advantage. Using BEGIN/END reduces the number of automation statements that must be scanned. You should design your automation table so that each message prefix (such as DSI), each MSU major vector (such as major vector X’0000’), and specific subvectors have BEGIN/END sections.

If you dedicate BEGIN/END sections to the message and MSU classes, the number of statements that must be scanned is smaller than if all of the IF-THEN and ALWAYS statements are listed. If each message class has its own unique BEGIN/END section, placing ALWAYS as the last statement in a section prevents the automation table from being scanned further if no automation table match is found.

- Order BEGIN/END sections by frequency of use. If a large number of VTAM messages are received, a BEGIN/END section to handle IST messages should be at or near the top of the automation table to decrease the number of statements that must be evaluated before the correct BEGIN/END section is found.

- Order the statements within BEGIN/END sections by frequency of use. Minimizing the number of statements that must be searched to find the correct matching statement reduces processing.

Using Other Techniques to Improve Efficiency

The following guidelines are other techniques you can use to decrease the average search time for the NetView program to locate matching statements and to perform actions.

- Place statements for entire classes of messages or MSUs that are frequently received, but not processed near the top of the table or BEGIN/END sections. IF-THEN statements and ALWAYS statements can specify that no action be performed, eliminating unnecessary processing time. For example, if you do not automate commands issued at a NetView terminal, you can add the statement in the following example at the top of your table:

  IF HDRMTYPE = '*' THEN ;

- Some comparison items take longer to evaluate than others. Comparison items with the potential to be relatively slow include MSUSEG comparison items that specify complex locations, the IBM-supplied automation table function (ATF) program DSICGLOB, and any lengthy ATF programs you have written. Isolate these items by placing them in BEGIN/END sections started with an IF-THEN statement so that the NetView program evaluates the items only when the comparison in the IF-THEN statement is true. You can also isolate items by placing them after a logical-AND operator (&). In this case, the NetView program evaluates the items only if the conditions before the & are met.

- Avoid unnecessary NetView automation processing by using the operating system message facility (MPF for MVS) to suppress unneeded messages. Only those system messages that require NetView automation or an operator’s action should be forwarded to the NetView program. See “Limiting System Messages” on page 6 for more information.

- Eliminate simple command procedures whose function can be processed in the automation table directly. Consider the following examples:
  - Instead of calling a command list from the automation table to check the value of a NetView global variable, you can bypass the processing needed to execute the command list by checking the global variable value directly in the automation table using the DSITGLOB or DSICGLOB ATFs.
– The THRESHOLD comparison item is useful for specifying particular actions to take place when a condition has happened at least a specified number of times within a specified time period. Using THRESHOLD in the automation table is more efficient than calling a command procedure to make a similar determination. The THRESHOLD counters are reset when a new automation table is loaded.

– If you call a command procedure to evaluate a complex condition that the existing automation table comparison items cannot address, consider writing your own ATF to perform this processing.

- Minimize use of the CONTINUE automation table action that scans the entire table. The CONTINUE action is useful for things such as setting defaults for the table or a section of the table and then continuing the search for an additional match, as specified by the statement in the following example:

```
ALWAYS SYSLOG(Y) NETLOG(N) DISPLAY(Y) CONTINUE(Y) ;
```

If specified on every statement, CONTINUE(Y) increases processing time. The entire automation table must be searched to determine all automation actions that should take place. If you use CONTINUE(Y) to determine multiple IF-THEN statements that all result in actions to be performed, design your table so that scanning does not need to continue through the entire table by doing one of the following:

– Specify CONTINUE(N) on the last statement where you expect to find a match for a particular message or MSU.

– Specify ALWAYS as the last statement of the BEGIN/END section to prevent continued statements from being scanned outside of an enclosing BEGIN/END section.

- If automation slows because many unsolicited messages are queued on a single task waiting for automation table processing, you can use the ASSIGN command to split the messages among several tasks. If you use the ASSIGN command, you can still use the automation table for final routing of the message; the ASSIGN command just gets the message to the automation table faster.

For more information about the NetView automation table, message routing, and ASSIGN processing, refer to [Tivoli NetView for z/OS Automation Guide](#).

### The AUTOCNT Command

The AUTOCNT command produces a report describing the usage of automation table statements in the active NetView automation table. You can also use the AUTOCNT command to reset the automation table usage counters. You can display the report as multiline messages or place the information in a file.

The AUTOCNT command can request information and statistics on message-type automation statements, on MSU-type automation statements, or both. You can request summary information or detail. The detailed information tells you how many messages and MSUs were compared to each automation table statement, and how many matched.

Message CNM493I is written to the network log each time a match is found in the NetView automation table that results in the scheduling of a command or command list for execution with the EXEC action. The sequence number (if any) and the member name of the statement that executed the command or command list are included in the CNM493I message. Use this message to determine how frequently commands are executed from the automation table and by which automation table statements. If you use the AUTOCNT command to analyze
automation table activity, you might want to prevent the CNM493I message from being written to the network log. You can use the CNM493I parameter of the DEFAULTS and OVERRIDE commands or the CNM493I action in the automation table to control whether the message is written.

If you use the AUTOCNT command to analyze automation table activity, you might want to prevent the CNM493I message from being written to the network log. You can use the CNM493I parameter of the DEFAULTS and OVERRIDE commands or the CNM493I action in the automation table to control whether the message is written.

For syntax and other information about the AUTOCNT command, and for information about the CNM493I action of the automation table, refer to the NetView for z/OS Automation Guide For information on the CNM493I option of the DEFAULTS and OVERRIDE commands, refer to the NetView online help.

**Detail Reports**

Figure 8 illustrates the output of a detail report for MSG-type automation table statements. Detail reports for MSU-type automation table statements provide the same types of data.

```
- DW08001 AUTOMATION TABLE MSG DETAIL REPORT BY OPER1

DW0803I --------------- ( AUTOSEG1 MESSAGE DETAILS 03/30/00 14:32:42 ) ---------------
DW0805I |<-- PERCENTAGES -->|
DW0806I STMT SEQ MEMBER COMPARE MATCH E C A MATCH/ COMP/ MATCH/ COUNT COUN TCII COMP TOTAL TOTAL
DW0807I NUMBER NUMBER NAME COUNT COUNT C I I COMP TOTAL TOTAL

DW0808I -----------------------------------------------------------------------------------

DW0809I 00001 00008000 AUTOSEG1 2304 798 34.6 100.0 34.6
DW0809I 00002 00001000 AUTOSEG1 798 177 22.2 34.6 7.7
DW0809I 00003 00001400 AUTOSEG1 621 9 1 1.4 27.0 0.4
DW0809I 00004 00001600 AUTOSEG1 612 0 1 0.0 26.6 0.0
DW0809I 00005 00002000 AUTOSEG1 612 612 X 100.0 26.6 26.6
DW0809I 00007 000012700 AUTOSEG1 1506 160 10.6 65.4 6.9
DW0809I 00008 00002900 AUTOSEG1 160 52 2 32.5 6.9 2.3
DW0809I 00009 00003400 AUTOSEG1 108 1 0.9 4.7 0.0
DW0809I 00010 00003700 AUTOSEG1 107 107 X 100.0 4.6 4.6

 Figure 8. Message Detail Report

In addition to the statement number, sequence number, and member name, the detail report contains the following information for each automation table statement.

- Conditional comparisons (COMPARE COUNT)
  The counter that is incremented when the associated conditional statement is selected for evaluation.

- Evaluation matches (MATCH COUNT)
  The counter that is incremented when the associated conditional statement is evaluated as true, resulting in execution of all automation actions specified on the statement.

- Executed commands (E C)
  This column reports the number of commands that are executed for this automation statement when there is an evaluation match. If the number of EXEC actions with CMD keywords is greater than 99, an asterisk (*) appears in the column.

- Continue indicator (C I)
A report column marked X indicates that the conditional statement contained a CONTINUE action, causing the NetView program to continue to scan the automation table. CONTINUE(Y) actions cause additional conditional processing for later statements in the table, and can enable a conditional match on additional statements.

- **Always statement indicator (A I)**
  A report column marked X indicates that the statement was an ALWAYS. For ALWAYS statements, the MATCH/COMP field is always 100%.

- **Match to compare percentage (MATCH/COMP)**
  A statistic calculated by dividing the ratio of MATCH COUNT by the COMPARE COUNT of the conditional statement, multiplied by 100. If the number of matches and the number of comparisons are both zero, the ratio is shown as -.- to indicate division by zero.

- **Compare percentage (COMP/TOTAL)**
  A statistic calculated by dividing the ratio of COMPARE COUNT of the conditional statement by the total number of messages or MSUs, multiplied by 100. If the number of comparisons against this statement and the total number of messages or MSUs processed by automation are both zero, the ratio is shown as -.- to indicate division by zero.

- **Match percentage (MATCH/TOTAL)**
  A statistic calculated by dividing the ratio of MATCH COUNT of the conditional statement by the total number of messages or MSUs, multiplied by 100. If the number of matches for this statement and the total number of messages or MSUs processed by automation are both zero, the ratio is shown as -.- to indicate division by zero.

Any numeric column value that exceeds 99999999 is overwritten with eight asterisks (*)

**Suggestions for Using Detail Reports**

- The report output represents a snapshot of the automation table data, and that automation might be in progress for some messages or MSUs while the data is being collected. There might be minor discrepancies between totals in a summary report and values in a detail report produced at the same time.

- The percentage fields can be interpreted intuitively:
  - COMP/TOTAL is the percentage of messages or MSUs processed through the table that compare against this statement. The first statement always has a COMP/TOTAL percentage of 100%, because all messages or MSUs compare against the first statement.
  - MATCH/TOTAL is the percentage of messages or MSUs processed through the table that match on this statement.
  - COMP/MATCH is the percentage of messages or MSUs compared against this statement that match.

- Smaller COMP/TOTAL percentages indicate better performance. You can reduce the COMP/TOTAL percentages by using BEGIN/END sections and by ordering statements within sections by match frequency or MATCH/TOTAL percentage. In other words, statements with higher MATCH/TOTAL percentages should be located early in their BEGIN/END sections. Likewise, BEGIN-END sections with higher MATCH/TOTAL percentages should be located early in the automation table.

- COMP/MATCH percentages of 100% are expected on ALWAYS statements. The ALWAYS indicator field (A I) is included in the report so that you can spot these
without referring to a LISTING file. If you have other statements with a COMP/MATCH percentage of 100%, you should determine whether the statement is coded correctly.

- The executed commands field (E C) multiplied by the MATCH COUNT field equals the number of commands executed as a result of matches on the automation table statement. This product (E C times MATCH COUNT), summed for all automation table statements, represents the total number of commands executed from the automation table. This total is shown as the TOTAL COMMANDS EXECUTED field in the summary report.

- Examine statements with high MATCH COUNT values that execute commands.
  - Use the LOADCL command to preload command lists that are executed frequently.
  - For command processors that are executed frequently, ensure that RES=N is not coded on the CMDMDL statement in initialization member DSICMD.

For more information on making command lists and command processors resident, see “Preloading Command Lists” on page 28 and “Command Processors” on page 39.

Summary Reports

Figure 9 illustrates the output of a summary report for MSG-type automation table statements. Summary reports for MSU-type automation table statements provide the same types of data, except for TOTAL ROUTES EXECUTED, because the ROUTE keyword of the EXEC action is not applicable to MSUs.

- DW0801I AUTOMATION TABLE MSG SUMMARY REPORT BY OPER1

DWO810I ------------( AUTOSEG1 MESSAGE SUMMARY 03/30/00 14:32:42 )-------
DWO812I STATISTICS STARTED = 03/30/99 13:32
DWO813I TOTAL MSGS PROCESSED = 2304
DWO814I MSGS MATCHED = 958
DWO815I MSGS RESULTING IN COMMANDS = 9
DWO816I TOTAL COMMANDS EXECUTED = 9
DWO817I TOTAL ROUTES EXECUTED = 1
DWO818I AVERAGE COMPARES/MSG = 2.58
DWO819I TOTAL MSGS/MINUTE = 38
DWO820I MINUTES ELAPSED = 60
DWO880I -------------------------------------------------------------------

Figure 9. Message Summary Report

The summary report contains the following information.

- Date and time of usage report generation.
  The date is in the format mm/dd/yy. The time is in the format hh:mm:ss, where hh is based on a 24-hour clock. The date and time are reported in the label message for the SUMMARY statistics (messages DWO810I and DWO811I).

- Date and time of start of usage count monitoring.
  The date is in the format mm/dd/yy. The time is in the format hh:mm:ss, where hh is based on a 24-hour clock. The date and time are reported in message DWO812I.

- Total number of messages or MSUs processed.
  A count of all the messages or MSUs that have passed through the automation table.

- Total number of messages or MSUs matched.
The number of messages or MSUs that were acted upon by at least one automation table statement. An ALWAYS statement causes a message or MSU to be considered a match.

- Number of messages or MSUs resulting in command execution.
  A count of the number of messages or MSUs that resulted in one or more commands being executed from automation table statements.

- Total commands executed for messages or MSUs.
  The total number of commands executed by all automation table statements during the period when statistics were taken. The EXEC action with the CMD keyword indicates a command is executed from the automation table.

- Total routes executed for messages.
  The total number of routes executed by all automation table statements during the period when statistics were taken. The EXEC action with the ROUTE keyword (and without the CMD keyword) indicates a route is executed from the automation table.

- Average number of compares per message or MSU.
  The number of compares divided by the number of messages or MSUs that had passed through the automation table.

- Average number of messages or MSUs processed per minute.
  The number of messages or MSUs processed by the NetView automation table divided by the number of minutes since the last reset or load of the automation table.

- Number of minutes elapsed.
  The amount of time, in minutes, since the last AUTOCNT RESET command or since the current active automation table was activated.

Using Summary Reports

The following suggestions can help you use the information in the summary tables:

- The AVERAGE COMPARES/MSG field or AVERAGE COMPARES/MSU field is a good indicator of the efficiency of the automation table. Performance improvements to the automation table, such as increased use of BEGIN/END sections, usually result in a decrease in the AVERAGE COMPARES/MSG value. Monitor this value before and after making changes to the automation table to ensure that your changes do not have a negative effect on performance.

- The TOTAL MSGS/MINUTE field or TOTAL MSUS/MINUTE field is the TOTAL MSGS PROCESSED field divided by the MINUTES ELAPSED field. You can calculate other rates, such as total commands executed per minute or total routes per minute, by performing the arithmetic on the appropriate fields. Increased filtering by the operating system message processing facility should reduce the TOTAL MSGS/MINUTE rate.

- Consider setting an EVERY timer under an autotask to invoke the AUTOCNT command periodically. The AUTOCNT output can be used as important historical data in spotting trends in automation activity. Because summary reports are shorter than detail reports, you might create them more frequently, perhaps every hour, to monitor activity rates.

- The automation table statistics are set to zero when an automation table is loaded. Prior to loading a new table, you might want to issue the AUTOCNT command for the old table to collect its statistics.
Automating Hardware Monitor Records

You can automate problem notifications sent to the hardware monitor by generating messages and sending the messages to the automation table for processing. Many problem records sent to the hardware monitor are management service units (MSUs). You can generate messages from the MSUs or automate MSUs directly using the MSUSEG and HIER conditions in the NetView automation table. Direct automation is more efficient than automation of records converted to messages.

Refer to "Tivoli NetView for z/OS Automation Guide" for more information about automating MSUs.

Filtering Hardware Monitor Records

Filtering unnecessary hardware monitor records prevents processing by the NetView program, VSAM, and your hardware monitor operators. The following filter types are available:

- The ESREC filter determines whether the record should be sent to the events and statistics database and whether the record is eligible for the remaining filters.
- The AREC filter determines whether the record should be sent to the alerts database, and thus be made available to operators viewing the hardware monitor alerts panels. The AREC filter also determines whether the record is eligible for the remaining two filters.
- The OPER filter determines whether BNJ146I and BNJ030I messages should be created from the alert for automation.
- The ROUTE filter determines whether the alert should be forwarded to the hardware monitor focal point.

You can set each of the filter types to BLOCK or PASS. A filter set to BLOCK prevents the filter or any filters dependent on it from taking effect. For example, an AREC filter set to BLOCK prevents the OPER and ROUTE filters from receiving the alert.

You can set the hardware monitor recording filters using the hardware monitor SRFILTER command. You can set filters for MSUs that undergo automation processing using the SRF automation table action. You can use the automation table to override settings made using SRFILTER. Both SRFILTER and the automation table can set color and other highlighting attributes.

Filtering unnecessary hardware monitor records from the events and statistics database and the alerts database can increase the productivity of operators and save substantial processing time. Use the following guidelines when setting filter options:

- Allow required events to pass the ESREC and AREC filters.
- Set the ROUTE filter to PASS only for alerts that must be passed to a focal point system.
- Set the OPER filter to PASS only for alerts that you want to automate by creating BNJ146I and BNJ030I messages.

For more information on hardware monitor filters, see "Hardware Monitor Filters" on page 45.
Automation Tasks (Autotasks)

An automation task (autotask) is an operator station task (OST) that is not associated with a terminal. Autotasks can be used to monitor status and to issue responses to system messages.

Do not use the following under an autotask:

- The &PAUSE statement for the NetView command list language.
- The PARSE PULL and PARSE EXTERNAL statements for REXX.
- The WAIT FOR OPINPUT statement for high-level languages.

These statements do not provide a time-out facility. If you pause in a command list executed under an autotask, use a facility that includes a GO command, because the GO command breaks the wait implied by the paused statement.

Using Multiple Autotasks

Using multiple autotasks has the following advantages:

- **Improved reliability**
  
  If you have only one autotask and it is logged off or disabled, this can affect the entire automation process. Having multiple autotasks minimizes the impact of an autotask failure.

- **Improved problem determination**
  
  For example, if VTAM fails, having an autotask specifically designed for VTAM processing makes it easy for an operator to determine the autotask recovery stage.

- **Improved audit trail**
  
  It is easier to determine what work has been done by an autotask (for example, by looking at the network log) when the work is divided among multiple tasks.

For an MVS environment, you can use multiple autotasks to distribute the automation workload across multiple processors, thus taking advantage of multitasking to improve throughput. Throughput for an automation workload can be constrained by contention for host processors or by contention for the command list data control block (DCB), which synchronizes input and output (I/O) to the command list data set.

For an automation workload in which the command lists have been preloaded using the LOADCL command, no I/O is necessary to the command list data set; therefore, processor capacity is the only constraint on automation throughput. When an \( n \)-way processor (where \( n \) is the number of processing engines) is used with such a workload, additional autotasks beyond the number \( n \) probably do not offer a significant throughput improvement.

This does not imply, however, that there is no value in using more than \( n \) autotasks in an \( n \)-way processor environment. For example, it can be useful to separate the automation of critical messages from the rest of the automation workload by assigning them to their own autotask.

Using Full-Screen Automation

Using full-screen automation, a REXX, PL/I, or C program can perform any NetView full-screen component tasks that a NetView operator could perform using the same component and a 3270 display. Full-screen automation is intended to be
used as an automation tool. It is not intended to be used by NetView operators to create fast paths or to simplify commands.

Using full-screen automation, an application program is able to:
• Read data from a NetView application panel
• Write data to a NetView application panel
• Execute the PF, PA, Enter, and Clear function keys on a NetView application panel

Sample CNMS1098 is included with the NetView program. This sample provides an example for coding and using the full-screen automation function. CNMS1098 uses a BGNSESS command to logon to TSO from the NetView program and display data from the TSO log on the NetView user's panel.

Full-screen automation uses a new task type, the VOST, or 'Virtual OST'. The VOST is like an OST, except that the 'session' is a pipe to an owning task. Full-screen automation provides two new commands (ATTACH and DETACH) to enable procedures to invoke, control, and end full-screen applications.

The ATTACH command is used to begin a VOST and assign (queue) work to it. The DETACH command causes a VOST started by an ATTACH to terminate.

Two commands show status of VOSTs currently active.
• LIST STATUS=VOST will show a line like the following:

  TASKNAME: DSI#0001 OWNER: OPER2 ATTACHN: NLDM STATUS: ACTIVE

• TASKUTIL will show tasks with TYPE=VOST if there are VOSTs active:

<table>
<thead>
<tr>
<th>TASKNAME</th>
<th>TYPE</th>
<th>DPR</th>
<th>CPU-TIME</th>
<th>N-CPU%</th>
<th>S-CPU%</th>
<th>MESSAGEQ</th>
<th>STORAGE-K</th>
<th>CMD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSI#0001</td>
<td>VOST</td>
<td>250</td>
<td>0.01</td>
<td>42.29</td>
<td>0.10</td>
<td>120</td>
<td>NLD</td>
<td>N/A</td>
</tr>
<tr>
<td>DSIMONIT</td>
<td>OPT</td>
<td>255</td>
<td>2.78</td>
<td>16.45</td>
<td>0.04</td>
<td>4</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>AAUTSKLP</td>
<td>DST</td>
<td>247</td>
<td>0.17</td>
<td>14.54</td>
<td>0.03</td>
<td>0</td>
<td>1552</td>
<td>N/A</td>
</tr>
<tr>
<td>CNM01PPT</td>
<td>PPT</td>
<td>255</td>
<td>3.29</td>
<td>12.52</td>
<td>0.03</td>
<td>0</td>
<td>164</td>
<td>N/A</td>
</tr>
<tr>
<td>DSILog</td>
<td>DST</td>
<td>254</td>
<td>0.15</td>
<td>4.31</td>
<td>0.01</td>
<td>0</td>
<td>19</td>
<td>N/A</td>
</tr>
<tr>
<td>OPER2</td>
<td>OST</td>
<td>251</td>
<td>1.73</td>
<td>3.87</td>
<td>0.01</td>
<td>0</td>
<td>287</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Notes:
1. DSI#xxxx is not an available choice; it is assigned by the NetView program when an ATTACH command is issued.
2. The example shows only one VOST active at this time. TASKUTIL shows all active tasks, including VOSTs. The TYPE column shows the type of each active task.

On average, full-screen automation uses more processor resources than manual execution of commands or procedures. All storage allocated during the ATTACH phase of full-screen automation is released during the DETACH step, except for the normal storage growth seen when an OST logs on. The OST storage growth is released when the task from which the ATTACH was issued is terminated. Because of the potential below-the-line storage growth if many VOSTs are active at once, careful consideration should be used when determining what to automate using full-screen automation. This is true in cases where full-screen automation is used with tasks that are not normally logged off.
Command and Message Forwarding

You can transmit commands and messages between a distributed system and a focal point using either the RMTCMD command or OST-NNT sessions (see “NetView-NetView Communication” on page 116 for more information). If all of your systems run NetView Version 2 Release 2 or a later release, RMTCMD is the recommended method of command and message forwarding. RMTCMD uses the high performance LU 6.2 transport for better performance and does not require operators to manually start sessions before forwarding commands. You can use OST-NNT sessions to communicate with earlier releases of the NetView program. Refer to Tivoli NetView for z/OS Automation Guide for more information about command and message forwarding.

If you decide to use OST-NNT sessions for message forwarding, consider using persistent sessions over leased lines to connect the distributed system and the focal point. Using persistent sessions avoids the overhead of setting up and taking down nonpersistent sessions. Using leased lines instead of switched lines avoids subarea dial overhead.

The NetView program provides message-forwarding samples that demonstrate the use of OST-NNT sessions. The samples include capabilities for backup and intermediate focal points, as well as persistent and nonpersistent sessions. If you decide to use these samples and you expect to forward more than two or three messages each minute, use the LOADCL command to preload the NetView command lists that accomplish the message forwarding (the CNME70xx series). See “Command Lists” on page 28 for more information about preloading command lists.

Separation of the Automation Workload from Other NetView Workloads

In some environments, separating the automation workload from other NetView workloads can offer advantages in automation responsiveness and availability.

You can separate workloads by:

- Running multiple copies of the NetView program
- Using z/OS Work Load Manager (WLM) enclaves to separate system and network automation workloads within a single NetView address space

Multiple NetView Programs

You can use one copy for system automation and another copy for network automation and problem determination.

Although separating the automation workload from other NetView workloads can offer advantages, a disadvantage is that dividing the NetView workloads between different address spaces might not decrease the overall NetView host processor utilization and it might increase overall NetView storage use.

If you plan to use multiple copies of the NetView program, consider setting the dispatching priority of the NetView program that performs system automation above VTAM and the subsystems being automated. The NetView program performing network automation and problem determination should have a dispatcher priority below VTAM and critical applications.

Refer to Tivoli NetView for z/OS Automation Guide for more information about running multiple copies of the NetView program.
Single NetView Program Using WLM Enclaves

To use the Work Load Manager (WLM) for separating system and network automation workloads within the NetView program, perform the following steps:

1. Either code or uncomment the WLM style statement.

   **Note:** If the WLM style statement is not coded or uncommented, the NetView program does not use the WLM services. A sample WLM style statement is shipped in the NetView style file but is commented out.

   For more information, refer to *Tivoli NetView for z/OS Installation: Configuring Additional Components*.

   To code the WLM style statement in the style file, the name that you code for the SubSystemName keyword is the subsystem that WLM recognizes as the specific instance of the NetView program.

   Uncommenting the style statement sets the WLM subsystem name to the NetView domain ID.

2. **Before** you start the NetView program, create a WLM service definition. The service definition consists of at least the following:
   - A service policy
   - A workload
   - A service class

   When separating the management of the NetView program’s network and system automation (SA) subtasks, the NetView program creates z/OS enclaves to manage those two sets of subtasks so that users can assign different performance goals to the enclaves. System automation subtasks include all those not connected with network management.

   These two types of NetView enclaves should be classified to service classes with velocity goals and relative importance assigned. The goals should have approximately the same velocity value, but the goal assigned to NetView system automation enclaves should be more important than the goal assigned to any NetView network enclaves. There is no need to define a separate service class for NetView, if existing service classes in your service definition satisfy these conditions.

   If SA/390 or other system automation is used, a *high* velocity goal and importance could be assigned. For non-system automation NetView subtasks, a *medium* velocity goal and importance could be assigned. For example, a goal of velocity = 50 and an importance of 1 could be assigned. For non-system automation NetView subtasks, a goal of velocity = 40 and an importance of 2 could be assigned to give appropriate weighting to the system automation NetView subtasks.

   For instructions on defining performance goals and service classes to z/OS, refer to the IBM publication *z/OS MVS Planning: Workload Management*.

   - **Service classification rules**

     NetView subtasks are classified into a service class based on one or more of the following work qualifiers:
     - Subsystem type
       - NETV
     - Subsystem instance (SI)
       - The NetView WLM SubSystemName style statement value
The WLM service definition must be installed on a WLM couple data set, and a WLM service policy using that WLM service definition must be activated.

After the NetView program is started, every NetView subtask is classified into a WLM service class (based on the customer coded WLM classification rules for the NETV subsystem type). The NetView program joins the subtask to a WLM enclave with the same service class.

If a WLM enclave for the service class does not exist, the NetView program tells WLM to create one. When the NetView subtask ends, the NetView program tells WLM to delete the enclave.

If no NetView subtask is joined to a particular enclave (created by the NetView program), the enclave is deleted.
Chapter 3. Tuning for AON

AON performance considerations are described in this chapter.

Tuning Techniques

Consider the following NetView techniques when tuning AON:

- Perform NetView tuning according to the applicable information for your release of NetView. You should pay particular attention to session monitor tuning and to VSAM tuning.
- In any online system, I/O to databases is of primary importance. Do not put the NetView logs, the session monitor databases, the AON automation log (if used) and the AON status file all on the same DASD, or on DASD that is highly utilized.
- Do not specify secondary extents for logs if you are using AUTOFLIP.

The DSICMD Resident Option

The resident option can offset performance. Ensure that all modules are designated RES=Y in the CNMS1005 sample that become DSICMD in NetView remain. All AON modules designated RES=Y in the EZLCMD sample should remain.

You can make programs and REXX procedures resident with the NetView LOADCL command. This is done automatically at AON initialization if the programs and procedures are specified with the RESIDENT entry in the control file. A number of such entries are shipped in the EZLCFG01 and FKVCFG01 samples. This list can be added to, or can be reduced if analysis of the MAPCL output shows infrequently used procedures.

Ensure other programs and procedures are resident as appropriate. Look in the NetView log at message CNM493I. This message indicates that a command procedure was issued out of the automation table. All such procedures should be resident for maximum performance. You can also use tracing to observe the flow of AON tasks.

Note: Tracing temporarily increases system usage. Consider making resident any procedures you see which are not already resident.

DSICLD Library

In addition to program residency, DSICLD library concatenation can also affect performance. Keep the number of libraries to a minimum, to reduce program search time. You may want to have two user program libraries, a small one concatenated ahead of CNMCLST and a larger one after CNMCLST. For maximum efficiency each library should reside on a different storage director.

Automation Table

You can use the BEGIN/END feature to group all of your messages together in short sections. This cuts automation table processing time considerably. Change the order of the IF THEN statements in the automation table, so that the statements are ordered by FREQUENCY, which places the most frequently processed message in the first positions.
at the TOP. You can determine this by issuing the AUTOCNT command. For more information about the AUTOCNT command, refer to the NetView online help.

Code early outs in the automation table. That is, if you are aware of messages that are being passed to NetView and are not driving programs out of the automation table, code IF THEN statements to exit near the top of your table. For example, if you know that there are no message triggers for IES messages (and these messages are not being stopped by MPF list with AUTO=N), you can code the following early out in the top of your automation table:

```
IF TEXT(1)="IEF". THEN DISPLAY(N)
```

Ordinarily, messages such as these require an entry in the MPF list specifying AUTO=N, so NetView would not have to process the messages at all. If you are using another NetView for console automation, these messages must come over the subsystem interface (SSI). In these cases, early outs are necessary in the AON automation table.

To further tune message processing, follow all of the preceding suggestions and then at the bottom of your automation table, code an entry that traps on all other messages (IF MSGID=.) and runs a program that writes that message to a file. Then, do whatever possible to omit them from processing.

**DDF Tree and Panel**

Ensure EZLPNLS contains all DDF panels that you want to access. This slows down DDF initialization just a bit, but speeds up operator access and DDF update later.

Use the dynamic updating of resources by type (as shipped with AON) as much as possible. Avoid hardcoding resources onto panels and avoid having them as unique entries in EZLTREE. Keep EZLTREE as small as possible.

**Node Automation**

AON does not know when a resource goes away suddenly, or if the outage was intentional, unless an operator has specified it as inactive. For example, you may have specified a switched PU on a token ring or an X.25 line is to be dropped after business hours. If your network has many such devices, use RECOVERY control file entries with the NOAUTO parameter to forestall pointless recovery attempts during off hours. You can code these statements generically with wildcard names, if you have a naming convention, to decrease the number of statements required.

**Operations**

If you must recycle AON NetView during peak times of the day, do not code DDFREFRESH=Y on the ENVIRON SETUP control file entry. This means that DDF does not know about resources that are already down, but it is updated when they are running again. You can add them later by a restricted NETSTAT call, or manually with DDFADD.

NETSTAT uses considerable cycles if TYPE=PHYSICAL is required. To manually add an AON resource to DDF, you need to know the status you want it to show, which is determined by priority. The priority value is found on the DDF entry in the control file (PR=nnn). For example, the priority for inactive is 150 (DDF INA*,PR=150,CLEAR=Y). When you add the resource to DDF with an inactive status, you want to do so in such a way that DDF deletes it automatically when
the resource comes active. This requires that you specify on the DDFADD the same
basic information that AON does. Specify the information in the following format:

```
DDFADD sysname.resource_name(resource_type),
RV=resource_name,IN=/resource_name/,DA='some text',
PR=nnn
```

The DA field must contain some message text but it need not be the same text
AON uses.

Use DBMAINT sparingly online. It is important to clean up and reorganize your
databases to maximize performance, but DBMAINT uses a lot of cycles and should
not be used during peak periods. Set a timer to run it during off hours.

Keep the number of operators who are notification operators to a minimum to save
CPU cycles. Many operators often prefer DDF to the messages.

### TCP/IP Support for AON

To reduce monitoring overhead, avoid monitoring non-critical resources. Define
low monitor intervals only for the highly critical resources and define greater
monitor intervals for resources which may be less critical.

Monitor the TASKUTIL message queues and DISPPI buffer queues and increase the
number of TSO servers and AUTTCP tasks and AUTMSG tasks as required to
reduce the backlog of queues.
Chapter 4. Tuning for Command Procedures

A command list is a list of commands and statements designed to perform a specific function for the user. For the NetView program, command lists can be written using the NetView command list language or REXX. In the NetView program, a command processor is a module written in a high-level language (HLL) or assembler language and is invoked as a command. A command procedure is either a command list or a command processor written in PL/I or C.

For more information on command lists and command processors, see the following publications:
- Tivoli NetView for z/OS Customization Guide
- Tivoli NetView for z/OS Customization: Using Assembler
- Tivoli NetView for z/OS Customization: Using PL/I and C
- Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language

Tuning Techniques

Following are the major tuning techniques for command procedures, arranged in order of expected effect on performance; the most important tuning considerations are listed first. These recommendations are described in detail in this chapter.

1. Preload frequently used command lists using the LOADCL command. Consider invoking a command list to issue LOADCL, loading the most frequently used command lists last. See “Command Lists” on page 28.
2. For systems using PL/I command processors, define preinitialized environments for your frequently run PL/I programs. You can use the HLLENV LIST STATS command to help you decide how many preinitialized environments you need. See “Running PL/I Programs in a Preinitialized Environment” on page 40.
3. For systems using PL/I or C command processors, minimize or eliminate the I/O needed to find and read the run-time routines during command processor execution. See “Command Processors” on page 39.
4. Preload frequently used command processors by not specifying RES=N on their CMDMDL statements in DSICMD. See “Command Processors” on page 39.
5. Consider using the REXX/370 compiler to compile REXX command lists. Compiled REXX command lists are more efficient than interpreted REXX command lists. See “Compiled REXX/370 Command Lists” on page 31.
7. For systems using REXX command lists, group frequently used external functions and subroutines in REXX function packages. You can also modify the sample file CNMSJ1M11 to minimize the search time for NetView system functions. See “REXX Function Packages” on page 32.
8. Use task global variables instead of common global variables where possible. The processing demands of task global variables are smaller than the processing demands of common global variables. See “Global Variables” on page 42.
9. If you expect to use more than 200 task or common global variables, specify
the expected number of variables in the NetView constants module
DSICTMOD. You can use the QRYGLOBL command to display the expected
number of variables coded in DSICTMOD, as well as the actual number of
variables found. See "Global Variables" on page 42.

10. Consider using or adapting the AUTODROP command list (CNMS8003) to
manage preloaded command lists. See "Managing Command Lists with
AUTODROP" on page 29.

Command Lists

Command lists that are written using the NetView command list language or
REXX are interpreted. That is, statements in the command list are translated and
executed one at a time.

Before being interpreted, the entire command list is read into storage (unless you
used LOADCL). However, trailing blanks are removed from command lists to save
storage. Each time a command list is invoked, it is read from the command list
data set (DSICLD). When users modify command lists, changes are put into effect
immediately. Nested command lists are read into storage as they are invoked.

Preloading Command Lists

You can preload frequently used command lists into storage with the LOADCL
command. Preloading command lists saves time. When a preloaded command list
is invoked, it executes immediately. There is no I/O delay. Preloaded command
lists are placed above the 16 MB line in MVS. If command lists are preloaded,
every operator uses the same copy. If command lists are not preloaded, multiple
copies of the same command list can be loaded when invoked by different
operators. This can cause storage problems depending on the size of the command
list and the frequency of invocations.

The DROPCL command removes a command list that was previously loaded into
storage by the LOADCL command. The MAPCL command displays the command
lists that have been preloaded, the date and time each was loaded into storage, and
the number of invocations for each command list since it was loaded.

If you plan to preload several command lists using the LOADCL command,
consider writing a command list to issue the LOADCL command invocations.
Order the LOADCL command invocations so that the most frequently used
command lists are loaded last.

You can use the detail report produced by the AUTOCNT command to select
command lists for preloading. Look for automation table statements with a high
match count that execute commands. Preload command lists that are executed by
these statements.

You can also use the PIPE stage INSTORE to add, delete, or replace frequently
used command lists into a pool of storage. The in-storage members are read from
storage instead of disk by DSIDKS disk services or any NetView process based on
DSIDKS, such as BROWSE or the < stage.

Using the STRIP stage with PIPE INSTORE enables you to strip trailing blanks
before loading, thus reducing storage demand. When the CRYPTO parameter is
used with PIPE INSTORE, the member is encrypted when it is loaded into
memory. This should be used only when the security of the member is important,
since additional processing is required to encrypt the data when this stage is executed and decrypt the data each time the member is read.

The INSTORE stage is used by the REXX CLIST CNME1054 (MEMSTORE) to automatically load frequently used members. MEMSTORE will use the LIST MEMSTAT command to monitor members with elevated disk usage, load these members into memory, or unload members that have relatively low usage from memory.

Using the two parameters for MEMSTORE will allow you to control the amount of storage allocated to in-storage members, which resides above the 16 MB line in MVS, and the monitoring interval used to obtain member usage information.

The amount of processing for MEMSTORE is directly related to the monitoring interval. LOADCL may run a CLIST slightly faster, but storage usage can be optimized better with MEMSTORE. Also, MEMSTORE will monitor and adjust CLIST loading based upon usage without user intervention.

Managing Command Lists with AUTODROP

A sample command list, AUTODROP (CNMS8003), provided by the NetView program can help you manage the number of command lists that have been preloaded into storage using the LOADCL command. AUTODROP uses the MAPCL and DROPCL commands to conditionally drop command lists from storage.

AUTODROP uses the following parameters:

- **Uses**: Specifies the minimum number of times the command list must be used to remain in storage. The default is 0.
- **Days**: Specifies the minimum number of days the command list must be loaded to be dropped from storage. The default is 0.
- **Hours**: Specifies the minimum number of hours the command list must be loaded to be dropped from storage. The default is 1.

**Example: Using the AUTODROP command**

The AUTODROP command drops any preloaded command lists with a use count of five or less that has been loaded for three hours or more, as shown in the following example:

```
AUTODROP 5 0 3
```

A command is not dropped from storage unless both the minimum time specified by the days and hours parameters has elapsed and the command list has been invoked the minimum number of times necessary (or less) since the command list was loaded.

**Determining Command Lists to Drop**

You can use the AUTODROP command list to manage commands preloaded with LOADCL. Use the following steps to determine the subset of the most frequently used command lists:

1. Preload a large subset of your command lists and then issue AUTODROP from a timer to remove infrequently used command lists.
2. Drop and reload the command lists that are to stay resident to reset their load times and use counts.
Note: Reload the command lists in ascending order of frequency, with the most heavily used ones last.

3. Preload another large subset and repeat the procedure until all command lists have been processed.

If the storage requirement for the most frequently used command lists is too high, consider making the AUTODROP criteria more stringent, with a higher use count or a shorter time interval.

Subroutines

Instead of using nested command lists for subroutines, consider calling the subroutine, as shown in the following examples:

- for the NetView command list language:
  &RETURN = -RI
  &GOTO -SUB1
  -RI...
  &EXIT
  -SUB1...
  &GOTO &RETURN

- for REXX:
  CALL SUB1...
  EXIT
  SUB1: PROCEDURE...
  RETURN

If you use nested REXX command lists, do not use the CALL statement to invoke them. Instead, place the command list name in single quotes. The CALL statement causes the system libraries to be searched before the NetView command list library. When the command list name is in single quotes, the command list name is passed to the NetView program, and only the NetView library is searched.

REXX Command Lists

Most command lists written in REXX perform more efficiently than command lists written in the NetView command list language. This is true for command lists that perform mathematical functions or string manipulation. However, processing global variables in REXX command lists can require more processing time than NetView command list language command lists. Note the following considerations:

- When global variables are identified in NetView command lists by &TGLOBAL and &CGLOBAL statements, all subsequent assignments to the variables cause updates to the global variables. In REXX, the programmer has explicit control over access to global variables using the NetView GLOBALV command.

- When the NetView program processes a GLOBALV command issued from within a REXX command list, it accesses both the global variable and the REXX local variable corresponding to the global variable. Accessing the REXX local variable during NetView GLOBALV command processing requires processing time not needed for NetView command lists.

- You can minimize the use of the GLOBALV command in REXX to optimize the NetView global variable processing. With REXX command lists, the REXX local variable corresponding to the global variable can be manipulated many times without updating the global variable.
REXX local variables are accessed while processing other NetView commands, such as GETMTYPE, GETMSIZE, and PARSEL2R. These commands pass information back to the REXX command list by updating the REXX local variables. These NetView commands require more processing time when invoked from REXX command lists than when invoked from NetView command lists.

**How to Compile REXX Procedures**

You can compile AON REXX programs for improved performance. The REXX programs in the RESIDENT entries of the AON control file are good choices for compiling.

**Note:** Module level tracing is not available for compiled REXX programs, although entry and exit tracing is available. You may need to reinstate the interpreted versions of the REXX programs to obtain traces for problem diagnosis.

For more information about using the REXX compiler, refer to the *IBM REXX/370 Compiler and Library User’s Guide and Reference*.

**Compiled REXX/370 Command Lists**

You can use the REXX/370 compiler to compile REXX command lists written for execution on the NetView program. Compiled REXX command lists are more efficient than interpreted REXX command lists. When compiling REXX command lists, specify the compiled EXEC (CEEXEC) output format. You can do the compilation on a separate MVS system with the REXX/370 compiler installed.

**Storage Considerations**

The size of a compiled REXX command list often exceeds the size of the source command list. The CONDENSE compiler option enables you to significantly reduce the size of the CEXEC type output. Use the CONDENSE compiler option to:

- Reduce the amount of disk space required by compiled REXX command lists.
- Reduce the amount of virtual storage required by compiled REXX command lists preloaded with the LOADCL command.
- Reduce the amount of I/O activity required to load compiled REXX command lists.

When a condensed compiled REXX command list is invoked, the command list is automatically uncondensed. A condensed compiled REXX command list requires more storage while it is running, for the following reasons:

- During the uncondense operation, an additional 128 KB of storage is required.
- While a condensed compiled REXX command list is running, both the condensed and uncondensed copy exist in storage.

Additional CPU time is required to uncondense the compiled REXX command list. Otherwise, the performance characteristics of a condensed compiled command list are the same as those of an uncondensed compiled command list.

**Performance Considerations**

The performance improvements that you can expect when you run a compiled REXX command list depend on the type of command list. A program that performs many arithmetic operations shows the greatest improvement. A program that mainly issues commands (such as NetView, VTAM, or MVS commands) shows limited improvement, because the compiler cannot decrease the time taken in processing the commands.
REXX Function Packages

REXX function packages are groups of external functions and subroutines that are packaged together. When the REXX language processor processes a function call or a call to a subroutine, the language processor searches the function packages before searching load libraries. Grouping frequently used external functions and subroutines in a function package allows faster access to the functions and subroutines, resulting in better performance.

Both the NetView program and REXX interpreter can have user, local, and system function packages. The NetView supplied/provided functions for REXX are in the NetView system function package. When a function is called, the function package search order is:

1. NetView user
2. REXX interpreter user
3. NetView local
4. REXX interpreter local
5. NetView system
6. REXX interpreter system

You can improve the performance of NetView system functions by making the NetView system function package higher in the search order so that it can be found sooner. Take the following steps to modify the function package section of the NetView sample file CNMSJM11. The labels in the following steps refer to the examples in Figure 10 on page 33, Figure 11 on page 34, and Figure 12 on page 35:

1. Move PACKTB_NAME from the bottom of the list to either replace or follow PACKTB_USER_NAME (see Figure 10 on page 33 shows the initial order), depending on if you have defined your own user function package.

2. Adjust the NetView function package table header counters.

   - If you replace PACKTB_USER_NAME with PACKTB_NAME, change PACKTB_SYSTEM_TOTAL and PACKTB_SYSTEM_USED from “1” to “0” (see Figure 11 on page 34).
   - If you have PACKTB_NAME following PACKTB_USER_NAME, change PACKTB_USER_TOTAL and PACKTB_USER_USED from “1” to “2”, and PACKTB_SYSTEM_TOTAL and PACKTB_SYSTEM_USED from “1” to “0” (see Figure 12 on page 35).

3. Assemble and link edit the sample file.

The following figures illustrate how you can change the search order of function packages. The first figure represents the sample file as it is shipped with NetView. The second and third figures show how to modify the file if you did not define a user function package, or if you did define a user function package, respectively.
PACKTB_HEADER DS 0C REXX Function Package Table HDR
PACKTB_USER_FIRST DC A(PACKTB_USER_ENTRY) Address of first USER PACKTB entry
PACKTB_USER_TOTAL DC F'1' Total number of USER PACKTB entries
PACKTB_USER_USED DC F'1' Number of used USER PACKTB entries
PACKTB_LOCAL_FIRST DC A(PACKTB_LOCAL_ENTRY) Address of first LOCAL PACKTB entry
PACKTB_LOCAL_TOTAL DC F'1' Total number of LOCAL PACKTB entries
PACKTB_LOCAL_USED DC F'1' Number of used LOCAL PACKTB entries
PACKTB_SYSTEM_FIRST DC A(PACKTB_SYSTEM) Address of first SYSTEM PACKTB entry
PACKTB_SYSTEM_TOTAL DC F'1' Total number of SYSTEM PACKTB entries
PACKTB_SYSTEM_USED DC F'1' Number of used SYSTEM PACKTB entries
PACKTB_LENGTH DC F'8' Length of each PACKTB entry
PACKTB_FFFF DC XLB'FFFFFFFFFFFFFFFF' End marker

PACKTB_USER_ENTRY DS 0C REXX USER Function Package Table Entry
PACKTB_USER_NAME DC CL8'DSIRXUFP' Name of USER Function Package
PACKTB_LOCAL_ENTRY DS 0C REXX LOCAL Function Package Table Entry
PACKTB_LOCAL_NAME DC CL8'DSIRXLFP' Name of LOCAL Function Package
PACKTB_SYSTEM_ENTRY DS 0C REXX SYSTEM Function Package Table Entry
PACKTB_NAME DC CL8'DSIRXFPG' Name of SYSTEM Function Package

Figure 10. CNMSJM11 before Modification to Switch Function Package Search Order
PACKTB_HEADER - Tivoli NetView REXX Function Package Table Header

PACKTB_USER_FIRST DC A(PACKTB_USER_ENTRY) Address of first USER PACKTB entry
PACKTB_USER_TOTAL DC F'1' Total number of USER PACKTB entries
PACKTB_USER_USED DC F'1' Number of used USER PACKTB entries
PACKTB_LOCAL_FIRST DC A(PACKTB_LOCAL_ENTRY) Address of first LOCAL PACKTB entry
PACKTB_LOCAL_TOTAL DC F'1' Total number of LOCAL PACKTB entries
PACKTB_LOCAL_USED DC F'1' Number of used LOCAL PACKTB entries
PACKTB_SYSTEM_FIRST DC A(PACKTB_SYSTEM) Address of first SYSTEM PACKTB entry
PACKTB_SYSTEM_TOTAL DC F'0' Total number of SYSTEM PACKTB entries
PACKTB_SYSTEM_USED DC F'0' Number of used SYSTEM PACKTB entries
PACKTB_LENGTH DC F'8' Length of each PACKTB entry
PACKTB_FFFF DC XL8'FFFFFFFFFFFFFFFF' End marker

PACKTB_ENTRY - Tivoli NetView REXX Function Package Table Entries

PACKTB_USER_ENTRY DS 0C REXX USER Function Package Table Entry
PACKTB_NAME DC CL8'DSIRXFPG' Name of SYSTEM Function Package
PACKTB_LOCAL_ENTRY DS 0C REXX LOCAL Function Package Table Entry
PACKTB_LOCAL_NAME DC CL8'DSIRXLFP' Name of LOCAL Function Package
PACKTB_SYSTEM DS 0C REXX SYSTEM Function Package Table Entry

Figure 11. CNMSJM11 After Modification with No User Function Package Defined
Refer to the Tivoli NetView for z/OS Customization: Using Assembler for information about adding functions to the NetView function packages.
Estimating REXX Environments

The IRXANCHR table is a Time Sharing Option Extensions (TSO/E) table used to reserve storage for REXX environments. Both the NetView program and TSO/E refer to this table when allocating storage for each REXX environment that is activated. If IRXANCHR is set to accommodate 24 REXX environments, when the 25th TSO/E REXX EXEC or NetView REXX command list that needs a REXX environment tries to start, it will not run because it cannot access a REXX environment. When this happens and you are running a NetView REXX command list, you receive message CNM416I with a return code of 20 and a reason code of 24. This error message means you have run out of REXX environments.

To ensure that your system has enough REXX environments for all processing, change the value for IRXANCHR. Because the NetView administrator is often not the same person as the TSO/E administrator, it is recommended that you maintain a separate copy of the IRXANCHR table for NetView use.

Follow these steps to estimate the number of NetView REXX environments you require in IRXANCHR:

1. Obtain a copy of the IRXANCHR table from your TSO/E administrator. This table is stored in the TSOANCH member of the SYS1.SAMPLIB data set.
2. Determine the maximum number of REXX environments that a human NetView operator might need at one time. Not all operators will use the same number of REXX environments.
3. Multiply the number in the previous step by the maximum number of human operators that will be using the NetView program at the same time.
4. Determine the total number of automated operator tasks you will use, and add that total to the number in the previous step. (This assumes that each automated task will use only one REXX environment at a time.)
5. To the previous total, increase the number to include new operators you might add later and for work shifts that may occasionally need extra operators.
6. Determine the maximum number of NetView tasks that may be active at one time and add this number to the previous total. This number represents the total number of data REXX environments.
7. Each REXX environment requires two entries in IRXANCHR. Multiply the total from the previous step by 2 and add one more to that doubled number. The result is the grand total.
8. Replace the TOTAL setting in the IRXANCHR table header in the TSOANCH member, which you obtained from the TSO/E administrator, with the grand total.
9. Change the Assembler DS statement that identifies the amount of storage needed for the total number of IRXANCHR table entries. Refer to the instructions in the TSOANCH sample for details.
10. Run the TSOANCH sample job to assemble your updated IRXANCHR load module.
11. If the load module is not link-edited as part of the TSOANCH job, link-edit your updated IRXANCHR table in a NetView private library and specify that library name in a STEPLIB DD statement in the NetView startup procedure.

In addition to adjusting the total number of REXX environments in IRXANCHR, you can tune the default number of REXX environments that the NetView program retains for each operator during an operator session. The NetView product ships with a default setting that retains three REXX environments for each operator.
(REXXENV=3). However, this number might have been adjusted on your system. To see the current setting, issue a LIST DEFAULTS command, and check the REXXENV setting.

Note: Data REXX environments are independent of the DEFAULTS and OVERRIDE command settings for REXX. Every active NetView task is assigned only one REXX environment that is dedicated to data REXX.

If an operator attempts to start four REXX command lists and each command list needs a REXX environment, the NetView program requests the environments, regardless of the REXXENV setting. However, when two or more of the operator’s REXX command lists complete, the NetView program frees only one of the four REXX environments now assigned to that operator. When REXXENV=3 and three or more REXX environments have been assigned to an operator, the NetView program keeps three of them active until the operator logs off or until a reset of the REXXENV value is processed. When the operator logs off, all REXX environments assigned to that operator are returned to the free pool where they can be assigned to another operator.

Waiting for a REXX environment to be assigned (rather than using one that is already retained) is a relatively small performance impact for human operators; however, the impact can be more significant on automated operator tasks. Automated tasks, which have the global command priority set to LOW and REXXENV is set to 1 or greater, will not have to wait for a REXX environment to be assigned.

Storage Considerations
The storage associated with each REXX environment can grow over time, depending on the needs of the REXX command lists that use the REXX environments. Because all REXX command lists do not have the same storage needs, the REXX environments eventually grow to meet the needs of the most demanding REXX command list.

- The storage associated with each REXX environment can grow over time, depending on the needs of the REXX command lists that use the REXX environments. Because all REXX command lists do not have the same storage needs, the REXX environments eventually grow to meet the needs of the most demanding REXX command list (REXXENV).
- To limit the number of REXX environments required by the NetView program, and thus reduce the amount of storage required for each task using REXX, use the NetView DEFAULTS and OVERRIDE commands (RWXXSLMT).

Before running REXX command lists in an MVS environment, determine the amount of storage required to initiate a REXX environment. TSO/E REXX, by default, gets sufficient storage for an average REXX command list, with about six levels of nested invocations. You can change the amount of storage acquired by using the DEFAULTS or OVERRIDE commands. REXX command lists that use many REXX variables or that nest more than six levels increase this storage as needed. Each REXX command list requires approximately 12K of storage to get started.

Tuning REXX Environments
Before the REXX language processor can process a REXX command list, a REXX language processor environment must exist. A REXX language processor environment is the environment in which the REXX language processor interprets
or processes the command list. This environment defines characteristics relating to how the command list is processed and how the language processor accesses system services.

When a REXX command list is executed in the NetView program, the REXX interpreter sets up a language processor environment for the NetView program. When the command list ends, this unique environment can be held for reuse by the same task. The NetView program retains these REXX environments to improve REXX environment initialization performance.

Three operands of the DEFAULTS and OVERRIDE commands affect how REXX environments are handled:

- **REXXENV** is the maximum number of REXX environments retained for a task. If you set this number to zero, the NetView program does not save any REXX environment. This adversely affects the performance of all NetView REXX command lists.

- **REXXSTOR** is the amount of storage allocated when a REXX environment is initialized. The default amount is determined by the REXX interpreter, and is sufficient for REXX command lists with up to six levels of nested invocations. If you set this number to zero, storage is acquired as needed, but performance is degraded for the first command list using a given REXX environment.

- **REXXSLMT** is the maximum amount of storage that a REXX environment is allowed to accumulate before being ended, after its current use is completed. Storage associated with a REXX environment can increase, depending on the needs of the REXX command lists that have been interpreted under the environment. Because each REXX command list can have different storage needs, REXX environments can grow to meet the storage needs of the most demanding REXX command list. If the storage for an environment grows to exceed the amount of storage specified by REXXSLMT, the environment is freed when the REXX command list finishes executing.

Note: The values for REXXENV, REXXSLMT, and REXXSTOR do not apply to Data REXX environments.

REXXSTOR is the lower limit of storage in each environment. REXXSLMT is an upper threshold to prevent environments that grow very large from being allocated and not freed until the owning task terminates.

The following are tuning recommendations for the REXXENV, REXXSTOR and REXXSLMT operands of the DEFAULTS and OVERRIDE commands:

- For most situations, a small REXXENV value (1 or 2) is sufficient, because most operators will have only one or two REXX command lists active at once.

- Consider setting REXXSTOR to zero, because it affects only the first REXX command list interpreted under a REXX environment. A value of zero minimizes the potential for storage waste.

- Set a value for REXXSLMT so that REXX environment storage does not grow indefinitely.

- If you want to minimize storage at the expense of extra processing time, set both REXXENV and REXXSTOR to zero. With these settings, storage for REXX environments is allocated only when needed, and is freed as soon as possible.

- Use the TASKUTIL command to monitor the amount of queued storage allocated for different tasks. REXX environments are allocated as queued storage under operator station tasks (OSTs) or autotasks. The environments are part of the storage total for the task shown in output from the TASKUTIL command.
you notice a significant storage growth for a task that is executing many REXX command lists, consider adjusting the values for REXXENV, REXXSTOR, and REXXSLMT.

Notes:
1. An entry (2 slots) in the the REXX anchor table, IRXANCHR, is required for each non-nested REXX command list to run.
2. If a REXX command list is invoked from another REXX command list, a new environment is not required. The nested command list uses the environment of the primary command list.

Refer to Tivoli NetView for z/OS Command Reference for information about the DEFAULTS and OVERRIDE commands. Refer to the Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language for information about nesting REXX command lists from the assembler, PL/I, or C languages.

Command Processors

Each command processor must have a CMDMDL statement in the DSICMD initialization member. Do not specify RES=N on the CMDMDL statements for frequently used command processors. If you do not specify RES=N, the modules are made resident, eliminating the I/O needed to load them every time they are invoked.

You can use the AUTOCNT command to identify command procedures that are executed frequently from the automation table. See “The AUTOCNT Command” on page 11 for more information.

If you are using the generic automation receiver (NVAUTO), modify the CMDMDL statement for DSINVGRP to make the command processor resident.

Command Processors Written in a High-Level Language

Command processors that are written in an HLL are compiled rather than interpreted. In terms of performance, compiled command procedures are usually processed much faster than interpreted command procedures. If you are using PL/I programs, consider running them in a preinitialized environment. See “Running PL/I Programs in a Preinitialized Environment” on page 40 for more information.

You can improve initialization overhead for command processors that do not use preinitialized environments by avoiding I/O in searching for and fetching the HLL run-time libraries. Consider the following recommendations:

- If your NetView startup job control language (JCL) has a STEPLIB DD statement, eliminate it as discussed in “STEPLIB DD Statements” on page 124. This eliminates the I/O needed to search the STEPLIB for each LOAD, LINK, or XCTL system macro executed. If a STEPLIB DD statement exists, I/O is required to search for the HLL run-time libraries.

- If you are using command processors written in PL/I or C, you can preload the PL/I and C run-time libraries by coding the following CMDMDL statements in DSICMD:
  
  IBMBLIIA CMDMDL MOD=IBMBLIIA,TYPE=R,RES=Y
  EDCXV CMDMDL MOD=EDCXV,TYPE=R,RES=Y
  EDCX24 CMDMDL MOD=EDCX24,TYPE=R,RES=Y
Preloading the HLL run-time libraries eliminates I/O for the directory search and the fetch for run-time libraries.

See the appropriate MVS publication for more information about improving production library performance.

Set the ISA and HEAP sizes carefully for your HLL command processors. For best performance, the initial stack allocation should be large enough to satisfy all requests for stack storage. Likewise, the initial heap segment should be large enough to satisfy all requests for heap storage. Use the facilities provided by the HLL for tuning the ISA and HEAP sizes.

**Running PL/I Programs in a Preinitialized Environment**

Before a command processor written in a high-level language (HLL) can be executed, a run-time environment (or execution environment) must exist. A run-time environment is a set of resources that are used to support the execution of a program. Preinitialization allows the run-time environment to be initialized once and used for multiple program executions. Without preinitialization, the run-time environment is created and terminated for each program execution.

You can use the HLLENV command to define the number of preinitialized environments that are allocated by the NetView program. The HLLENV command enables you to specify two pools of preinitialized environments using the REGENVS and CRITENVS keywords.

- Regular environments (REGENVS) are allocated when the HLLENV command is issued, and are available to any PL/I command processor that is capable of using preinitialized environments.

**Note:** If you want a PL/I command processor to use a preinitialized environment, you must link the command processor with DSIEXPLP instead of DSIEXPLI.

- Critical environments (CRITENVS) are available only to PL/I command processors capable of using preinitialized environments that also have bit 4 in HLLOPTS set to 1. Critical environments are not allocated when the HLLENV command is issued, but are allocated on an as-needed basis.

Allocation of environments from the two pools works as follows:

- For command processors that are capable of using preinitialized environments but that do not have bit 4 set in HLLOPTS:
  - If an environment is available from the regular pool, it is used.
  - Otherwise, a new environment is created for this command processor, and the environment is terminated when the command processor completes execution.

- For command processors that are capable of using preinitialized environments and that have bit 4 set in HLLOPTS (critical command processors):
  - If an environment is available from the regular pool, it is used.
  - If an environment is not available from the regular pool but is available from the critical pool, it is used.
  - Otherwise, a new environment is created for this command processor, and the environment is terminated when the command processor completes execution. If the number of environments in the critical pool is less than the CRITENVS value specified with the HLLENV command, a new environment
is created and added to the critical pool. This makes it more likely that a 
preinitialized environment will be available the next time a critical command 
processor is executed.

For more information on preinitialized environments and the HLLENV command, 
refer to the \[Tivoli NetView for z/OS Customization: Using PL/I and C\]

**Suggestions for Using Preinitialized Environments**

- Review your PL/I command processors and decide which ones are good 
candidates for using preinitialized environments. Good candidates include 
installation exits and frequently used command processors, such as those issued 
from the NetView automation table. Some restrictions apply to command 
processors using preinitialized environments. Refer to the \[Tivoli NetView for z/OS Customization: Using PL/I and C\] for more details.

- Modify CNMSTYLE to issue the HLLENV command to set up the environment 
pools.

- Use the HLLENV LIST STATS command to monitor usage of the regular and 
critical environment pools. Figure 13 contains sample output.

```
BNH040I IBMHLPLI PREINITIALIZED ENVIRONMENT STATISTICS
BNH041I STATISTICS RESET AT: 03/30/00 16:44:04
BNH042I PISA: 4096 PHEAP: 4096 DEFAULT: NOTPREINIT
BNH043I NUMBER REQUESTED. REGENVS: 4 CRITENVS: 2
BNH044I ALLOCATED. REGENVS: 4 CRITENVS: 1
BNH045I IN USE. REGENVS: 4 CRITENVS: 1
BNH046I MOST NEEDED. REGENVS: 4 CRITENVS: 1
BNH047I TIMES USED. REGENVS: 4 CRITENVS: 1
BNH048I TIMES UNAVAILABLE. REGENVS: 0 CRITENVS: 1
BNH049I AVERAGE NEEDED. REGENVS: 2.50 CRITENVS: 1.00
BNH053I IBMHLPLI ENVIRONMENT STATISTICS RESET
```

**Figure 13. Sample Output of the HLLENV LIST,STATS,RESET Command**

The goal in tuning the regular and critical pool allocations is to minimize the 
number of times that preinitialized environments are not available when they 
are needed.

- Use of preinitialized environments is a processing versus storage tradeoff. For 
the additional storage requirement of preinitialized environments, you have a 
reduction in CPU processing. Monitor NetView storage usage with the 
RESOURCE command. Some environment storage (stack storage) is allocated 
below the 16 MB line. Therefore, even if you do not have a 31-bit storage 
constraint, you should monitor 24-bit storage usage.

- The storage associated with each preinitialized environment can grow over time, 
depending on the needs of the PL/I command processors using them. Because 
the storage requirements for different PL/I command processors vary, all the 
preinitialized environments eventually grow over time to meet the needs of the 
most demanding PL/I command processor. To limit this growth, periodically 
change the number of allocated environments to zero with the HLLENV 
command, allowing them to be freed. Then reset the requested number of 
environments with the HLLENV command to have them reallocated.

For more information on preinitialized environments and the HLLENV command, 
refer to the \[Tivoli NetView for z/OS Customization: Using PL/I and C\]
Global Variables

The following are classes of NetView global variables:

- Task global variables are accessible to any command procedure running under the task, as well as from the automation table using the DSITGLOB automation table function (ATF). Task global variables can be set to null, but once the variables are created, some storage is allocated for the variables until the task ends.

- Common global variables are accessible by any command procedure running under the NetView program from the automation table using the DSICGLOB ATF. Common global variables can be set to null, but once the variables are created, some storage is allocated for the variables until the NetView program ends. Common global variables can be updated directly from any task. Access to the common global variables is serialized to provide data integrity using the system enqueue and dequeue facility.

Accessing task global variables is faster than accessing common global variables, because task global variables do not require overhead of the system enqueue and dequeue facility. Keep this in mind when deciding whether to use task or common global variables for an application.

Enhancing Performance

Specify the number of task and common global variables you expect to use in the NetView constants module DSICTMOD. Increasing the expected number of variables improves the access time for the variables by optimizing the control block search algorithm. If you specify a larger number, more storage is required. For common global variables, the amount of additional storage should not exceed 64 K, even if you expect more than 100000 variables.

For task global variables, set the number of expected variables carefully. Additional storage is allocated for every task that uses task global variables. For example, an expected number of 1000 variables would require about 2.5 K of additional storage for each task using task global variables. If you expect 5000 task global variables, about 6 K of additional storage is required per task.

For more information about setting values in DSICTMOD, refer to the Tivoli NetView for z/OS Installation: Configuring Additional Components. You can use the QRYGLOBL command to determine the actual number of task or common global variables. Use this information to help you determine which values should be specified in the NetView constants module. This will improve system performance related to global variable retrieval. Refer to the Tivoli NetView for z/OS Command Reference for the QRYGLOBL command syntax.

If you are writing an assembler command processor that updates multiple global variables, consider using the NUMVARS option of the DSIVARS macro. This updates multiple global variables with a single macro invocation.

Save/Restore Processing

In using the Save/Restore function for global variables (GLOBALV SAVEC, SAVET, RESTT, and RESTC), keep in mind that a separate VSAM I/O is required for saving each variable, regardless of whether you specify groups of variables using a wildcard character (*). The Save/Restore VSAM data set uses local shared resources (LSR) by default. Restore operations can be buffered together, resulting in
reduced I/O. Save operations use VSAM writes, which are not deferred with LSR. If you consider using the deferred write (DFR) performance option, see Chapter 9, "Tuning for VSAM" on page 97 for a discussion of the potential risk of losing data contained in the buffers in the event of an abnormal end of the NetView program.

Do not save every global variable each time it is used in a command procedure. Save only the most critical global variables in that manner. The processing required for GLOBALV SAVE and RESTORE is an order of magnitude greater than the processing for GLOBALV GET and PUT.
Chapter 5. Tuning for the Hardware Monitor

The hardware monitor is the component responsible for managing host and network problem information. The hardware monitor manages this information using a filtering mechanism that controls data recording and the generation of alerts to the operator.

Tuning Techniques

Following are the major tuning techniques for the hardware monitor, arranged in order of expected effect on performance, with the most important tuning considerations listed first. These recommendations are described in detail in this chapter.

1. Use the ESREC and AREC filters to control what data is logged to the hardware monitor database as events, statistics, and alerts. For records that are automated using the NetView automation table, consider blocking the recording filters from the automation table. See "Hardware Monitor Filters".

2. Use the ALCACHE statement to specify the number of alerts to be kept in storage. See "Using the ALCACHE Statement in CNMSTYLE" on page 48.

3. For environments with more than 10 alerts a minute, use the viewing filter to minimize the number of Alerts-Dynamic panels that are updated for each alert. See "Alerts-Dynamic Panel" on page 47.

4. Reorganize the VSAM database frequently to avoid control interval (CI) and control area (CA) splits. Use the LISTCAT command to determine whether splits are occurring. You can use this information to reorganize the database if necessary. See "VSAM Database Maintenance" on page 105 and "LISTCAT Command" on page 101.

5. Use the HMSTATS command to monitor the amount of hardware monitor work being done on your system. HMSTATS displays event and alert workload counters, and statistics about the alert cache and alerts dynamic (ALD) screen update processing. See "HMSTATS Command" on page 50.

6. For environments in which alerts are forwarded to a focal point host, use the LU 6.2 method of forwarding alerts where possible, rather than LUC alert forwarding or the alert notification forwarding mechanism that is based on message BNJ146I. See "Using the NPDA.ALERTFWD Statement" on page 49.

7. Use the RATE statement to stop database logging of rapidly recurring events for a resource. See "RATE Statement Initialization Specifications" on page 56.

8. Consider coding MACRF=DFR in CNMSTYLE instead of using the LSR option. See "Local Shared Resources (LSR) and Deferred Write (DFR)" on page 97.

Refer to "Automating Hardware Monitor Records" on page 16 and "Filtering Hardware Monitor Records" on page 16 for more information about alert automation.

Hardware Monitor Filters

The hardware monitor uses several filters in processing statistical records and events from the network. Use the SRFILTER command to set the following filters:
Event and statistical recording (ESREC) filter
Determines which event and statistical records are recorded in the database. Unimportant events can be filtered using ESREC (example: NPDA SRF ESREC BLOCK xxxx).

Alert recording (AREC) filter
Determines which event records are also recorded as alerts.

Operator (OPER) filter
Determines which alerts produce messages BNJ030I and BNJ146I for the authorized receiver.

Route (ROUTE) filter
Determines whether alerts are to be forwarded to the alert focal point (provided an alert focal point exists).

Color (COLOR) filter
Determines the color in which an alert is displayed when the alert is presented on the Alerts-Dynamic, Alerts-Static, or Alerts-History panels.

TECROUTE filter
Sets a filter for converting alerts to Tivoli Enterprise Console (TEC) events and forwarding the events to the Enterprise console. An alert must pass the ESREC and AREC filters before the TECROUTE filter is applied to the alert.

Use the SVFILTER command to determine which alerts can be viewed by a particular operator, as well as controlling viewing of the Total Events and Total Statistics panels.

Refer to the Tivoli NetView for z/OS Command Reference for syntax and use of the SRFILTER and SVFILTER commands.

Effective use of filters can have a significant effect on the hardware monitor resource requirements. You can use many criteria to filter data. Your environment determines which are appropriate.

You can automate records using the MSUSEG and HIER conditions in the NetView automation table. If you do not need to record automated records to the hardware monitor VSAM database, you can save processing time by blocking the recording filters (ESREC, AREC, OPER, and ROUTE) in the automation table. Refer to “Automating Hardware Monitor Records” on page 16 and “Filtering Hardware Monitor Records” on page 16 for more information about automating problem records and blocking recording filters.

Figure 14 on page 47 represents the hardware monitor filter structure.
For each alert that it processes, the hardware monitor checks to see if operators are on the Alerts-Dynamic panel. For each operator on the Alerts-Dynamic panel, the hardware monitor checks the operator’s viewing filter to determine if the operator’s panel should be updated. The processing required for Alerts-Dynamic panel updates can be significant if several operators are on the Alerts-Dynamic panel, and each must be updated with every alert. Minimize the average number of Alerts-Dynamic panels that are updated for each alert using the viewing filter. If your installation has more than 10 alerts per minute or a large number of Alerts-Dynamic panels active, use the viewing filter aggressively.

The following are recommendations to reduce the amount of processing required for Alerts-Dynamic panel updates:

- Before you leave your workstation, exit the Alerts-Dynamic panel. It is not a productive use of system resources to update an unattended panel.
- If you are on the Alerts-Dynamic panel and you experience a burst of panel updates (a rapidly rolling screen). By re-entering the ALD command to go back to the Alerts-Dynamic panel, the screen will be updated with only the most recent alerts.

The HMSTATS command displays statistics about ALD screen update processing on your system, including the average number of screen updates per alert (UPDATES/ALERT) and the total view filtering percentage (% FILTERED). Refer to "HMSTATS Command" on page 50 for more information.

**Figure 14. Hardware Monitor Database and Filters**

**Alerts-Dynamic Panel**
Using the ALCACHE Statement in CNMSTYLE

You can use the ALCACHE statement to improve hardware monitor performance if your NetView operators do both of these:

• Receive many alerts (either at steady rates or in bursts)
• Frequently monitor the Alerts-Dynamic panel

The ALCACHE statement is part of the CNMSTYLE initialization member and enables you to specify the number of alerts kept in storage. By keeping alert records in storage, all VSAM I/O to retrieve those alerts from the Hardware Monitor database is eliminated. This allows the Alerts Dynamic Panel to be updated without the additional overhead of the VSAM I/O. Also, the Hardware Monitor main task, BNJDSERV, uses less processor time when alerts are kept in storage through ALCACHE. In general, the more storage you allocate for alerts, the better the hardware monitor’s performance.

In Tivoli NetView for OS/390 V1R4 and later, the maximum hardware monitor alert wrap count was increased from 999 to 9999. If ALCACHE is coded to WRAPCNT in CNMSTYLE, storage usage for the Hardware Monitor can increase by more than 10 times that of previous releases.

Table 5 shows examples of how the ALCACHE statement can affect the performance of your hardware monitor. Use the information to determine the appropriate ALCACHE statement for your system. This table assumes that one alert requires 500 bytes of storage.

Table 5. Determining the Best ALCACHE Statement for Your System

<table>
<thead>
<tr>
<th>System Usage Characteristic</th>
<th>ALCACHE Statement</th>
<th>Storage Allocated</th>
<th>Performance Benefit</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware monitor performance is important because your NetView operators frequently monitor the Alerts-Dynamic panel.</td>
<td>ALCACHE WRAPCNT</td>
<td>Storage allocated depends on the alert wrap count. You can specify the alert wrap count with the W(wrap) statement in CNMSTYLE or with the SWRAP command. For example, if you specify W AL 50, then 25000 bytes of storage (500 bytes x 50 alerts) are allocated for alerts.</td>
<td>Performance benefit for monitoring the Alerts-Dynamic panel for both bursts of alerts and a steady stream.</td>
<td>The storage allocated depends on the amount of alert-cache storage acceptable to your system.</td>
</tr>
<tr>
<td>NetView operators rarely monitor the Alerts-Dynamic panel.</td>
<td>ALCACHE NONE</td>
<td>None</td>
<td>None</td>
<td>If the Alerts-Dynamic panel is rarely used, putting alerts in storage does not help your performance. Therefore, do not use storage for alerts.</td>
</tr>
<tr>
<td>Your system receives an average of 10 alerts per minute and bursts of up to 60 alerts at once.</td>
<td>ALCACHE 10</td>
<td>500 bytes x 10 alerts = 5000 bytes of storage</td>
<td>Performance benefit during steady states.</td>
<td>Although you can save on storage and performance during steady states, performance during bursts is not as good as it would be if you defined a larger alert-cache size.</td>
</tr>
</tbody>
</table>
Table 5. Determining the Best ALCACHE Statement for Your System (continued)

<table>
<thead>
<tr>
<th>System Usage Characteristic</th>
<th>ALCACHE Statement</th>
<th>Storage Allocated</th>
<th>Performance Benefit</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Your system receives an average of 10 alerts per minute and bursts of up to 60 alerts at once.</td>
<td>ALCACHE 60</td>
<td>500 bytes x 60 alerts = 30000 bytes of storage</td>
<td>Performance benefit during both alert bursts and steady states.</td>
<td>Performance is improved during bursts, but more storage is used.</td>
</tr>
<tr>
<td>System utilization percentage is low.</td>
<td>ALCACHE NONE</td>
<td>None</td>
<td>None</td>
<td>The performance gained by keeping alerts in storage is minimal because system utilization is low. Therefore, do not use storage for alerts.</td>
</tr>
<tr>
<td>Your system can spare only a small amount of storage for alerts.</td>
<td>ALCACHE 10</td>
<td>500 bytes x 10 alerts = 5000 bytes of storage</td>
<td>Performance benefit during steady states.</td>
<td>The storage allocated depends on the amount of alert-cache storage acceptable to your system.</td>
</tr>
</tbody>
</table>

The HMSTATS command displays the current ALCACHE setting and statistics on the alert cache usage. A good ALCACHE setting will have a “% SATISFIED” value near 100% without wasting storage. Refer to [HMSTATS Command on page 50](#) for more information.

**Using the ALERTLOG Statement in CNMSTYLE**

The ALERTLOG statement enables you to make a VSAM performance versus processor time decision in regards to reading/writing records to the Hardware Monitor VSAM data base. With RSTDRANG, alert records are logged in one key range in the data base, and non-alert records are logged in another key range in the data base. With RANDRANG, the default, alerts are logged throughout the data base, they are not restricted to a single key range. The ALERTLOG statement is processed only for new (empty) data bases, and it is ignored for existing data bases. Performance is quite comparable for RSTDRANG and RANDRANG in both small databases and/or until the wrap count is exceeded. Once the wrap count is exceeded or in databases with more than 10,000 alerts being held, Hardware Monitor initialization time will take up to four times longer if RSTDRANG is coded.

**Note:** Hardware Monitor does not support key-ranged databases. Unpredictable results can occur.

**Using the NPDA.ALERTFWD Statement**

The following methods provide alert forwarding:

- Using an LU 6.2 session.
- Using an LUC session.
- Does not support intermediate node (nested) focal points.
- Converting the alert to a message (BNJ146I) and then forwarding the message over a NetView-NetView task (NNT) to the focal point. The message is then converted back to an alert.
The NPDA.ALERTFWD statement in CNMSTYLE controls which alert forwarding method is used. Forwarding over LU 6.2 sessions is extremely beneficial for hardware monitor performance for networks that incorporate forwarding alerts across an intermediate focal point.

**Note:** In releases prior to NetView V1R1, the only option for forwarding alerts across an intermediate focal point is to convert the message BNJ146I.

CPU usage is less than half when implementing LU 6.2 intermediate alert forwarding instead of converting the message BNJ146I. This CPU reduction is recognized both in steady state and bursts of alerts. When an intermediate focal point is not used, LU 6.2 alert forwarding provides comparable performance to LUC forwarding.

Both the LU 6.2 and the LUC methods reduce VSAM recording at the host and have other usability advantages. The LU 6.2 alert forwarding method is the recommended method. For additional information refer to the *Tivoli NetView for z/OS Installation: Configuring Additional Components* and the *Tivoli NetView for z/OS Automation Guide*.

**HMSTATS Command**

HMSTATS is an unsupported, internal serviceability tool that is helpful in tuning the Hardware Monitor. Use the HMSTATS command to monitor the amount of hardware monitor work being done on your system. HMSTATS displays event and alert workload counters, and statistics about the alert cache and alerts dynamic (ALD) screen update processing. Figure 15 on page 51 shows a sample of output generated from the HMSTATS command.
The HMSTATS command also has a RESET option, which sets all of the workload counters to zero.

The following information is displayed with the HMSTATS command:

**RECEIVED COUNTS:**

**TOTAL TRAFFIC:** 289792

**FROM EP:** 0

**RECORDED:** 274545

**RECORDED (GMFALERT):** 0

**FILTER COUNTS:**

**TOTAL LVL1% LVL2% LVL3% LVL4% LVL5%**

**EVENTS/STATS:** 277528 0 1 78 18 0

**ALERTS:** 65000 % NON-GENERIC: 21%

**OPER:** 0

**ROUTE:** 0

**MISCELLANEOUS**

**EXTERNAL LOG:** 274529 **RATE VALUE:** 12

**CORRELATORS:** 94238 **PURGE IN PROGRESS:** NO

**ALERT CACHE**

**ALCACHE: WRAPCNT:** 112

**TABLE ENTRIES:** 112 **TABLE STORAGE:** 46K

**REQUESTS:** 842908 **RETRIEVALS:** 425701

**% SATISFIED:** 100%

**ALD COUNTS**

**NUMBER OF ALDS:** 8 **ALDS BEHIND:** 1

**TOTAL UPDATES:** 11334 **TOTAL FILTERED:** 414421

**UPDATES/ALERT:** 0.2 % FILTERED: 97%

<table>
<thead>
<tr>
<th>OPERID</th>
<th>DOMAIN</th>
<th>BEHIND</th>
<th>UPDATES</th>
<th>FILTERED</th>
<th>FILTER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOICE</td>
<td>CNM12</td>
<td>67</td>
<td>64324</td>
<td>99%</td>
<td></td>
</tr>
<tr>
<td>KIOSK</td>
<td>CNM12</td>
<td>365</td>
<td>47505</td>
<td>99%</td>
<td></td>
</tr>
<tr>
<td>BIGS6S</td>
<td>CNM12</td>
<td>0</td>
<td>14894</td>
<td>94%</td>
<td></td>
</tr>
<tr>
<td>OPER20</td>
<td>CNM12</td>
<td>0</td>
<td>119</td>
<td>96%</td>
<td></td>
</tr>
<tr>
<td>OPER02</td>
<td>CNM12</td>
<td>0</td>
<td>201</td>
<td>93%</td>
<td></td>
</tr>
<tr>
<td>OPER12</td>
<td>CNM12</td>
<td>0</td>
<td>38</td>
<td>98%</td>
<td></td>
</tr>
<tr>
<td>DAVE</td>
<td>CNM12</td>
<td>1252</td>
<td>65</td>
<td>1544</td>
<td>96%</td>
</tr>
<tr>
<td>ABRAHAM</td>
<td>CNM12</td>
<td>0</td>
<td>543</td>
<td>1085</td>
<td>66%</td>
</tr>
</tbody>
</table>

**END OF HARDWARE MONITOR STATISTICS DISPLAY**

Figure 15. Example of Output from HMSTATS Command

The HMSTATS command also has a RESET option, which sets all of the workload counters to zero.

The following information is displayed with the HMSTATS command:

**RECEIVED COUNTS:**

**TOTAL TRAFFIC**

The total number of items that have been received by the hardware monitor.

**FROM EP**

The total number of alerts received that were forwarded from NetView entry points.

**RECORDED**

The total number of items that have been recorded to the hardware monitor database.

**RECORDED (GMFALERT)**

The total number of items that have been recorded for GMFHS.

**FILTER COUNTS:**
EVENTS/STATS
The total number of events and statistics which have been recorded to the hardware monitor database.

Level 1-5 %
The percentage of the events/statistics records that were written at the given level in the resource hierarchy.

ALERTS
The total number of alerts which have been recorded to the hardware monitor database.

% NON GENERIC
The percentage of the alerts recorded which are not in generic format.

OPER The total number of alerts which passed the OPER filter to generate messages BNJ030I and BNJ146I.

ROUTE
The total number of alerts which passed the ROUTE filter and will be forwarded to the alert focal point.

MISCELLANEOUS:
EXTERNAL LOG
The total number of external log records that have been written.

CORRELATORS
The total number of correlators received.

RATE VALUE
The value used for the RATE statement (in seconds) in CNMSTYLE.

PURGE IN PROGRESS
Indicates whether a database purge is in progress.

ALERT CACHE:
ALCACHE
The value coded on the ALCACHE parameter in CNMSTYLE (or the default), which is either NONE, WRAPCNT, or a numeric value.

ALERT WRAPCNT
The current alert wrap count value.

TABLE ENTRIES
The number of entries in the alert cache table.

TABLE STORAGE
The number of kilobytes of storage allocated for the alert cache table.

REQUESTS
The number of attempts made to retrieve an alert from the alert cache.

RETRIEVALS
The number of successful attempts to retrieve an alert from the alert cache.

% SATISFIED
The percentage of alert cache requests that were satisfied. A request can be satisfied by either retrieving the alert, or by determining that no next alert exists.

ALD COUNTS:
NUMBER OF ALDS
The number of operators that have an alerts dynamic (ALD) display active.

ALDS BEHIND
The number of ALD screens that are behind. An ALD screen is considered behind if there are alerts recorded to the database that have not yet been checked against the operator’s VIEW filter.

TOTAL UPDATES
The total number of times an ALD screen has been updated with an alert. This is a cumulative count for all ALD screens, including those that are no longer active.

TOTAL FILTERED
The total number of times an alert was blocked by an operator’s VIEW filter. This is a cumulative count for all ALD screens, including those that are no longer active.

UPDATES/ALERT
The average number of ALD screens that are updated for each alert. This number is calculated by dividing the total number of times an ALD screen has been updated with an alert (TOTAL UPDATES) by the total number of alerts that have been recorded to the database (the ALERTS field in the FILTER COUNT section). This result is intended to be a long running average, and does not account for alerts that have been recorded to the database but not yet checked against operator VIEW filters.

% FILTERED
The average VIEW filtering percentage. This number is calculated by dividing the TOTAL FILTERED field by the sum of the TOTAL UPDATES and TOTAL FILTERED fields.

For each operator that has an ALD screen active:
OPERID
The operator’s ID.

DOMAIN
The domain the operator is in.

BEHIND
The number of alerts which have been recorded to the database, but have not yet been checked against the operator’s VIEW filter.

UPDATES
The number of alerts that have passed the operator’s VIEW filter and been sent to the operator’s ALD screen.

FILTERED
The number of alerts that have been blocked by the operator’s VIEW filter.

FILTER %
The operator’s VIEW filtering percentage. This number is calculated by dividing the FILTERED field by the sum of the UPDATES and FILTERED fields.
Using the DSRBO Parameter in CNMSTYLE

For the hardware monitor, the solicited data services request block (DSRBO) value specifies the projected number of concurrent user requests for services from the BNJDSERV data services task. To determine the number of DSRBOs that BNJDSERV needs on this host, consider the number of cross-domain conversations in which this host can be involved at any given time, and the number of operators performing distributed database retrieval from this host.

The default value is 5. You can use the DSRBS command specifying the BNJDSERV data services task to display the current use of the hardware monitor DSRBs. Refer to “Data Services Request Blocks (DSRBs)” on page 110 for more information about tuning the DSRB allocation for a data services task and for more detail about the DSRBS command.

Wrap Counts

Events, statistical data, and alerts are all logged to the hardware monitor database. The defaults for the number of event and statistical records that are logged are 25 records for each resource. After 25 records, the data is wrapped. For alerts, the number of records logged before wrapping is 100. You can decrease the default values to make the hardware monitor database smaller. Use the SWRAP command to change the wrap count of the records.

SWRAP Command

The SWRAP (SW) command establishes the number of event or statistical records to be retained for a specified resource or the total number of alert records to be retained on the hardware monitor database. You can issue the command only for resources against which data has been logged on the hardware monitor database.

When this command requests a reduction in wrap count, the oldest records are deleted immediately. If the wrap count is small, it may appear that the oldest record is not being wrapped off, because the new set of records fits on the panel without deleting the old records.

For more information about the SWRAP command, refer to Tivoli NetView for z/OS Command Reference.

Note: The SWRAP command can result in loss of error data.

Example: Using the SWRAP Command

SW EV 5 N UNIT1

This SWRAP command sets the event wrap count to 5 for resource UNIT1.

Initialization Specifications

Use the WRAP (W) statement in CNMSTYLE to set wrap counts at initialization for resources not yet in the database. The W statement does not alter the wrap count for resources already existing on the hardware monitor database. The W statement assigns initial wrap values when the first error record for a particular resource is received.

The following WRAP command sets the wrap count to 10 for event records for a line:
Note: An alert wrap count that is too low can trigger the RATE statement.

For more information about the WRAP (W) and RATE statements, refer to the Tivoli NetView for z/OS Administration Reference.

Error-to-Traffic (E/T) Ratio Thresholds

The hardware monitor calculates ratios of temporary errors to traffic. The hardware monitor determines this ratio for statistical data that it receives regarding communication lines attached to a 37x5, 3710, or 3708 and other channel-attached controllers. This ratio provides a warning when a line might be experiencing many temporary errors. The hardware monitor provides this warning by issuing an event with the description as the following example shows:

\[ \text{ERROR TO TRAFFIC RATIO EXCEEDED} \]

The hardware monitor generates an event notification whenever it receives statistical data in which the ratio of temporary errors to traffic is greater than the threshold values.

The error-to-traffic (E/T) threshold defaults to a value of 3% for communication lines and to 1% for all channel-attached communication controllers.

The default threshold of 3% is not appropriate for all lines. Lines vary widely in the ratio of temporary errors to traffic. Some lines may have a normal range of 3% to 11%. For other lines, the threshold should be at least 15% so that an event is generated for an abnormal condition only. Not only do meaningless E/T ratio events clutter up the hardware monitor events panels, they also degrade performance. More data is sent to the hardware monitor database when an event notification is generated from statistical data.

You can change the E/T thresholds in the hardware monitor by issuing the SRATIO operator command and specifying the thresholds in the hardware monitor initialization program.

SRFILTER and SRATIO Commands

The SRATIO operator command changes the threshold for any resource that has existing statistical data on the hardware monitor database. Changes to the thresholds are not maintained if the database is reinitialized. You may want to write a command procedure containing SRATIO commands for all lines where the default threshold is not appropriate. You can then execute this command procedure when you reinitialize the hardware monitor database.

When you specify a line name in the command, all controllers on that line use the new threshold you supply. The SRATIO command in the following example changes the E/T threshold for a line named LINE1234 to 15.0%.

\[ \text{NPDA SRATIO 150 N LINE1234} \]

The value 150 means 15.0%.

Use the following SRATIO command to prevent alerts from being generated based on statistical data for a given line. The SRATIO command in the following example prevents alerts from being generated for LINE1234.

\[ \text{NPDA SRATIO OFF N LINE1234} \]
Use the following hardware monitor SRFILTER (SRF) commands to disable thresholds for all resources in your network, for example, to display no warning facility for statistical data.

```
NPDA SRF ESREC BLOCK C 044E0
NPDA SRF ESREC BLOCK C FEE40
NPDA SRF ESREC BLOCK C FEE64
NPDA SRF ESREC BLOCK C FEF45
NPDA SRF ESREC BLOCK C FFD45
NPDA SRF ESREC BLOCK C FFD46
```

The SRFILTER commands in this example prevent the logging of events and alerts based on exceeding E/T ratios. The statistical data is still logged.

As with all set recording filter (SRFILTER) commands, the filter settings are lost when you shut down the NetView program. Use a command procedure to execute your SRFILTER commands each time the NetView program is activated, or code your initial automation table to set your filters when the hardware monitor is initialized.

If you place SRFILTER commands in a NetView initialization command list, you need to add a short timer delay, perhaps 1 or 2 minutes, to ensure that the hardware monitor initialization has been completed before the command list is executed. Refer to the sample NetView automation table DSIDNMAT (CNMS7003) for information on how to invoke filters during hardware monitor initialization.

### RATIO Statement Initialization Specifications

You can also specify thresholds for the hardware monitor at initialization using the RATIO (R) statement in CNMSTYLE. These thresholds do not change for resources that already have statistical data logged on the hardware monitor database. Therefore, for resources already on the database, the new thresholds do not take effect until the next time you reinitialize the database. If you periodically reinitialize the database, specify E/T thresholds in the R statement.

To change thresholds for the hardware monitor, a ratio statement in CNMSTYLE is required for each line. The RATIO command in the following example changes the threshold for a line named LINE1234 to 15.0%.

```
R LINE LINE1234 150
```

The value 150 means 15.0%. The resource type (LINE) is required in a RATIO statement.

### RATE Statement Initialization Specifications

You can also specify the maximum rate at which events can be logged to the Hardware Monitor database. The purpose of this function is to stop database logging of repetitive events for a resource, such as alert flooding or alert streaming. The RATE function compares the time between an event being wrapped off the database and the new event record being recorded on the database. A filter is set to block the recording of events from the resource if the difference is less than the time specified on the RATE statement. The RATE statement filters by resource name. When the filter is set, the message BNJ045I is issued. Determine why the filter was set and correct the cause of the stream of alerts. The filter entry can then be deleted.

The RATE statement is specified in member CNMSTYLE. The recommended RATE is one event per second for the predominant wrap count. This works out to 25
seconds for the default wrap count of 25. If a RATE statement is not coded in CNMSTYLE, the RATE value is set to zero. A zero value turns off the RATE function. Refer to the [Tivoli NetView for z/OS Administration Reference](http://link) for more information about coding these statements.

**Note:** An event that is blocked by a recording filter set by the RATE statement is not sent to automation. If automation is required for these alerts, code an AUTORATE statement in CNMSTYLE.

---

### Event/Automation Service

It is recommended that the connection mode in the alert adapter configuration file member IHSAAACFG and in the message adapter configuration file member IHSAMCFG be set to `connection_oriented` for the event delivery method. The default value is `connection_less` which specifies that a new connection should be established (and destroyed) for each event that is sent. `Connection_oriented` specifies that a connection should be established when the adapter is initialized and maintained for all events sent. A new connection is established only if the connection is lost. The connection is destroyed when the adapter is terminated.

This same change should be made on the Tivoli Enterprise Console server desktop. When creating your TME® desktop rule base files, change `tec_forward.conf` in TEC_RULES to `connection_oriented`.

It is also recommended that the trace_level in the Event/Automation Service initialization file member IHSAINIT be set to OFF.
Chapter 6. Tuning for the Session Monitor

The session monitor component of the NetView program collects information about Systems Network Architecture (SNA) sessions. The base data about a session is called session awareness (SAW) data. See “SAW Data” on page 60 for more information. Other types of data that can be collected about sessions include:

- Accounting data
- Availability data
- Session path information unit (PIU)
- Network control program (NCP) trace data
- Response time monitor (RTM) data

Major Tuning Techniques for the Session Monitor

Following are the major tuning techniques for the session monitor, arranged in order of expected effect on performance, with the most important tuning considerations listed first. These recommendations are described in detail in this chapter.

1. Evaluate the requirements for and use of session history data at your installation. Filter unnecessary data using either the KCLASS DASD parameter or the DASD sense code filter. You can use sample CNMSJM10 (described in the Tivoli NetView for z/OS Installation: Configuring Additional Components) to analyze your session monitor database and determine which sense codes occur most frequently. Session ends that are filtered do not require VSAM I/O activity. See “DASD Option” on page 71 and “DASD Filtering” on page 73.

2. Evaluate the requirements for and use of session awareness data at your installation. Filter unnecessary data using either the VTAM SAW filter (in ISTMGC10) or the NetView SAW filter, which uses the KCLASS SAW option. Use of the VTAM SAW filter is preferable, because it eliminates NetView processing for filtered sessions. See “SAW Data” on page 60 and “SAW Option” on page 69.

3. Purge and reorganize the database off-shift on a regular basis to reclaim free space in the database. If you can discard all of the session history data in the database, clear the session monitor database while the NetView program is active using the RESETDB command. Clearing the database with RESETDB is faster than purging and reorganizing the database. See “Managing the Session Monitor Database” on page 73.

4. Use selective trace rather than global trace for PIU tracing. Trace only the sessions for which you need trace data. See “Estimating Storage Usage” on page 135 to estimate the data’s storage requirements. See “Trace Data” on page 63 and “KEEPPIU Parameter” on page 66.

5. Use KCLASS KEEPSESS values greater than zero only for session pairs or groups of sessions with a high incidence of session ends to keep those sessions from filling up the database. The KEEPSESS option requires additional I/O processing for session ends. See “KEEPSESS Option” on page 72 and “DGROUP Option” on page 72.

6. If you do not use the PURGE command or the PURGEB command list to manage your VSAM database, set the PURGE parameter in AAUPRMLP to DASD. Otherwise, use the default setting of SPEED. See “PURGE Parameter” on page 75.
7. Evaluate the requirements for, and use of, availability data and accounting data at your installation. Filter unnecessary data using the SESSTATS parameter on the INITMOD statement. See “AVAIL Option” on page 70.

8. Use the SESSMDIS command to monitor the amount of session monitor work being done on your system. SESSMDIS provides session counts, storage by data type, and traffic counts. See “SESSMDIS Command” on page 75.

9. Minimize processing during network activation by starting the session monitor after the network is active (a session monitor warm start). Fewer SAW buffers are required to get session awareness data for the network, and PIU data for control sessions is not sent. See SAW Buffer Allocation and Tuning.

10. If you are collecting PIU trace data or using accounting, tune the PIU buffer allocation to achieve buffer fullness between 80% and 95%. See “Trace Data” on page 63.

11. Set the LUCOUNT parameter to the number of logical units (LUs) in your network. The value of LUCOUNT is used to optimize control block search algorithms and allocate session-related storage. See “LUCOUNT Parameter” on page 79.

12. If you are collecting PIU trace data, use KEEPPIU values that are even divisors of 84 to avoid wasting storage. See “KEEPPIU Parameter” on page 66 and “KEEPPIU Option” on page 70.

13. Use different KEEPPIU values for each keep class where possible to improve the locality of reference for PIU data. See “KEEPPIU Option” on page 70.

14. If you issue the COLLECT RTM command to gather response time data from PUs with the RTM feature, issue separate COLLECT commands for subsets of the PUs. Collecting from all of the PUs at once can cause a large amount of request units to flow in the network at once. Stagger separate COLLECT commands to spread out the flow of RUs over a longer period of time. See “RTM Data Collection” on page 79.

15. Use the CANCEL command only as a last resort. If CANCEL must be issued, try to close the active NLDM VSAM database (AAUVSPL or AAUVSSL) by issuing the SWITCH AAUTSKLP,T command. This does not actually do a switch, but closes the active VSAM database. If that is not effective, issue the NetView STOP FORCE command for each active VSAM task. If it is necessary to use the MVS CANCEL command to bring down NetView and you have DFR specified, it may be necessary to delete and redefine the affected database. See “Local Shared Resources (LSR) and Deferred Write (DFR)” on page 97.

SAW Data

SAW data gives session status, session partners, and configuration data about these types of sessions:
- LU-LU
- SSCP-LU
- SSCP-PU
- SSCP-SSCP
- CP-CP

SAW data collection requires 310 to 500 bytes per session, depending on whether the session is in the same domain, cross-domain, or cross-network.

The session monitor does not collect any other data about a session (for example, RTM or trace data) if it does not have SAW data for the session.
You can keep SAW data selectively using either the SAW option of the KCLASS statement, the VTAM SAW filter, or the SAW Filter Exit (DSIEX20). See "SAW Option" on page 69 for more information about session awareness filtering.

### SAW Buffer Allocation and Tuning

The SAW buffer allocation is defined by the NLDM.SAWNUM and NLDM.SAWSIZE statements in CNMSTYLE.

SAW buffers are allocated in VTAM private storage. VTAM allocates the buffers above the 16-MB line.

The DSIAMLUT task receives the SAW buffers from VTAM and forwards them to the AAUTSKLP task for processing.

The SAW buffering ratio, the average number of SAW notifications sent per buffer, helps to characterize VTAM SAW buffering behavior. You can calculate the SAW buffering ratio for your system with the following formula, using fields that are displayed with the SESSMDIS command. To calculate the SAW buffering ratio over an interval using two observations of the SESSMDIS command, take the differences in the session start, session end, and SAW buffer counts and use the following formula to calculate.

\[
\text{SAW buffering ratio} = \frac{(\text{session starts} + \text{session ends})}{\text{SAW buffers}}
\]

Refer to "SESSMDIS Command" on page 75 for more information.

An understanding of how VTAM buffers session awareness notifications is helpful in tuning the SAW buffer allocation.

### Tuning the SAW Buffer Allocation

VTAM SAW buffering behavior occurs in the following ways:

- During steady state periods when the session monitor is keeping up with the SAW data sent from VTAM, the SAW buffering ratio is close to 1.
- During a session monitor warm start, VTAM packs SAW buffers.

A data space (ISTNMSDS) is used for transferring SAW data between VTAM and the session monitor. The following VTAM start options control VTAM SAW buffering behavior. Refer to the appropriate VTAM publication for more information on the VTAM start options.

### SAW Data Space Packing Factor

The SAW data space packing factor (start option SAWMXQPK or ISTRACON constant RACSAWPK) calculates the number of SAW data space buffers to queue before packing SAW buffers. The packing threshold equals SAWMXQPK times the number of SAW buffers in VTAM private storage (defined by the SAW BUFNUM parameter in AAUPRMLP). The default value is zero. The default value causes VTAM to pack SAW buffers if the data space is not empty. This occurs if an earlier SAW buffer sent by VTAM was not received by the session monitor. This packing behavior improves performance because as traffic increases, the buffering efficiency increases. Use the default value of zero for SAWMXQPK.
SAW Data Space Limit Factor
The SAW data space limit factor (start option SAWMAXDS or ISTRACON constant RACSAWLM) calculates the maximum buffer limit for the SAW data space. The maximum number of SAW data space buffers equals SAWMAXDS times the number of SAW buffers in VTAM private storage (defined by the SAW BUFNUM parameter in AAUPRMLP). The default value for SAWMAXDS is 100. If the session monitor is unable to keep up with the SAW traffic, VTAM queues SAW buffers in the SAW data space. VTAM continues adding SAW buffers to the data space until the maximum buffer limit is reached.

SAW Buffer Limit
The SAW buffer limit (start option MXSAWBUF or ISTRACON constant RACMXBUF) sets the maximum number of SAW buffers that can be allocated in VTAM private storage. SAW buffers back up in VTAM private storage only after the SAW data space has reached its maximum buffer limit (controlled by SAWMAXDS). When this happens, VTAM tries to conserve SAW buffers by placing only SAW data associated with termination into the buffers. When the SAW buffer limit is reached, VTAM ends the session awareness function, releasing the SAW data space and the allocated buffers. When buffers back up in VTAM private storage, extra VTAM processing time is needed to conserve SAW buffers and allocate new buffers from VTAM private storage.

Recommendations for Defining the SAW Buffer Allocation
In defining the SAW buffer allocation, data space, and limits, you can control the maximum amount of SAW data that is allowed to be kept in the system prior to the session monitor processing the data. Use the data space to keep unprocessed SAW data and avoid letting SAW data back up into VTAM private storage. The following are recommendations.

- Use the default value of zero for SAWMXQPK. This gives the best SAW buffering ratio.
- Use a BUFSIZE that is a multiple of 4 K (4 K—32 K). When VTAM is packing buffers, a larger BUFSIZE allows more SAW notifications to be sent for each buffer. You can monitor the SAW buffering ratio using fields from the SESSMDIS command.
- Use a small value such as 2 for BUFNUM. Because the data space is used to store unprocessed buffers, you do not need to define a large number of SAW buffers.
- Set SAWMAXDS in conjunction with the SAW BUFSIZE and BUFNUM parameters in AAUPRMLP to reflect the amount of data space storage that you are willing to devote to a backlog of SAW data. To determine the peak size of the SAW data space, multiply SAWMAXDS by BUFSIZE and BUFNUM. For example, if AAUPRMLP defines two 4K SAW buffers and SAWMAXDS is equal to 100, the maximum size of the SAW data space is 800K (2 * 4K * 100). Consider using a large number for SAWMAXDS to keep all unprocessed SAW buffers in the data space and avoid a backup into VTAM private storage. Small values for SAWMAXDS and MXSAWBUF result in VTAM ending the session awareness function prematurely.
- Set MXSAWBUF to a relatively low number (200 or less). If you define SAWMAXDS properly, a backup into VTAM private storage should not occur. If a backup does occur, a small value for MXSAWBUF causes VTAM to end the session awareness function before the backup is significant.
Trace Data

The session monitor provides two trace modes:
- Global
- Specific

Global Tracing

The global trace mode traces all sessions. The TRACESC and TRACELU parameters on the INITMOD statement in AAUPRMLP are used to specify whether global tracing is performed. These parameters are specified with the statements as shown in the following example.

```plaintext
NLDM.TRACESC=YES|NO
NLDM.TRACELU=YES|NO
```

Note: The default values are underlined.

TRACESC=YES indicates global tracing of SSCP/CP sessions (SSCP-LU, SSCP-PU, SSCP-SSCP, and CP-CP). TRACELU=YES indicates global tracing of LU-LU sessions. Global tracing can require a large amount of virtual and DASD storage. See "Estimating Storage Usage" on page 135 to determine the storage requirements before using global trace.

You can use the NLDM TRACE DISP command to determine which sessions are being traced. Global NLDM TRACE defaults can be changed dynamically.

Selective Tracing

If you do not want to use global trace, you can specify selective trace with the session monitor TRACE command. For example, if you want to trace only SSCP-SSCP sessions, do the following:

1. Specify TRACESC=NO and TRACELU=NO in initialization member AAUPRMLP. You can code this as shown in the following example.

```plaintext
NLDM.TRACESC=NO
NLDM.TRACELU=NO
```

2. Issue a TRACE START command for each SSCP that will be in session with VTAM. Assuming three other SSCPs whose names are CDRM1, CDRM2, and CDRM3, the session monitor TRACE commands are shown in the following example.

```plaintext
NLDM TRACE START CDRM1
NLDM TRACE START CDRM2
NLDM TRACE START CDRM3
```

You can code these commands in a NetView command list. You can also code a NetView command list that dynamically determines the SSCPs and starts the TRACE command for them. You can issue the VTAM display command, as shown in the following example, in a command list to identify the SSCPs.

```plaintext
(D NET,ID=VTAM,E)
```

Then issue NLDM TRACE START commands in the same command list for those SSCPs.

If your network experiences a high level of session failures, consider tracing the SSCP sessions. Using the KEEPPIU option of the KCLASS statement, keep more PIUs for the following:
- SSCP-SSCP sessions
- SSCP-PU sessions for gateway NCPs
- SSCP-LU sessions for VTAM applications

For all other SSCP sessions, keep fewer PIUs. For networks with hung terminals and protocol problems, consider tracing LU-LU sessions.

The type of information available from PIU tracing varies depending on the session type. The following are some examples:

- Session initiations (INITs) for SSCP-LU sessions for applications
- Cross-network session resource allocation (RNAAs, SETCVs) for SSCP-PU sessions for gateway NCPs
- Cross-domain and cross-network session initiations (CDINITs) for SSCP-SSCP sessions

Activate trace for other session types or resources when needed to reproduce a problem or to monitor selected sessions.

The accounting function summarizes the PIU data by session. When you use the accounting function, specified with SESSTATS=YES in AAUPRMLP, the session monitor receives and processes the same amount of PIU trace data as if global trace had been specified. If you specify a trace for the session with the TRACESC or TRACELU parameters or the TRACE command, the PIU data is available for viewing and recording. Tracing does not have to be active to use the accounting function.

If you use the availability function, specified with SESSTATS=AVAIL in initialization member AAUPRMLP, rather than the accounting function, the session monitor does not require tracing to be active. Therefore, the processing and storage associated with global tracing can be eliminated by specifying TRACESC=NO and TRACELU=NO in member AAUPRMLP.

The amount of host processing required for the accounting function is similar to that required for global trace. Global trace can require considerably more storage than accounting, however, depending on the number of PIUs kept in storage for each session. See "KEEPPIU Parameter" on page 66 for information about how to specify the number of PIUs to keep for a traced session.

**Note:** When you filter SAW data with VTAM, VTAM does not send PIU data for filtered sessions to the session monitor. If you filter SAW data with the session monitor KCLASS statement instead of the VTAM SAW filter, VTAM sends PIU data for filtered sessions to the session monitor and the data is discarded. See "SAW Option" on page 69 for more information on session awareness filtering.

If you want to use global trace or accounting and restrict the number of sessions for which PIU data is processed, you can use the VTAM SAW filter to filter unneeded sessions. Keep in mind that if you filter a session, no other data can be collected for that session.

**PIU Buffer Allocation and Tuning**

Define trace buffers in CNMSTYLE using the NLDM.PIUTNUM and NLDM.PIUTSIZE statements.

The DSIAMLUT task receives the PIU buffers from VTAM and forwards them to the AAUTSKLP task for processing. The storage for PIU buffers is common storage.
page fixed. If VTAM does not have enough trace buffers and starts overlaying them, the NetView program issues message AAU024I indicating the number of lost buffers. Losing PIU buffers is usually not a problem unless you are using the session monitor accounting function or are not getting the desired trace data.

Three goals for tuning the PIU buffer allocation are:
- Maximize the number of PIUs that are sent per buffer.
- Avoid losing PIU buffers.
- Avoid over allocating buffers, which wastes storage.

VTAM sends full PIU buffers unless the NetView program solicits a buffer, as it might when processing a session-end notification or when processing an operator request. Therefore, the optimum PIU buffer allocation depends on the rate at which session-end notifications are processed and the frequency of operator requests for PIU trace data.

In tuning the PIU buffer allocation, determine the optimum size of the buffers for your environment. Then adjust the number of buffers to allocate so that PIU buffers are not lost during periods of peak activity.

The SESSMDIS command and the external log record created by the NLDM RECORD STRGDATA command provide information that you can use to tune the PIU buffer allocation for your system. The SESSMDIS command displays session monitor traffic data and event counters, while the NLDM RECORD STRGDATA command writes similar information to the NetView external log. The NLDM RECORD STRGDATA command creates SMF record type 39, subvector X'0008'. Refer to Tivoli NetView for z/OS Command Reference for information about the syntax and use of those commands. Refer to the Tivoli NetView for z/OS Application Programmer's Guide for the format of the external log record.

Estimating PIU Buffer Fullness
Both the SESSMDIS and NLDM RECORD STRGDATA commands provide counters for the number of PIU buffers processed and the number of PIUs processed by the NetView program. Use the counters to estimate the fullness of the PIU buffers sent from VTAM to the NetView program using the following steps:

1. Collect the PIU buffer counts and PIU counts (with the SESSMDIS command) from the NetView program over a period of 1 to 2 hours during peak activity for your system.

2. Using your current PIU buffer size and the PIU buffer and PIU counts from your period of peak activity, estimate the average percent fullness of your PIU buffers using the following formula:

\[
\% \text{ full} = \frac{\text{PIU count} \times 48}{\text{PIU buffer count} \times \text{current PIU buffer size}} \times 100
\]

The constant 48 represents the size in bytes of the vast majority of PIUs that the NetView program processes. Because the PIU buffer contains some buffer header information, the buffer fullness percentage will approach but never equal 100%.

In general tune your buffer size to achieve buffer fullness of 80% or more. If your PIU buffers are less than 80% full and storage is a constraint on your system, you can decrease the buffer size.
If your PIU buffers are 95% full or more, you might benefit from increasing the PIU buffer size. A larger buffer size might allow you to get more PIUs sent in each buffer.

The samples contain a starting allocation of two 4 K PIU buffers. If you are in a large environment, you might want to use four 8 K PIU buffers for your starting allocation.

In the process of tuning your PIU buffer size, adjust the number of buffers so that the total size of your PIU buffer allocation (number of buffers multiplied by buffer size) does not vary significantly. Once you have arrived at a satisfactory PIU buffer size, you can adjust the number of PIU buffers so that you do not lose buffers during periods of peak activity, such as during a major node recovery.

If you still receive message AAU024I (indicating lost buffers) after tuning the PIU buffer allocation, the session monitor might not be dispatched frequently enough to sustain its workload. This is an indication of a performance problem, where higher priority work on your system (either in the NetView Program or in other address spaces) has high CPU usage. Raising the dispatch priority for AAUTSKLP (specified in CNMSTASK with the PRI parameter) from the default of 8 to 5 might help relieve this problem. However, you should try to address the underlying problem of high CPU usage with the TASKUTIL command and system performance monitoring tools, such as RMF.

PIU Data Space
A data space (ISTNMPDS) is used to transfer PIU trace data between VTAM and the session monitor. Using the data space does not affect selecting the proper PIU buffer size for your environment, but does affect the number of PIU buffers that you should specify.

The VTAM PIUMAXDS start option (or ISTRACON constant RACPIULM) is the PIU data space limit factor used to calculate the maximum buffer limit for the PIU data space. The maximum number of PIU data space buffers equals PIUMAXDS times the number of PIU buffers in VTAM private storage (defined by the PIU BUFNUM parameter in AAUPRMLP). The default value for PIUMAXDS is 200. If the session monitor is unable to keep up with the PIU traffic, VTAM queues PIU buffers in the PIU data space. VTAM continues to add PIU buffers to the data space until the maximum buffer limit is reached. VTAM then discards PIU data until buffers become available. When the session monitor detects that PIU buffers were lost, message AAU024I is issued.

Set PIUMAXDS in conjunction with the PIU BUFSIZE and BUFNUM parameters in AAUPRMLP to reflect the amount of data space storage that you are willing to devote to a backlog of PIU data. To estimate the peak size of the PIU data space, multiply PIUMAXDS by BUFSIZE and BUFNUM. For example, if AAUPRMLP defines four 8K PIU buffers and PIUMAXDS is equal to 200, the maximum size of the PIU data space is 6400 K (4 * 8 K * 200).

KEEPPIU Parameter
The KEEPPIU parameter in AAUPRMLP specifies the number of PIUs to keep for a traced session and affects virtual storage, processor storage, and DASD storage. The value provided in the samples is 7. KEEPPIU affects the amount of virtual storage the session monitor uses for the period a session is active, and the amount of DASD storage used when a session is deactivated. For each PIU kept, 50 bytes of virtual storage are used per session.
The session PIUs are kept in a wraparound area in virtual storage while the session is active and are stored in the database when the session ends. When the first PIU for a session is received, the entire KEEPPIU storage for that session is allocated. Specifying a large KEEPPIU value for sessions that do not have much traffic, such as SSCP-LU and SSCP-PU sessions, could result in wasted storage. See “KEEPPIU Option” on page 70 for information on tailoring the KEEPPIU value for different keep classes.

Use the NetView SESSMDIS command to display the amount of storage allocated for the PIU trace data. This information is helpful in determining a good value for KEEPPIU. See “SESSMDIS Command” on page 75 for a description of the syntax and use of this command.

**TRACEGW Parameter**

The TRACEGW parameter in AAUPRMLP specifies whether NCP gateway trace data is to be collected for cross-network sessions. If TRACEGW is set to YES, when the session monitor receives a session-start notification for an NCP, it requests the NCP to collect gateway trace data for cross-network sessions that pass through it. When cross-network sessions end, the NCP sends the data to the session monitor for recording to the VSAM database. If you do not use this trace data for problem determination or other purposes, you can set TRACEGW to NO and eliminate the extra processing and VSAM storage.

**Keep Classes**

You can use keep classes to keep selective SAW data for active sessions in addition to recording selective session data on sessions that have ended. Keep classes can also control the amount of trace data collected. As mentioned in “SAW Data” on page 60, if you plan to keep SAW data selectively, consider filtering SAW data with the VTAM SAW filter.

KCLASS and MAPSESS statements are used to define the selectivity of keep classes, as shown in Figure 16 on page 68. These statements are in a separate DSIPARM data set member, which is defined in CNMSTYLE by the NLD.M.KEEPMEM statement.

KCLASS defines selective processing options (AVAIL, SAW, DASD, KEEPSESS, DGROUP, and KEEPPIU). MAPSESS defines which sessions use a specified KCLASS.

KCLASS and MAPSESS definitions are loaded during session monitor initialization. You can change these definitions dynamically using the session monitor RELOAD command to load new members. Active sessions are not affected by the RELOAD command.

Keep-class processing is done on a session basis and options are applied when the session monitor receives a session start. The session partner names are used as criteria to search the MAPSESS statements until a match is found. You can use the question mark (?) and asterisk (*) for matching names when you follow naming conventions. These are called pattern-matching or wildcard characters. The ? wildcard character holds a place in the column, specifying that any character can be there. The * wildcard character specifies a match with any character in the column along with any characters to the end of the name.
**Network Resource Naming Conventions**

Network resource naming conventions enable keep-class coding. The following are example naming conventions:

- **CDRM**<sub>nnnn</sub>
  - CDRM names
- **NCPGW**<sub>nnnn</sub>
  - Gateway NCP names
- **NCP**<sub>nnnn</sub>
  - NCP names
- **nnnPU**<sub>nnnn</sub>
  - PU names
- **nnnLU**<sub>nnn</sub>
  - SNA LU names
- **nnnLU**<sub>B</sub>
  - Bisynchronous LU names
- **CICS**<sup>®</sup><sub>nnnn</sub>
  - CICS® applications
- **TSO**<sub>nnnn</sub>
  - TSO applications

Figure 16 uses this naming convention to specify processing options for different classes of sessions. The PLU name for SSCP sessions is VTAM in this figure.

**Figure 16. Example of KCLASS and MAPSESS Statements**

The search order against the MAPSESS statements is from top to bottom. The order of the MAPSESS statements and the accuracy of the PLU and SLU names are important. The first match is used for the keep-class member. When a session’s PLU and SLU names match a MAPSESS statement, the search ends and the keep-class processing is determined by the KCLASS to which the MAPSESS points.
Use the session monitor DISKEEP PIU command to verify that the intended sessions are being mapped to the desired KCLASS statements. Refer to the Tivoli NetView for z/OS Installation: Configuring Additional Components for more information about keep-class processing.

SAW Option

You can keep SAW data selectively by using either the SAW option of the KCLASS statement or the VTAM SAW filter. If you plan to keep SAW data selectively, filter SAW data with the VTAM SAW filter. When SAW data is filtered by the session monitor, VTAM transports the data to the session monitor, which discards the data. Filtering SAW data with VTAM avoids the overhead of transporting the filtered data. When sessions are filtered by the VTAM SAW filter, PIU data for the filtered sessions is not sent to the session monitor. VTAM session awareness filtering is described in VTAM Network Implementation Guide.

If you decide to filter SAW data, use care in deciding which sessions to filter. If the SAW data for a session is filtered, no other data can be collected for that session. Consider the following when filtering SAW data:

- SAW filtering with the session monitor KCLASS statement applies only to SSCP-LU and LU-LU sessions. You cannot filter SSCP-PU and SSCP-SSCP SAW data for the session monitor, because session awareness data for these sessions is required in many session monitor functions, such as the response time monitor (RTM), gateway trace, and boundary function trace.
- The VTAM SAW filter enables you to filter SSCP-PU and SSCP-SSCP sessions. Do not filter SSCP-PU and SSCP-SSCP session awareness data unless you are certain that you do not require this data for your environment. If you collect RTM data using the COLLECT RTM * command, filtering SSCP-PU session awareness for a subarea PU causes data from RTM devices in that subarea to be omitted from the collected data.
- RTM data requires session awareness data (SAW=YES) for the related SSCP-PU, SSCP-LU, and LU-LU sessions.
- Non-RTM SSCP-LU sessions and low-priority, stable LU-LU sessions are good candidates for SAW filtering.

DSIEX20-SAW Filter Exit

Installation exit DSIEX20 allows more granular processing of NetView session awareness (SAW) data records. This exit allows the user to accept or discard SAW records based on their content. The exit causes SAW data received from VTAM to be filtered by NetView Session Monitor prior to filtering associated with a KEEPMEM defined with an INITMOD statement in AAUPRMLP containing KCLASS and MAPSESS statements.

The sample DSIEX20 that is shipped with NetView filters out all SSCP-LU session data at session start. The sample DSIEX20 is meant to be used as an example. It should be customized to meet your individual needs. For more information on Exit 20, refer to Tivoli NetView for z/OS Customization: Using Assembler.

Using DSIEX20 can greatly reduce the CPU and storage resources used by the Session Monitor, primarily with NLDMs main task, AAUTSKLP. However, careful consideration should be given when coding and using this exit. SAW data filtered at initialization may cause certain other functions within session monitor not to have the required data needed for the function. For example, the DSIEX20 sample provided with NetView filters all SSCP-LU data. Having this data filtered at initialization would cause RTM data to be unavailable for the filtered sessions, since RTM must have SSCP-LU data.
**KEEPPIU Option**

If you are tracing, tailor the number of PIUs kept for different session types to optimize processing and control the amount of trace data kept. The KEEPPIU parameter value is determined as follows:

- If the KEEPMEM member has been coded and the session is mapped to a KCLASS, use the KEEPPIU value coded on the KCLASS statements.
- If the KEEPMEM member has not been coded, use the KEEPPIU value coded on the INITMOD statement in AAUPRMLP (or equivalent). The value in the samples is 7.

Where possible, each keep class should have a different KEEPPIU value. The PIU wrap areas for sessions with the same KEEPPIU value are grouped on the same pages of virtual storage. Separating keep classes by using different KEEPPIU values can improve the locality of reference for PIU data.

To minimize wasted storage, the values you specify for KEEPPIU should be even divisors of 84 (2, 3, 4, 6, 7, 12, 14, 21, 28, or 42). Avoid using values greater than 84. Larger values increase virtual storage requirement of the session monitor and cause the PIU wrap area to span more than one page of virtual storage, requiring a GETMAIN rather than using pooled storage. Small KEEPPIU values keep the virtual storage requirement of the session monitor to a minimum.

The following are some suggested KEEPPIU values for different types of sessions:

**LU-LU (APPL-APPL) sessions**
- 42

**LU-LU sessions for bisynchronous terminals**
- 14

**LU-LU sessions for SNA terminals**
- 12 or 7 (depending on available storage)

**SSCP-LU sessions for applications**
- 21

**SSCP-LU sessions for terminals**
- 4

**SSCP-PU sessions for gateway (GW) NCPs**
- 84

**SSCP-PU sessions for other NCPs**
- 42

**SSCP-PU sessions for other PUs**
- 6

**SSCP-SSCP sessions**
- 84

**CP-CP sessions**
- 84

**AVAIL Option**

The SESSTATS and LOG parameters in initialization member AAUPRMLP specify whether session start and session end records are written to the external log. The records are used to track session accounting (availability and PIU counts) or session availability. If you do not need session accounting, which includes PIU
trace counts, specify SESSTATS=AVAIL on the INITMOD statement in member AAUPRMLP. If you specify the availability option at initialization, you can define whether availability data should be kept for a class of sessions using the AVAIL parameter on the KCLASS statement. You can use the RELOAD command to dynamically change availability by keep class.

When global tracing is not requested (TRACESC=NO and TRACELU=NO) and the availability option is used rather than accounting, the session monitor CPU utilization and working-set size are reduced.

**DASD Option**

Data for a session is written to the VSAM database when the session ends. This data includes SAW data plus any optional trace and RTM data. For large networks or networks with a high incidence of session ends, the processing required for data recording can be substantial. An important tuning consideration is to evaluate your requirements and use of session history data. If there is history data you know you do not need, filter it out using the KCLASS DASD option.

The following is the list of values for the DASD option of keep-class processing:

- **DASD=BINDFAIL**
  - Record only if session fails during BIND (BINDF).
- **DASD=DATA**
  - Record only if trace or RTM data is present.
- **DASD=FAILURES**
  - Record if session fails or abnormal UNBIND occurs.
- **DASD=INITFAIL**
  - Record only if session fails before BIND (INITF).
- **DASD=NO**
  - Do not record any session data.
- **DASD=RTMDATA**
  - Record only if RTM data is present for this session.
- **DASD=SESSFAIL**
  - Record only if abnormal UNBIND occurs.
- **DASD=SESSNORM**
  - Record only if normal UNBIND occurs for this session.
- **DASD=TRACDATA**
  - Record only if trace data is present for this session.
- **DASD=YES**
  - Always record the session data (default).

The following is an example of coding multiple options on a KCLASS DASD statement.

```
DASD=(BINDFAIL,RTMDATA,INITFAIL)
```

You can code multiple conditions in a single specification. This example of the DASD option specifies that session data is recorded:

- If the session fails during BIND (BINDF)
- If RTM data is present for this session
- If the session fails before BIND (INITF)

**Filtering with the KCLASS DASD Option**

By filtering data recording with the KCLASS DASD option, you can reduce host processor utilization, VSAM I/O, and DASD storage used by the session monitor. Analyze the network problems your installation encounters and determine the types and amounts of session monitor information needed to diagnose the problems. For LU-LU sessions, consider using DASD=DATA, DASD=FAILURES, or
DASD=RTMDATA. For SSCP sessions for which you do not need information, use DASD=NO. For more information, see "DASD Filtering" on page 73.

If you issue the FORCE command, the session data is recorded regardless of how you coded the DASD parameter.

Special Relationships between the SAW and DASD Parameters

- Unless filtered by VTAM, SAW data is always kept for SSCP-SSCP and SSCP-PU sessions, no matter what is coded on the KCLASS statement.
- If you code SAW=NO on the KCLASS statement for LU-LU sessions, only BINDF and INITF are kept as if you coded DASD=(INITFAIL,BINDFAIL).
- If you code SAW=NO with the INITMOD statement in AAUPRMLP, SAW data is not kept for any session regardless of the session type.

KEEPSESS Option

The KEEPSESS value for a session, specified with either the KEEPSESS parameter in AAUPRMLP or the KEEPSESS option on the KCLASS statement, controls the number of session incidences that are recorded on the session monitor database for a given name pair or DASD group. See "DGROUP Option" for more information on DASD groups. The KEEPSESS value for a session is determined as follows:

- The KEEPSESS parameter on the INITMOD statement in AAUPRMLP indicates whether DASD session wrapping is used. If you do not specify a value, the default is zero and session wrapping is not used regardless of any KCLASS KEEPSESS values. Also, sessions will not be recorded into DGROUPs as defined on a KCLASS statement. If you do specify a value, the value is used as the global DASD session wrap count for sessions not mapped by MAPSESS/KCLASS statements and for mapped sessions having no KEEPSESS coded.
- If the global DASD session wrap count (the KEEPSESS parameter in AAUPRMLP) is greater than zero, the KEEPMEM member has been coded, and the session is mapped into a KCLASS through a MAPSESS statement, the value of the KEEPSESS option coded on the KCLASS statement is used. If the KEEPSESS option is not coded on the KCLASS statement, the global KEEPSESS value in the range of 1–999 will be used for sessions mapping into this KCLASS.
- If the global DASD session wrap count is greater than zero, the KEEPMEM member has been coded and the session is not mapped to a KCLASS, the global DASD session wrap count coded in AAUPRMLP is used. If the bottom MAPSESS statement is coded with PRI=* and SEC=* (for example, MAPSESS KCLASS=ALLOTHER,PRI=*,SEC=*), then all sessions that do not match a previous MAPSESS statement will be mapped into the KCLASS specified (ALLOTHER in this example).

See "Managing Database Size" on page 74 for recommendations on controlling the size of the session monitor database.

DGROUP Option

The DGROUP option can be used in conjunction with the KEEPSESS option on the KCLASS statement to control the number of session incidences that are recorded on the session monitor database for a group of sessions. Using the KEEPSESS option alone is not effective for controlling the number of session incidences on the database for applications, such as TSO and NetView, where the primary LU (PLU) session partner name contains a counter (for example, TSO00001, TSO00002, NETV001, NETV002). The DGROUP option can be used to group sessions together,
so that the KEEPSESS value can be applied to control the number of session incidences for the group instead of for the individual name pairs.

DGROUP specifies the grouping characteristics of all MAPSESS sessions mapping to a KCLASS statement. You can group sessions under a user-supplied name, or defer the DGROUP name until the session ends by using the *PRI or *SEC values. Deferring the DGROUP name enables the definition of multiple DGROUPs with a single KCLASS statement, with the DGROUP name being either the primary or secondary session partner name. All defined DASD GROUPs (DGRPs) can be displayed using the NLDM LIST DGRP command.

Figure 17 shows an example of how to use the KEEPSESS and DGROUP options to control the number of session incidences on the database for a group of sessions. In this figure, the NetView domain identifier is N2412, and the TSO VTAM application name is TSO12.

```
::
TSO  KCLASS SAW=YES,+
     DASD=FAILURES,+
     KEEPSESS=200,+
     DGROUP=(TSO,RENAME,PRI)
NETVIEW KCLASS SAW=YES,+
     DASD=FAILURES,+
     KEEPSESS=100,+
     DGROUP=(NETVIEW,RENAME,PRI)
::
TSO  MAPSESS KCLASS=TSO,PRI=TSO12*,SEC=*
NETVIEW MAPSESS KCLASS=NETVIEW,PRI=N2412*,SEC=*
::
```

Figure 17. Example of KCLASS and MAPSESS Statements using the KEEPSESS and DGROUP options

Managing the Session Monitor Database

The following sections describe techniques for managing the session monitor database.

**DASD Filtering**

You can filter session history recording using the DASD option of the KCLASS statement, which enables you to specify recording conditions for sessions mapped to a keep class. For example, you can record only sessions that have RTM data by specifying DASD=RTMDATA for the keep class. See “DASD Option” on page 71 for information about the DASD option.

You can also filter session history recording based on the sense codes and reason codes that accompany the session awareness notification. The sense code filter consists of 25 entries in DSICMOD (CNMS0055). Each entry consists of an 8-byte field specifying the sense and reason codes, and a numeric field indicating the number of bytes used for comparison in the filter. By modifying DSICMOD, you can filter DASD recording based on up to 25 sense codes.

To analyze your session monitor database use sample CNMSJM10, these samples print a report containing the distribution of the sense codes on the database. Use this report to determine which sense codes occur most frequently and where to concentrate your filtering efforts.
Filter sense codes that occur frequently and are not useful in network problem
determination. For example, X’087D0001’ is a normal response to an unsuccessful
cross-network session setup when you have multiple gateways and VTAM checks
an incorrect gateway name first.

Refer to the [Tivoli NetView for z/OS Installation: Configuring Additional Components](#) for information about deciding which sense codes to filter, how to add sense codes
for filtering, and how to end sense code filtering.

**Managing Database Size**

As sessions end, the session monitor records history data to the VSAM database.
The database eventually fills up. There are three principal methods to manage the
size of the database.

1. Clear the entire database using the RESETDB command or the CLEAR
   parameter of the DBAUTO command. While this method is quick, all of the
   session history data is lost at once, rather than in stages. To use this method,
   ensure that the REUSE operand is specified for both AAUVSPL and AAUVSSL
   on the cluster definition in member CNMSI201.

2. Purge data for sessions that ended before a specified time by using either the
   PURGE command or the PURGE parameter of the DBAUTO command. An
   advantage of this method is that the purge activity can be scheduled for an
   off-shift period, deferring the I/O needed to delete data and minimizing
   prime-shift session monitor I/O. For large databases, however, the purge
   activity can take a long time.

3. Use the KEEPSESS parameter or the KEEPSESS option of the KCLASS
   statement to control the number of session incidences on the database for a
   given name pair or group of sessions (specified by the DGROUP option). For
   example, if the KEEPSESS value for a name pair is 10, up to 10 incidences are
   maintained on the database. When the 11th incidence is recorded, the first
   incidence is erased. Using KEEPSESS requires additional I/O to delete data
   when sessions end. Session ends that delete data (KEEPSESS value exceeded)
   require approximately three times the I/O required to record sessions that do
   not delete data. See [“KEEPSESS Option” on page 72](#) for more information about
   KEEPSESS.

Method 1 (clearing the entire database) performs better than the other two
alternatives. However, if method 1 does not meet your needs, method 2 (purging
the database) is preferred over method 3 (using KEEPSESS). Purging data enables
you to defer session incidence deletions to off-peak hours.

Methods 2 and 3 require that you reorganize the database periodically to reclaim
previously used free space. Without periodic reorganization, the VSAM data set
eventually runs out of free space, causing an end-of-file condition. See [“VSAM
Database Maintenance” on page 105](#) for information about reorganizing a VSAM
database.

You can control your database with KEEPSESS and purge activity in combination.
If certain name pairs or groups of sessions have a high incidence of session ends
and you want to keep them from filling up your database, use KEEPSESS to limit
the session incidences for those session pairs, and purge the data for other
sessions. Because additional processing is required to control the wrap count,
purge activity is more efficient if you do not purge sessions controlled by
KEEPSESS.
**PURGE Parameter**

The PURGE parameter in AAUPRMLP is used to control writing an end-time record to the database for use in purge processing. PURGE=SPEED (the default) writes an end-time record. Specifying PURGE=DASD instructs the session monitor to optimize DASD space and not write the end-time record. Use the following guidelines for tuning this parameter.

- Use the default of PURGE=SPEED if you use the PURGE command to control your database.
- Set PURGE=DASD if you use KEEPSESS, because purge processing is not done and the end-time records are not needed.
- Use PURGE=SPEED if you use both PURGE and KEEPSESS (using KEEPSESS selectively and using purge processing for the rest of your database). PURGE=DASD has a more adverse effect on purge processing than the extra end-time record has on session history recording.
- Set the PURGE parameter to DASD if you use RESETDB. There is no need for the end-time record in this case. Ensure that your NLDM VSAM cluster definition (CNMSI201) has the REUSE operand. REUSE is required when using the RESETDB command.

**SMDR Command**

You can stop session monitor data recording by using the NLDM SMDR STOP command. This command is not recommended when a purge is running, as NLDM has separate control blocks for updating and deleting records from the VSAM database in use. When the SMDR STOP command is issued, sessions on the VSAM Record Queue go through normal cleanup processing, but the session history data is discarded instead of being written to VSAM. Recording to the external log (SMF) proceeds normally and is not affected by the SMDR command. The SMDR START command resumes the recording of session history data to VSAM. Message AAU273I is sent to the NetView log stating that VSAM recording has resumed. This message also notifies the user of the total number of sessions lost (session history data discarded) due to SMDR STOP.

If a severe VSAM I/O error occurs, NLDM will automatically deactivate session monitor VSAM recording. Browse the NetView log or system log for either message AAU272I or message AAU022I. Determine the error by interpreting the major and minor return codes contained in the error message. When the VSAM I/O error is resolved, issue SMDR START to resume VSAM recording.

SMDR QUERY can be used to determine database VSAM recording status. This is helpful if SESSMDIS is showing a backup on the VSAM Record Queue.

Refer to [Tivoli NetView for z/OS Command Reference](index.html) for a description of the SMDR command.

**SESSMDIS Command**

The SESSMDIS command displays session monitor session counts, storage use, and workload traffic information. SESSMDIS uses a view panel to display a subset of the counters that are written to the external log with the NLDLM RECORD STRGDATA command. Figure 18 on page 76 shows sample output of the SESSMDIS command.
### The following list describes the fields displayed:

1. **Session monitor options in effect**
   - **SAW**: YES
   - **LU Trace**: NO
   - **CP/SSCP Trace**: NO
   - **SESSTATS**: NO

2. **Session counts (current and maximum)**
   - **CP-CP**: 0 (Current), 0 (Maximum)
   - **SSCP-SSCP**: 0 (Current), 0 (Maximum)
   - **SSCP-PU**: 1 (Current), 35 (Maximum)
   - **SSCP-LU**: 8 (Current), 8 (Maximum)
   - **LU-LU**: 0 (Current), 0 (Maximum)

3. **Session Monitor Storage Usage**
   - **Resources**: 8K
   - **Sessions**: 16K
   - **SessionParms**: 0K
   - **RTM**: 0K
   - **RSCV**: 0K
   - **Other**: 1299K
   - **Total**: 1323K

4. **VSAM Record Queue**
   - **Current**: 0
   - **Maximum**: 3

5. **Session Monitor Workload since 05/26/01 15:29:59**
   - **SAW Buffers**: 86
   - **Session Starts**: 89
   - **Session Ends**: 44
   - **PIU Buffers**: 0
   - **PIUs**: 0
   - **Sessions Recorded**: 44

---

**Figure 18. Session Monitor Session and Storage Information Panel**

The following list describes the fields displayed:

1. **Session monitor options in effect**
   - **SAW**: Displays the setting of the SAW parameter in AAUPRMLP.
   - **LU Trace**: Displays the setting of the TRACELU parameter in AAUPRMLP.
   - **CP/SSCP Trace**: Displays the setting of the TRACESC parameter in AAUPRMLP.
   - **SESSTATS**: Displays the setting of the SESSTATS parameter in AAUPRMLP.

2. **Session counts (current and maximum)**
   - **CP-CP**: The number of active CP-CP sessions for which the session monitor is maintaining session awareness.
   - **SSCP-SSCP**: The number of active SSCP-SSCP sessions for which the session monitor is maintaining session awareness.
   - **SSCP-PU**: The number of active SSCP-PU sessions for which the session monitor is maintaining session awareness. This includes PU type 4 and PU type 2 sessions.
   - **SSCP-LU**: The number of active SSCP-LU sessions for which the session monitor is maintaining session awareness.
   - **LU-LU**: The number of active LU-LU sessions for which the session monitor is maintaining session awareness.
Filtered
The number of active sessions (of all types) that are being filtered by either the VTAM or the NetView SAW filters.

Session monitor storage usage

Resources
The number of KB of storage allocated for resource information.

Sessions
The number of KB of storage allocated for session information.

SessionParms
The number of KB of storage allocated for session parameter information. This information is kept only for sessions that are being PIU traced.

PIU Trace
The number of KB of storage allocated for PIU trace information. This storage is used for the PIU wrap areas. Modifying the KEEPPIU parameter or the KEEPPIU option of the KCLASS statement changes this number. The PIU wrap area for a session is allocated when the first PIU for the session is received.

SESSTATS
The number of KB of storage allocated for session accounting or availability information.

- If accounting is active (SESSTATS=YES), the accounting storage for a session is allocated when the first PIU for the session is received.
- If availability is active (SESSTATS=AVAIL), the availability storage for a session is allocated when the start notification for the session is received.

RTM
The number of KB of storage allocated for response time monitor (RTM) data. Modifying the KEEPRTM parameter can change this number. The RTM wrap area for a session is allocated when the first RTM data is received for the session. If you do not collect RTM data with the COLLECT command, the wrap area is allocated when the RTM data is received at session termination. A small KEEPRTM parameter is useful in that case.

RSCV
The number of KB of storage allocated for route selection control vector data.

Other
This field contains miscellaneous storage being used by the session monitor, such as work storage and internal control blocks.

Total
The total amount of storage allocated for active sessions. When sessions end, rather than freeing the storage back to the operating system, the session monitor keeps the storage and reuses it for new sessions (to avoid dynamic storage get and free requests). Therefore, the total storage given on the SESSMDIS panel is not necessarily equal to the total virtual storage allocated by the session monitor.

VSAM Record Queue (current and maximum)
The number of sessions that have ended and are waiting to be recorded to VSAM. This number includes sessions that may have their session history recording filtered (see “DASD Option” on page 71). Sessions that have ended spend a few seconds on the VSAM record queue so that related data can arrive from the network. If a large number of sessions end in a short amount of time, the record queue can build up. By repeatedly refreshing the SESSMDIS panel, you can watch the queue shrink and grow.

Session monitor workload traffic counters (total and 4-second rate)

The “since” time stamp indicates when the last NLDM RECORD STRGDATA command was issued or when the session monitor was started. The following traffic counters represent the number of traffic items processed since the time stamp.

**SAW Buffers**
- The number of session awareness buffers that have been processed.

**Session Starts**
- The number of session starts that have been processed.

**Session Ends**
- The number of session ends that have been processed.

You can use the previous three numbers to estimate the SAW buffering ratio. The buffering ratio can be calculated by summing the starts and ends, and dividing by the number of SAW buffers. The SAW buffering ratio is useful in tuning the SAW buffer allocation. See “SAW Buffer Allocation and Tuning” on page 61 for more information.

**PIU Buffers**
- The number of PIU buffers that have been processed.

**PIUs**
- The number of PIUs that have been processed.

You can use these numbers to estimate the PIU buffering ratio and the average fullness of the PIU buffers. For the PIU buffering ratio, divide the number of PIUs by the number of PIU buffers. See “PIU Buffer Allocation and Tuning” on page 64 for information on estimating the PIU buffer fullness percentage and tuning the PIU buffer allocation.

**Sessions Recorded**
- The number of sessions for which VSAM recording has taken place (sessions that are not DASD filtered).

You can use the number of sessions with the Session Ends count to determine NLDM’s session recording rate. If the number of sessions recorded is equal to the number of session ends, then no sessions have DASD recording filtered. If AAUTSKLP is showing high CPU usage from TASKUTIL output, consider filtering DASD recording. If the rate of session ends is constantly high, determine if certain sessions are being restarted and immediately ending again. This can be determined by running sample job CNMSJM10 and examining the output for specific sense codes that would indicate a failure (for example: INITF, BINDF). See “DASD Filtering” on page 73 for more information.

The SESSMDIS command can be executed under an autotask and its output directed to the network log. Consider setting a timer to execute SESSMDIS under an autotask, so that you can examine the SESSMDIS output later for workload activity trends. Beginning in Tivoli NetView for z/OS V5R1, the total current
explicit route (SART) count can be viewed. To obtain SART data, the SESSMDIS
color must be issued either in a window (for example, Window SESSMDIS) or
the results of the SESSMDIS command must be sent to the log. The SART data can
not be displayed on the SESSMDIS panel as space does not allow. An additional
line of output is returned for the Window SESSMDIS command and it is also returned
to the NetView log. It appears at the bottom of the SESSMDIS output in the
following format:

TOTAL CURRENT EXPLICIT ROUTES (SARTS): 37

The SART information can be used as a guide in setting the value of
NLDM.ER.COUNT in CNMSTYLE or in AAUPRMLP. Complete Window SESSMDIS
output looks similar to the following example:

DSI378I SESSMDIS DISPLAY
SESSMDIS SESSION MONITOR SESSION AND STORAGE INFORMATION 13:14:26
OPTIONS IN EFFECT SAW: YES LU TRACE: NO CP/SSCP TRACE: NO SESSTATS: NO
SESSION COUNTS CP-CP SSCP-SSCP SSCP-PU SSCP-LU LU-LU FILTERED
CURRENT: 0 1 1 29 5 0
MAXIMUM: 0 1 1 29 6 0
SESSION MONITOR STORAGE USAGE
RESOURCES: 8K SESSIONS: 12K SESSION PARMS: 0K
PIU TRACE: 0K SESSTATS: 0K RTM: 0K
RSCV: 0K OTHER: 1673K
TOTAL: 1693K

VSAM RECORD QUEUE
CURRENT: 0 MAXIMUM: 2
SESSION MONITOR WORKLOAD SINCE 05/16/02 13:13:51
SAW SESSION PIU PIUS SESSIONS
BUFFERS STARTS ENDS BUFFERS RECORDED
4 SECONDS: 0 0 0 0 0
TOTAL: 4 39 2 0 0 2

TOTAL CURRENT EXPLICIT ROUTES (SARTS): 1

### RTM Data Collection

The COLLECT RTM command gathers response-time data from PUs with the RTM
feature. If you collect data from a large network, issue separate COLLECT
commands for subsets of the PUs rather than collecting from all of the PUs at once
(using COLLECT *), which can cause a large burst of RUs to flow in the network.
Stagger the separate COLLECT commands to spread out the flow of RUs over a
longer period of time.

### LUCOUNT Parameter

LUCOUNT is a parameter in the initialization member for AAUTSKLP (default
AAUPRMLP) that:

- Specifies the number of LUs known by the session monitor. The default value is
  4000.
- Optimizes control block search algorithms.
- Allocates session-related storage.

Specify the number of LUs in your network for LUCOUNT. This number should
include SNA interconnection (SNI) sessions in your network. The number you
specify does not have to be exact; however, too small a value could hinder access
to session blocks and cause an increase in NLDM initialization time and CPU
usage. A value in excess of the number of LUs can result in unused virtual storage
but may improve access to session blocks. It is better to over estimate LUCOUNT
than under estimate. For every 250 LUs specified, the search tables require 4K
virtual storage.
You can use the SESSMDIS command to determine a value for LU COUNT. Summing the SSCP-LU and LU-LU session counts should provide a conservative estimate.
Chapter 7. Tuning for the NetView Management Console

The NetView management console (NMC) uses interactive graphics to display pictures, or views, that represent a network, a portion of a network, or a group of networks at various levels of detail. These views show the network resources and systems that you are monitoring. When you monitor a network, resource status changes are reflected graphically in the views.

NetView management console consists of a two tier client-server relationship. The NMC console and the NMC server run on the platforms shown in Table 6.

Table 6. NMC topology server/NMC topology console platforms

<table>
<thead>
<tr>
<th>NMC Topology Console</th>
<th>NMC Topology Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Windows NT®</td>
<td>• Windows NT</td>
</tr>
<tr>
<td>• Windows® 2000</td>
<td>• Windows 2000</td>
</tr>
<tr>
<td>• AIX®</td>
<td>• AIX</td>
</tr>
<tr>
<td>• Solaris</td>
<td>• SuSE Linux for S/390®</td>
</tr>
<tr>
<td>• HP/UX</td>
<td></td>
</tr>
<tr>
<td>• Red Hat Linux (Intel)</td>
<td></td>
</tr>
<tr>
<td>• SuSE Linux Professional</td>
<td></td>
</tr>
</tbody>
</table>

Client and server code is installed on each workstation and runs as a Java™ application, displaying data from RODM. The server portion of the workstation code, in turn, acts as a client requesting data from the host NetView program. NMC was adapted from the graphic data server, therefore, most of the GMFHS code is unaffected.

Because NMC function is accomplished through cooperative processing between the host and the programmable workstation components, this section describes tuning techniques for each component. For more information about host tuning techniques, see “Host Tuning Techniques” on page 86.

Workstation Tuning Techniques

This section describes tuning techniques for the NetView management console that can improve performance at the workstation. These tuning techniques are arranged in order of expected effect on performance, with the most important tuning considerations listed first.

1. Determine the amount of working set storage required for the NetView management console on the workstation. See “Storage Estimates” on page 83.
2. Use powerful client workstations if there is a significant amount of view update activity and a large total number of resources are displayed in the active views. See “Hardware Requirements” on page 83 and “Client Performance” on page 84.
3. Limit the background pictures used in client workstation views. See “Using Background Pictures” on page 85.
4. Use high-speed connections instead of low-speed connections between the status focal point and the server workstation. See “Status Focal Point to Programmable Workstation Connectivity” on page 85.
5. **Take advantage of the server-client configuration features.** See “Server-Client Configurations” on page 85.

If you are experiencing delays when views are being built on the workstation, consider the following:

- When running on a Windows NT workstation, bring up Task Manager and select the Performance window. This window will show storage and CPU usage for the workstation. Other platforms have similar tools (TOP for SUN, etc.). This can be helpful in determining whether data is being sent or processed in the event of a potential suspended (hung) situation.
- Be careful when clicking on a view. Watch the spinning icon on the top right of the NMC Client window. When it is spinning, data is being requested/processed.
- Do not click on another view until the previously selected view is completely displayed. Doing so causes NMC to suspend or possibly even abend.
- If a view seems to be suspended and is not displayed, use the exit door to logoff of the NMC Client window. When it is spinning, data is being requested/processed.
- If a view seems to be suspended and is not displayed, use the exit door to logoff of the NMC Client window. Check the client, server, and NetView logs for an indication of a potential problem. If no error messages are found, attempt logging back on to the client.
- Whenever possible, try to not have background views open. Status changes are dynamic in NMC. If multiple views are open, each one will be checked to determine if it needs to be updated. To close background views, right-click on the background view and select the **Close** option.
- Monitor all of the clients on each server. Make sure that clients logoff when not using NMC and at the end of each day. Failure to do this results in excessive network traffic and potential network performance problems.
- Beginning with Tivoli NetView for OS/390 V1R3, the NMC code is shipped with the required level of Java. Java RunTime Environment 1.1.7 (for example, jre 1.1.7) is used rather than a Java Development Kit (jdk). This reduces installation and setup time by removing the required customization associated with installing a jdk.
- It is recommended to have only one component of the NMC code client or server active on a workstation at a time to maintain stability and performance.
- If Task Manager on the client shows excessive CPU usage due to Java and NMC, consider upgrading the client workstations CPU to Pentium® II 400Mhz or higher.

In addition to the above tuning recommendations, there are several operands that can affect NMC performance.

In the server.properties file shipped with NMC, changing the following values will affect performance:

- **statusUpdateAndViewChangeTicker** — Default is 1000 (1 second).
- **statusUpdateInterval** — Default is 1.
- **statusFlushCount** — Default is 15.
- **viewChangeInterval** — Default is 3.
- **viewChangeFlushCount** — Default is 10.

The above values should all be lowered simultaneously. The new values will be picked up when the NMC server is recycled. Monitor NMC performance after making the changes.
Storage Estimates
To ensure optimum performance of the workstation-based components of the NetView management console, minimize the amount of paging and swapping that can occur. You can do this by providing as much memory as will be used on a regular and consistent basis. There must be enough workstation memory to consistently contain the working set. If severe memory constraints exist, abends can occur.

Hardware Requirements
This section lists hardware requirements for the following environments:

- "Intel Platform Workstations"
- "UNIX® Platform Workstations"
- "Servers and Consoles" on page 84

Intel Platform Workstations
The following are the minimum requirements beyond those required for Intel platform workstations:

- Pentium family or later processor that can achieve a SPECInt95 score of at least 13.5
  For example, a Pentium II, 400Mhz machine, with 512KB L2 cache qualifies.

  Note: Most manufacturers provide the SPECInt95 scores for each of their processors. A substantial list of SPECInt95 scores (for machines from a variety of manufacturers) is available on the web at:
  http://www.spec.org/osg/cpu95/results/cint95.html

- 256 MB of memory (RAM)
- 15 MB of additional fixed disk space for the NetView management console
- Screen resolution of 1024 by 768, with a maximum of 256 colors recommended for the color palette

  Note: Higher resolution and a larger color palette than recommended can severely degrade performance.

UNIX® Platform Workstations
The following are the minimum requirements beyond those required for UNIX platform workstations:

- Any processor that is capable of achieving a SPECInt95 score of at least 13.5
  For example, an RS6000 43P-150 machine, with at least a 375Mhz PowerPC® 604e and at least 1MB L2 cache qualifies.

  Note: Most manufacturers provide the SPECInt95 scores for each of their processors. A substantial list of SPECInt95 scores (for machines from a variety of manufacturers) is available on the web at:
  http://www.spec.org/osg/cpu95/results/cint95.html

- 256 MB of memory (RAM)
- 15 MB of additional fixed disk space for the NetView management console

  Note: For LINUX on z/Series requirements, please refer to the README file on the CD shipped with the NMC code.
• Screen resolution of 1024 by 768, with a maximum of 256 colors recommended for the color palette

Note: Higher resolution and a larger color palette than recommended can severely degrade performance.

Servers and Consoles
The following are additional requirements beyond the minimum requirements for Intel or UNIX platform workstations:

• 256 MB (RAM) for each console instance
• Between 256 and 384 MB of swap/page space
The formula for calculating each console instance is: 1.5 to 2.5 times the amount of RAM for systems with 256 MB (or more).

Notes:
1. If you have less than the minimum amount of RAM, you could easily have situations where you will page heavily. Not only will this degrade performance, but you will probably require more page/swap space for this activity. If this occurs, add another 33% to the combined total of memory and swap/page space.
2. If you run other applications on these systems, increase the amount of installed RAM and page/swap space accordingly to provide room for the additional applications to work without adversely affecting the NetView management console.
3. For LINUX on z/Series requirements, refer to the README file on the CD shipped with the NMC code.

Client Performance
If many resources are in the views displayed at a client workstation, consider using a high-performance processor for this client workstation. The activity of drawing and redrawing the views can take considerable amounts of processor resources. This resource use can be particularly high when view navigation is necessary during the arrival and display of a large number of resource status updates. This situation is likely when a large burst of status changes occurs, such as during network activation.

Ensure that the minimum hardware requirements, as outlined in “Hardware Requirements” on page 83 and in the Tivoli NetView for z/OS Program Directory, are met. If the CPU and storage requirements are not met, then performance degradation will occur. Depending on the number of objects in a view, NMC can use 100% of the CPU on a Pentium II 233Mhz workstation. While 100% CPU may be maintained for 5 to 10 seconds on a Pentium II 233Mhz, slower CPUs will obviously remain 100% busy for longer periods of time.

To minimize the number of resources actively displayed at a client workstation, use views containing aggregate resources that represent the real resources you are monitoring. From aggregate resources, you can navigate to views containing failing resources by using the fast path to failing resource feature. You can customize the effect real resources have on an aggregate resource. Refer to the Tivoli NetView for z/OS NetView Management Console User’s Guide for information about view navigation and adjusting aggregation for a resource.

Closing views that you are no longer using helps to ensure that client workstation resources are available to handle a burst of client activity.
If you use the Cycle Windows window to cycle through views you have open for monitoring, lower values for the cycle delay interval will result in higher view drawing activity.

**Using Background Pictures**

Adding a background picture to a view can impair client workstation performance. Including a background picture can increase response times for the initial view display and whenever the view is redrawn. The impaired performance is caused by the storage requirements of the background picture and the CPU resources needed to draw it.

The CPU and memory requirements vary depending on the detail and complexity of the background picture. If you are using detailed or complex background pictures, increased storage requirements can increase response times for any other activity on the client workstation if these requirements cause your working set storage requirements to exceed the amount of available memory.

**Status Focal Point to Programmable Workstation Connectivity**

The speed of the connections from the status focal point to the server workstation has a large impact on NetView management console performance for large bursts of updates.

Processing of bursts generated by network failures is affected by link speed more than is processing of bursts generated by network recovery.

These two factors combined cause the number of status changes per unit of time to be greater for network failure than for recovery. Because of these factors, the larger throughput provided by higher-speed connections is of more benefit during network failures.

If possible, use token-ring connections from the status focal point to the server workstation. Token-ring connections are the most desirable because of their higher transmission speed capability.

**Server-Client Configurations**

Because the IP communication feature is being used as the communication vehicle for the NetView management console, you can connect console workstations with the server workstation in a number of ways. For example, you can have the server workstation attached to a token ring and in session with console workstations on the same ring. The client workstation can be anywhere that an IP session can be established. Also, all NetView management console workstations can be combined server-console workstations.

When you select the workstation server-console configuration, several factors can affect the overall performance. The server workstation maintains an in-storage database of the current network resource status for opened views. The activity associated with resource status changes is both CPU and memory intensive. The server workstation should be used as a stand-alone server workstation. Do not monitor views that can be subject to continuous or large bursts of status update activity because updating the graphical views is CPU intensive. Also, ensure that other applications that might be running on the server and console workstation are not CPU and memory intensive.
Host Tuning Techniques

This section describes tuning techniques for the NetView management console at the host.

NETCONV

The NETCONV command establishes an LU 6.2 communication session between the status focal point host and server workstation. When this session is established, the status focal point forwards the current status of all monitored resources to the server workstation.

If the NETCONV command is issued before the network is activated, the focal point forwards the initial status (never active) of all the monitored resources to the server workstation. When the network is activated, the current status of all monitored resources is also forwarded to the workstation.

Issue the NETCONV command after network activation to avoid processing multiple status updates at network activation time and reduce elapsed time for status display.

Refer to Tivoli NetView for z/OS Command Reference for more information about using the NETCONV command.

DUIGINIT Parameters

The following parameters in DUIGINIT (the GMFHS initialization member) affect NMC performance:

- LCON-STATUS-DELAY-MAX (The default value is 10.)
- LCON-STATUS-DELAY-TIME (The default value is 50.)
- LCON-EVCHANGE-BUFFER-INTERVAL (The default value is 500 [5 seconds]).
- LCON-AGG-BUNDLE-INTERVAL (The default is 500 [5 seconds]).

Use the following procedure to change the parameters:

- Reduce all parameters simultaneously
- Recycle GMFHS
- Monitor NMC performance to determine if the changes have improved NMC performance.

Refer to the Tivoli NetView for z/OS Administration Reference for more information about these parameters.
Chapter 8. Tuning for the Resource Object Data Manager

The NetView Resource Object Data Manager (RODM) is a data cache that is designed to store network configuration and status information about system resources. RODM enables you to automate network management functions associated with the resources defined to RODM. In addition, you can write RODM applications to perform other network management and automation tasks.

For more information on the object-oriented terms used by the NetView program to describe RODM and its data model, refer to the Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer’s Guide.

Tuning Techniques

The following major tuning techniques for RODM are described in this chapter:

1. Keep RODM logging to a minimum for production systems by using LOG_LEVEL 8. See “Customization Parameters” on page 95.
2. See “Estimating Storage Usage” on page 135 to determine virtual storage use and DASD space requirements for the checkpoint data sets. See “RODM Data Sets” on page 88. Ensure that the MVS system has an adequate paging system. If your system is storage-constrained, consider moving workloads to other systems or enhancing the MVS paging system.
3. Checkpoint RODM whenever there have been significant changes to the structure or topology of the objects in RODM, such as after loading objects with the loader facility. Do not use checkpoints to capture the status of objects in RODM. When a RODM warm start is performed, the RODM applications which created the objects should update the object status when the application is initialized after the RODM warm start. A warm start is relatively fast. See “Warm Start and CHKPT Commands”.

Note: If your only RODM applications are MSM and SNATM, do not use checkpoints.
4. Generate RODM API statistics to analyze the content and activity of RODM. See “RODM API Statistics” on page 89.
5. Generate RODM cell pool statistics to analyze the storage usage of RODM. See “RODM Cell Pool Statistics” on page 91.
6. Specify a minimum number of CONCURRENT_USERS. Extra storage is required for each user. See “Customization Parameters” on page 95.
7. Run RODM at the same dispatching priority as the NetView program.
8. If you will be writing a RODM application, see “Programming Recommendations” on page 95.

Warm Start and CHKPT Commands

A warm start takes significantly less time than a cold start. The actual data still resides on the data-in-virtual (DIV) checkpoint data set and is brought into extended storage (in 4 KB pages) as the data is referenced. The warm start provides only an in-storage map of the DIV data set.

Take a checkpoint of RODM when there have been significant changes to the structure or topology of the objects in RODM, such as after loading objects with
the loader facility. The checkpoint provides a map of the latest changes to the
VSAM linear data sets (LDSs). Do not use checkpoints to capture the status of
objects in RODM. When a RODM warm start is performed, the RODM
applications that created the objects should update the object status when the
application is initialized.

Note: If your only RODM applications are MSM and SNATM, do not use
checkpoints. If you do not use the RODM checkpoint function, you can
disable it and avoid allocating the checkpoint data sets described in "RODM
Data Sets". To disable the checkpoint function, comment out the DD
statements for EKGMST, EKGTRAN, and EKGD00X in your copy of the
RODM startup procedure (EKGXRODM).

RODM Data Sets

There are four data definition names in the supplied RODM JCL sample
EKGXRODM that define the RODM checkpoint VSAM linear data sets (LDSs). The
data definition names are:

EKGMST
The data set for the master window. It is recommended that you use the
system defaults for the master window size, which is 4 cylinders.

EKGTRAN
This is the checkpoint data set for the segment window storage. The
recommended allocation is 96 cylinders, which is the default.

EKGD001 and EKGD002
The checkpoint data sets for data window (checkpoint) storage. Allocate
sufficient checkpoint storage to contain all data model storage, plus extra
storage in case your system has a burst of activity that overruns the
current allocation. The default space allocation is 72 cylinders.

Note: See "Estimating Storage Usage" on page 135 to determine space allocations
for these data sets. If you are migrating from an earlier release of NetView, it
is likely that you will need to increase your checkpoint data set allocations.

The sample VSAM LDS definitions are in member EKGDWIND.

You can use multiple checkpoint data sets to minimize the average size needed
and provide for increased efficiency during the checkpoint. You can define
additional checkpoint LDSs by adding data definition names in the EKGDWIND
member (such as EKGD003 and EKGD004).

Note: If your only RODM applications are MSM and SNATM, do not use
checkpoints. If you do not use the RODM checkpoint function, you can
disable it and avoid allocating the checkpoint data sets. To disable the
checkpoint function, comment out the DD statements for EKGMST,
EKGTRAN, and EKGD00X in your copy of the RODM startup procedure
(EKGXRODM).

Error Messages EKG1110I and EKG1111I

If you receive error message EKG1110I, increase the size of the EKGTRAN data set
for segment window storage. Because you can define only one LDS for the
segment window allocation, a cold start is required.
If you receive error message EKG1111I, increase your space allocation for the EKGD001 and EKGD002 data sets, or add VSAM checkpoint LDSs (such as EKGD003 and EKGD004).

- Error message EKG1111I is issued even though you are not taking a checkpoint, when the size of your data model (classes and objects) grows beyond the size of the data windows present in your checkpoint data sets, another data window cannot be allocated. RODM continues to operate until the last data window is filled.
- You can take a checkpoint and warm start RODM when adding a checkpoint LDS.
- If the existing LDSs are reallocated with additional space, you must cold start RODM.

RODM API Statistics

RODM API statistics enable you to analyze the content and activity of RODM. Use the STATAPI parameter, of the MVS MODIFY command to generate API statistics in a type 8 log record and the CLEAR option to clear the counters.

RODM API statistics give you the number of times a call has been made or a method triggered since RODM was cold started and since the counters were cleared using the MVS MODIFY rodmname,STATAPI,CLEAR command for the following categories:

- User API (UAPI) calls
- Method API (MAPI) calls
- Object-specific (OS) methods triggered by RODM
- Object-independent (OI) methods triggered by users of RODM

RODM API statistics also specify the number of times a call was a success or failure. These statistics also specify the type of method:

- Change
- Notify
- Query

Figure 19 on page 90 is an example of the output from the RODM log formatter for log record type 8 API statistics.
Log_type : 8 (Statistics)  RBA : 6213116
Record number : 31277  Record Length : 1324
Transaction ID: 000000000000000Ox  Timestamp : Sun Mar 30 11:33:30 2000
User Appl ID :
API Version : 1
Stat Type : 5 (API Statistics)
Last Clear Timestamp : Sat Mar 29 19:59:03 1999
Output Timestamp : Sun Mar 30 11:33:31 1999
No. of Query Triggered : 0
No. of Change Triggered : 18140
No. of Notify Triggered : 2273
No. of Objdel Triggered : 0
No. of Permanent Entries: 14

Permanent Count Data :
   Function ID : 1302 (Create a Class)
        Perm UAPI Count : 0000000000000044x
        Perm MAPI Count : 00000000000001Ex
   Function ID : 1304 (Create a Field)
        Perm UAPI Count : 00000000000006F6x
        Perm MAPI Count : 00000000000002Cx
   Function ID : 1306 (Create a Subfield)
        Perm UAPI Count : 0000000000001B67x
        Perm MAPI Count : 0000000000000252x
   Function ID : 1405 (Link 2 Objects - Methods Triggered)
        Perm UAPI Count : 0000000000000A91x
        Perm MAPI Count : 0000000000005D0Ex
   Function ID : 1407 (Unlink 2 Objects - Methods Triggered)
        Perm UAPI Count : 0000000000000801x
        Perm MAPI Count : 000000000000006Dx
   Function ID : 1409 (Create an Object)
        Perm UAPI Count : 00000000000030ACx
        Perm MAPI Count : 0000000000000267x
   Function ID : 1410 (Delete an Object)
        Perm UAPI Count : 0000000000000126x
        Perm MAPI Count : 000000000000147Fx
   Function ID : 1412 (Add Notification Subscription)
        Perm UAPI Count : 0000000000000003x
        Perm MAPI Count : 0000000000000147Fx
   Function ID : 1413 (Delete Notification Subscription)
        Perm UAPI Count : 0000000000000004CCx

Figure 19. RODM Log Record Type 8 for API Statistics (Part 1 of 2)
If you specify the CLEAR option when writing the API statistics, the regular count data counters are reset to zero after being written to the log (the permanent count data counters are not affected by the CLEAR option). You may find it convenient to write the API statistics on a timer basis using the NetView EVERY command. If you use the CLEAR option to clear the counters each time you write them, the counters will show which API calls were made during the timer interval. Clearing the counters periodically also ensures that the counters do not overflow.

Refer to Tivoli NetView for z/OS Diagnosis Guide for a description of all of the fields in RODM log record type 8.

RODM Cell Pool Statistics

You can use RODM cell pool statistics to determine whether RODM is using storage efficiently. To access these statistics, use the MVS MODIFY command with the STATCELL parameter (MVS MODIFY rodmname,STATCELL) to write cell pool use information to a type 8 RODM log record. See Tivoli NetView for z/OS Diagnosis Guide for a description of all of the fields in RODM log record type 8.
Figure 20 shows an example of the output from the RODM log formatter for log record type 8 segment and window statistics.

Log type : 8 (Statistics)  RBA : 6211080
Record number : 31276  Record Length : 2036
Transaction ID: 0000000000000000x  Timestamp : Sun Mar 30 11:33:31 1999
User Appl ID :
API Version : 1
Stat Type : 1 (Window Statistics)
Current pocket: 1
Avail. pocket : 1
No. of Entries: 33

Cell Size ( 0): 8  Pool Size : 1
No. in Use : 154233  High Water Mrk: 156185
In Use Percent: 20  Total Inuse % : 20
High Water % : 20
Histogram Data :
( 0) 0 ( 1) 0 ( 2) 0 ( 3) 0
( 4) 0 ( 5) 78448 ( 6) 19212 ( 7) 69835

Cell Size ( 1): 12  Pool Size : 1
No. in Use : 167074  High Water Mrk: 167074
In Use Percent: 22  Total Inuse % : 22
High Water % : 22
Histogram Data :
( 0) 0 ( 1) 1147 ( 2) 0 ( 3) 8285
( 4) 0 ( 5) 0 ( 6) 0 ( 7) 161887

Cell Size ( 2): 16  Pool Size : 1
No. in Use : 35885  High Water Mrk: 35885
In Use Percent: 4  Total Inuse % : 4
High Water % : 4
Histogram Data :
( 0) 0 ( 1) 964 ( 2) 0 ( 3) 16281
( 4) 0 ( 5) 759 ( 6) 0 ( 7) 26268

Cell Size ( 3): 24  Pool Size : 1
No. in Use : 27278  High Water Mrk: 27279
In Use Percent: 3  Total Inuse % : 3
High Water % : 3
Histogram Data :
( 0) 8 ( 1) 41631 ( 2) 1844 ( 3) 7
( 4) 110 ( 5) 11 ( 6) 768 ( 7) 1048

Cell Size ( 4): 32  Pool Size : 1
No. in Use : 1801  High Water Mrk: 1802
In Use Percent: 0  Total Inuse % : 0
High Water % : 0
Histogram Data :
( 0) 12 ( 1) 43 ( 2) 130 ( 3) 15305
( 4) 14 ( 5) 4 ( 6) 691 ( 7) 14015


Figure 20. RODM Log Record Type 8 for Segment and Window Statistics

The following describe the segment and window statistic fields in log record type 8, which is shown in Figure 20.

NO. OF ENTRIES
Specifies the number of entries in the cell pool array.

CELL SIZE
Specifies the cell size in bytes as defined in member EKGCUST.

POOL SIZE
Specifies the number of 4 KB pages that are allocated when a pool extension is needed (defined in member EKGCUST).
NO. IN USE
Specifies the number of cells that are unavailable.

HIGH WATER MRK
Specifies the high-water mark for in-use cells.

IN USE PERCENT
Specifies the percentage of in-use cells.

TOTAL INUSE %
Specifies the percentage of total cells in use.

HIGH-WATER %
Specifies the percentage for the high-water mark.

HISTOGRAM DATA
Lists eight counters associated with the cell size. These counters are used to show the distribution of storage size requests satisfied by the cell pool.

To evaluate the amount of window storage currently in use, multiply the number of cells in use (NO. IN USE) by the cell size for each of the entries in the cell pool array. The sum of the products is the amount of RODM data space storage currently in use.

Using the Histogram Data
Provisions for monitoring the allocation requests and the usage of data space storage are built into the RODM storage manager. As each request for storage is received and matched to a cell, an additional calculation is made. A table is allocated at initialization that contains the following information for each cell pool defined to the system:

- The cell size in bytes
- The pool size in pages
- The number of cells available
- The number of cells in use
- The high-water mark of cells in use
- A set of eight counters to reflect the approximate size of the actual storage request (plus 4 bytes of control information) in histogram format

Figure 21 is an example of histogram data.

<table>
<thead>
<tr>
<th>Cell Size (6): 48</th>
<th>Pool Size : 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Size (7): 64</td>
<td>Pool Size : 1</td>
</tr>
<tr>
<td>No. in Use : 4817</td>
<td>High Water Mrk: 4817</td>
</tr>
<tr>
<td>In Use Percent: 2</td>
<td>Total Inuse % : 2</td>
</tr>
<tr>
<td>High Water % : 2</td>
<td>Histogram Data :</td>
</tr>
<tr>
<td>(0) 0</td>
<td>(1) 0</td>
</tr>
<tr>
<td>(4) 6</td>
<td>(5) 4813</td>
</tr>
</tbody>
</table>

The histogram data can be used to tune the customizable cell pool sizes. To evaluate the histogram data for Figure 21.
1. Subtract the previous cell size from the current cell size (64 - 48 = 16). This is the size of the range of storage requests that are serviced by this cell pool.

2. Divide the result by 8 (16 / 8 = 2). This is the size of the range of storage requests for each of the eight counters in the histogram data.

3. Add 1 to the position value (0–7 becomes 1–8).

4. Multiply each position in the histogram by the result from 2:

   \[
   \begin{align*}
   1 \times 2 &= 2 \\
   2 \times 2 &= 4 \\
   3 \times 2 &= 6 \\
   4 \times 2 &= 8 \\
   5 \times 2 &= 10 \\
   6 \times 2 &= 12 \\
   7 \times 2 &= 14 \\
   8 \times 2 &= 16 \\
   \end{align*}
   \]

5. Add these results to the previous cell size (48) to get the maximum storage request size counted in each histogram position:

<table>
<thead>
<tr>
<th>Position</th>
<th>(0)</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increment</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>Size</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>56</td>
<td>58</td>
<td>60</td>
<td>62</td>
<td>64</td>
</tr>
</tbody>
</table>

The following results are from Figure 21 on page 93:

- **Size Requested**
  - Greater than 48 but less than or equal to 50: 0
  - Greater than 50 but less than or equal to 52: 0
  - Greater than 52 but less than or equal to 54: 0
  - Greater than 54 but less than or equal to 56: 0
  - Greater than 56 but less than or equal to 58: 6
  - Greater than 58 but less than or equal to 60: 4813
  - Greater than 60 but less than or equal to 62: 0
  - Greater than 62 but less than or equal to 64: 0

The results show that 4813 requests were for storage greater than 58 bytes but less than or equal to 60 bytes (59 or 60 bytes). If a cell size of 60 had been defined to RODM, 4 bytes times 4813 requests, or 19252 bytes, would have been saved.

Any additions or modifications you make to the cell pool sizes should be specified in the CELL_POOL definitions in EKGCUST. All cell sizes must be fullwords. Consider how many cells fit on a page, because all pool sizes are rounded up to the next page boundary.
Customization Parameters

The defaults supplied with the NetView program in EKGCUST are adequate for most systems. However, you can alter the following settings to meet your specific requirements.

- The number of CONCURRENT_USERS is initially set to 10. You may need to increase this value, but do not make it unnecessarily high because extra storage is required for each user.
- The number of ASYNC_TASKS is initially set to 5. To save storage, you may decrease this value to 2.
- LOG_LEVEL is initially set to 8 (record errors only). This is recommended for a production environment where more than one RODM API call a second is anticipated. If the log levels were changed on a test system for debugging purposes, consider changing LOG_LEVEL and MLOG_LEVEL back to 8 when switching to production.
- Method tracing should be done only for problem solving, not during production. Set MTRACE_TYPE to X’00000000’ to disable method tracing.

Programming Recommendations

This section contains programming recommendations for those writing RODM applications.

- Keep the number of RODM API calls to a minimum.
- When possible, use the following to combine multiple operations into a single API call:
  - Query multiple subfields (function ID 1508)
  - Change multiple subfields (function ID 1419)
  - Execute a list of functions (function ID 1600)
- Do not specify initial values on loader field definitions if they are not required. These values require extra processing at load time.
- Query fields by ID rather than by name when possible.
- Consider user methods similar to online CICS or IMS™ transactions. Use a minimum of processing and MAPI calls for methods that are likely to be run frequently.
Chapter 9. Tuning for VSAM

Input/output (I/O), specifically direct access storage device (DASD) I/O, is a major concern of performance and tuning, especially in a NetView environment. For databases, the NetView program uses VSAM data sets. The NetView program records messages to a log data set. The NetView program puts session data into a session monitor database, and network events and statistics into a hardware monitor database.

Tuning Techniques

Following are the major VSAM tuning techniques, arranged in order of expected effect on performance, with the most important tuning considerations listed first. These recommendations are described in detail in this chapter.

1. Use the CISIZE values that are used in the sample cluster definitions for best performance. Do not use the same CISIZE values for the hardware monitor and session monitor databases.

2. Start with the default LSR buffer pool allocations, and monitor the buffer miss percentage with the VSAMPOOL command. Increase the number of buffers for individual pools where needed, and reduce the number of buffers for pools that are not used frequently. See "Local Shared Resources (LSR) and Deferred Write (DFR) and "VSAMPOOL Command" on page 103.

3. Use the DBAUTO command to reorganize databases that are not deleted and redefined regularly. See "VSAM Database Maintenance" on page 105.

4. Consider using the deferred write (DFR) performance option for the hardware monitor and 4700 support facility databases to reduce I/O activity. The sample definitions for the session monitor database have DFR specified already. See "Local Shared Resources (LSR) and Deferred Write (DFR)"


Local Shared Resources (LSR) and Deferred Write (DFR)

The VSAM performance options of local shared resources (LSR) and deferred write (DFR) provide major improvements in VSAM database processing. LSR enables the sharing of common control blocks (I/O control blocks, I/O buffers, and channel programs). On GET requests, buffers are searched for direct record retrievals. Without LSR, VSAM performs I/O for direct retrievals regardless of whether the control interval (CI) containing the desired record is in storage.

DFR causes VSAM to defer the write I/O when records are directly inserted or replaced in direct mode. Without DFR, VSAM does not defer the I/O for direct inserts or replacement of records. With DFR, the buffers are written in these instances:

- No more buffers are available to perform a retrieve.
- The application issues the WRTBFR macro indicating that VSAM should write out the modified buffers.
- The database is closed.

If the NetView program ends without closing the databases, the records in the DFR buffers are not written to the databases.
If you specify DFR, you get both the LSR and DFR options.

Do not cancel the NetView program except as a last resort. If you must issue a FORCE command, try to close the databases by issuing the NetView SWITCH command with the T option. This closes the active database and does not perform a switch. If this procedure does not work, issue the NetView STOP FORCE command for each active VSAM task. If you must use the MVS FORCE command to bring down the NetView program and you have specified DFR, you might have to delete and redefine the affected databases. The exposure of having records not written to the databases is minimized by the extended specify task abnormal exits (ESTAEs) that trap abends and close the databases. However, if the system operator ends the NetView program with the MVS FORCE command, the ESTAEs are not driven.

**Definitions for LSR and DFR**

To define LSR and DFR values, code the DSTINIT statement shown in the following example in the NetView initialization member for each data services task that uses VSAM.

```
DSTINIT MACRF=xxx
```

In this example, `xxx` is either LSR or DFR.

Table 7 lists the MACRF and CISIZE values for the NetView components and facilities that appear in the sample definitions.

<table>
<thead>
<tr>
<th>Component</th>
<th>Member</th>
<th>MACRF</th>
<th>3390 DASD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Index CISIZE</td>
</tr>
<tr>
<td>Central site control</td>
<td>DSIKINIT</td>
<td>LSR</td>
<td>2048</td>
</tr>
<tr>
<td>facility</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware monitor</td>
<td>CNMSTYLE</td>
<td>LSR</td>
<td>2560</td>
</tr>
<tr>
<td>Network log</td>
<td>n/a</td>
<td></td>
<td>1024</td>
</tr>
<tr>
<td>Trace log</td>
<td>DSITRCBK</td>
<td>LSR</td>
<td>512</td>
</tr>
<tr>
<td>Save/Restore</td>
<td>DSISVRTD</td>
<td>LSR</td>
<td>4096</td>
</tr>
<tr>
<td>Session monitor</td>
<td>AAUPRMLP</td>
<td>DFR</td>
<td>1536</td>
</tr>
<tr>
<td>4700 support facility</td>
<td>BNJ36DST</td>
<td>LSR</td>
<td>3072</td>
</tr>
</tbody>
</table>

**Notes:**

1. The CISIZE values specified for the data and index components in the samples shipped with the NetView program are based on using an IBM 3390 (using ICF catalogs). Because of the higher capacity of the 3390, different data buffer selections were made. These new selections result in a new index control interval size for one cluster. If other types of devices are used to allocate these clusters, these operands might need to be adjusted for optimal use of the device.

   LSR is the default for the hardware monitor and 4700 Support Facility databases. Consider using DFR for these databases, because DFR gives better performance. LSR is used in the samples because some environments cannot tolerate losing records in the hardware monitor database, even though the possibility is remote.
Neither LSR nor DFR is recommended for the network log. The DSILog task buffers records before writing them to DASD. Log browse does not work if LSR or DFR is used for the network log.

**Buffer Pool Sizes**

The VSAM buffer pool allocation is defined in module DSIZVLSR. The VSAM BLDVRP (build VSAM resource pool) macro creates the DSIZVLSR module. Figure 22 shows the sample buffer pool allocations for MVS sample member CNMSJM01.

```
DSIZVLSR CSECT BLDVRP
| BUFFERS=(7168(4), 8192(20), 16384(4), 18432(20), 20480(20), 24576(20)), |
| KEYLEN=96, MF=L, MODE=24, RMODE31=BUFF, SHRPOOL=0, STRNO=30, TYPE=(LSR,DATA) |

BLDVRP
| BUFFERS=(512(3), 1536(30), 2048(30), 2560(30), 3072(10), 4096(30)), |
| KEYLEN=96, MF=L, MODE=24, RMODE31=BUFF, SHRPOOL=0, TYPE=(LSR,INDEX) |

END
```

---

**Figure 22. Sample BLDVRP Macros Defining VSAM Buffer Pools for CNMSJM01**

**Note:**

The buffer sizes in DSIZVLSR above correspond to the default index and data CICSIZE values, which are based on using 3390 DASD (using ICF catalogs).

If there are buffer sizes defined in DSIZVLSR that none of your databases use, remove them to decrease storage usage. Use the VSAMPOOL command to monitor usage of the LSR buffer pools. See “VSAMPOOL Command” on page 103.

**The KEYLEN Parameter**

The KEYLEN parameter specifies the maximum key length of the data sets that share this pool. This keyword should be specified only on the data pool.

**The STRNO Parameter**

The STRNO parameter specifies the potential number of requests, in the range of 1–255, that can be issued concurrently for all the data sets sharing the resource.
pool. Set STRNO to the total of all the DSRBO values for data services tasks that use the LSR resource pool. The sample value of 30 is sufficient for most environments. If you modify the DSRBO values for some of the DSTs, ensure that you adjust the STRNO parameter accordingly.

The BUFFERS Parameter
The BUFFERS parameter specifies the size and number of buffers in each buffer pool in the resource pool. When you open a database and specify LSR or DFR, VSAM looks for a buffer pool for the INDEX and DATA components, depending on their control interval sizes. A buffer pool that is the same size as the control interval is chosen. If a buffer pool with the same size has not been defined, the next higher buffer pool size is chosen. Databases with the same control interval sizes share the same buffer pool. Allocate enough buffers of a particular size to satisfy all DSTs sharing the buffer pool.

The BUFFERS parameter is specified with values that separate the index and data control intervals into separate pools. Having separate index and data pools allows the critical index records to remain resident in memory without the need to allocate an excessive number of buffers.

The size of the LSR buffer pool allocation affects VSAM performance considerably. If you want to change the buffer pool allocation, you can do so by modifying the sample definition and running the VSAM BLDVRP macro to create a new DSIZVLSR module. Refer to Tivoli NetView for z/OS Installation: Configuring Additional Components for more information about running the VSAM BLDVRP macro.

Buffer Pool Size Recommendations
Consider the following tuning recommendations in determining the buffer pool sizes for your environment.

- Start with the default number of buffers for each buffer size in DSIZVLSR. See Figure 22 on page 99.
- Monitor the buffer usage with the VSAMPOOL command.
  - Remove buffer sizes that are never used to save storage.
  - For buffer sizes that are used infrequently, consider reducing the number of buffers to save storage.
  - For index buffers that are used frequently, the number of buffer finds (BFRFND) should be at least 10 to 20 times the number of buffer reads (BUFRDS). If this is not the case, consider increasing the number of buffers to reduce I/O activity.
  - For data buffers that are used frequently, the number of buffer finds should be greater than the number of buffer reads. If this is not the case, consider increasing the number of buffers.
  - When modifying the buffer allocations, monitor the buffer usage before and after making changes. If an increase does not result in improved performance, reduce the buffer allocation to its previous value.

See “VSAMPOOL Command” on page 103 for information about the VSAMPOOL command and for more information on tuning LSR buffer allocations.

Allocating the Buffer Pools in an MVS Hyperspace
For MVS systems, you can allocate the VSAM buffer pools in a hyperspace. Specifying hyperspace buffers is a method of reducing I/O to DASD by caching data in expanded storage. When you access the buffer pool, the page must first be
moved from expanded storage to central storage. The alternative is to let the 
operating system storage manager determine the location of pages using its page 
replacement strategy. If your system central storage is constrained, allocating the 
buffer pool in a hyperspace might help relieve this contention.

To use hyperspace buffers, the buffer sizes must be in multiples of 4096. Therefore, 
change the corresponding CISIZE values for your databases to be multiples of 
4096. Round the sizes up to the nearest multiple of 4096. For example, if the data 
CISIZE of a database is 22528, increase the CISIZE to 24576.

Note: *Specifying hyperspace buffers requires a modification to the VSAM 
BLDVRP macro (which is used to create module DSIZVLSR). See the 
appropriate MVS publication for a description of the syntax of the BLDVRP 
macro.

Monitoring VSAM Performance

NetView provides the LISTCAT and VSAMPOOL commands to assist in assessing 
the performance of VSAM. VSAMPOOL is available on MVS systems only.

LISTCAT Command

The LISTCAT command displays VSAM database definition and performance data 
for NetView data services tasks that have open VSAM databases. The information 
is similar to the data from the access methods services (AMS) LISTCAT command; 
however, the NetView LISTCAT command provides the information online, while 
the VSAM database is active.

The LISTCAT command is useful in tuning the VSAM databases and in validating 
the database definitions. This command is a full-screen command processor. After 
invoking the command, press the ENTER key each time you want updated 
information. The screens are automatically copied to the network log. If LISTCAT 
is run on the primary program operator interface task (PPT) or an autotask, the 
information is sent to the network log and command execution ends. This enables 
LISTCAT to run from a NetView timer command. Figure 23 on page 102 shows a 
sample of output from the LISTCAT command. The Tivoli NetView for z/OS 
Command Reference provides a brief description for each field that is displayed.
The following fields are useful in tuning the VSAM database.

1. **VSAM ACB Options**
   - **NSR**  
     - No LSR and no DFR
   - **LSR**  
     - Local shared resources
   - **DFR**  
     - Local shared resources and deferred write

   Under VSAM ACB options, the performance options of LSR and DFR are shown if they are in use for the data set.

2. **Cluster Information**
   - **STRNO**  
     - Number of VSAM strings currently active
   - **STRMAX**  
     - Maximum number of strings used

   These fields show the VSAM string activity for the data set. The number of VSAM strings (STRNO) is defined on the VSAM BLDVRP macro. See “Definitions for LSR and DFR” on page 98 for more information about the BLDVRP macro. The correct value for STRNO is the sum of the number of DSRBOs for DSTs using the resource pool. Although no tuning is necessary here, these fields show how the high-water mark for concurrent operations (STRMAX) compares to the number of VSAM strings defined (STRNO).

3. **Data Component Information**
   - **NEXT**  
     - The number of extents in the data component.
   - **NCIS**  
     - The number of data control interval splits.
   - **NSSS**  
     - The number of data control area splits.
   - **NEXCP**  
     - The number of EXCP (execute channel program - SVC 0) macros issued by VSAM against the data component.
   - **NLOGR**  
     - The current number of records in the data component.
NRETR
The number of records that have been retrieved from the data
component, whether or not they are for update.

NINSR
The number of records that have been inserted into the data
component before the last record. Records originally loaded and
records added to the end are not included in this statistic.

NUPDR
The number of records that have been retrieved for update and
rewritten. This value does not reflect records that are deleted only,
but a record that is updated and then deleted is counted.

NDELR
The number of records that have been deleted from the data
component.

AVSPAC
The number of bytes available in the data component.

ENDRBA
The number of bytes used in the data component.

HALCRBA
The number of bytes allocated in the data component.

Index Component Information
NEXT  The number of extents in the index component.
NIXL   The number of levels of records in the index.
NEXCP  The number of EXCP (execute channel program - SVC 0) macros
        issued by VSAM against the index component.
NLOGR  The current number of records in the index component.
AVSPAC The number of bytes available in the index component.
ENDRBA The number of bytes used in the index component.
HALCRBA The number of bytes allocated in the index component.

Notes:
1. The logical record counts (NRETR, NINSR, NUPDR, and NDELR) show the
cumulative number of total records retrieved, inserted, updated, and deleted
since the file was last created, indicating the type of data set activity that has
taken place.
2. If the number of levels in the index component (NIXL) is excessive, VSAM
performance suffers. Reorganize the data set so that the number of index levels
is three or fewer.

VSAMPOOL Command
The VSAMPOOL command displays statistics about NetView VSAM resource pool
utilization when the NetView program has been defined to use local shared
resources (LSR) or deferred writing of records (DFR). The LSR resource pool is
subdivided into buffer pools determined by control interval sizes. You define the LSR resource pool and buffer pools with the DSIZVLSR module. See “Definitions for LSR and DFR” on page 98 for more information.

VSAMPOOL lists all of the buffer pools that are using LSR and DFR. The output shows the total usage per control interval size (CINV). The display shows separate statistics for the DATA and INDEX VSAM LSR/DFR buffers that were defined in DSIZVLSR. Figure 24 shows a sample of output from the VSAMPOOL command.

The following information is displayed for each buffer pool (see Figure 24).

- **CINV**  Control interval size (or buffer size) for the buffer pool
- **BUFNO**  The number of buffers in the buffer pool
- **BFRFND**  The number of requests for retrieval that could be satisfied without an I/O operation (the data was found in a buffer)
- **BUFRDS**  The number of reads to bring data into a buffer
- **NUIW**  The number of non-user initiated writes (writes that VSAM was forced to perform because no buffers were available for reading the contents of a control interval)
- **UIW**  The number of user-initiated writes (PUTs not deferred or WRTBFRs)
- **ERCT**  The number of write errors that have occurred

The VSAMPOOL command is useful in tuning the size of the LSR buffer pools.

The most useful statistic is the lookaside hit ratio, which is calculated as follows:

\[
\text{lookaside hit ratio} = \frac{\text{BFRFND}}{\text{BUFRDS}}
\]

Figure 24. Sample Output from the VSAMPOOL Command Using 3390 DASD
The lookaside hit ratio gives an indication of the adequacy of the LSR buffer allocation. The optimal lookaside hit ratio depends on your environment. Consider the following in general:

- For data buffers, the lookaside hit ratio should be 1 or greater; values higher than 5 are unusual for databases with high activity.
- For index buffers, the lookaside hit ratio should be 10 or greater. The number of buffer finds (BFRFND) should be at least 10 times greater than the number of buffer reads (BUFRDS). The higher the lookaside hit ratio, the better.

The following approach is suggested for tuning the allocation for the LSR buffer pools:

- Remove buffer sizes that are not used to save storage.
- For buffer sizes that are used infrequently, consider reducing the number of buffers to save storage.
- For frequently used buffer sizes, increase the number of buffers if the lookaside hit ratio is less than 10 for index buffers, or less than 1 for data buffers.
- When increasing the buffer allocation:
  1. Monitor the lookaside hit ratio for the current buffer pool allocation.
  2. Increase the number of buffers in one or more of the buffer pools. The new buffer allocation does not take effect until the NetView program is stopped and restarted.
  3. Monitor the lookaside hit ratio for the new allocation.
  4. Repeat steps 2 and 3 until the lookaside hit ratio does not improve with an increase in the number of buffers for the buffer pool.
- When decreasing the buffer allocation:
  1. Monitor the lookaside hit ratio for the current buffer pool allocation.
  2. Decrease the number of buffers in one or more of the buffer pools. (The new buffer allocation does not take effect until the NetView program is stopped and restarted.)
  3. Monitor the lookaside hit ratio for the new allocation.
  4. Repeat steps 2 and 3 until the lookaside hit ratio degrades noticeably with a decrease in the number of buffers for the buffer pool.

VSAM Database Maintenance

Use the VSAM access method services (AMS) REPRO or IMPORT and EXPORT commands to reorganize a database to recover space lost through CI and CA splits. Use of these commands can improve performance in accessing the database and in reducing database size. Free space decreases the likelihood of CI and CA splits thereby, improving performance. This, in turn, decreases the chance that VSAM will move a set of records to a different cylinder away from other records in the key sequence. When a direct insert occurs that does not result in a split, VSAM inserts the records into available free space.

The NetView IDCAMS command enables you to use AMS utility commands that are stored in a data set. This function enables you to perform the NetView VSAM database maintenance from the NetView program without having to shut down the VSAM database.

You can use the PURGEDB command to delete unneeded data from the hardware monitor and session monitor VSAM databases. If you use PURGEDB, use the VSAM AMS REPRO or IMPORT and EXPORT commands to reclaim free space.
To delete historical data from the hardware monitor or session monitor databases, use the RESETDB command to clear the databases while the NetView program is active. RESETDB is simpler than deleting and redefining the databases and is faster than using PURGEDB to purge the entire database. Ensure sure that the REUSE operand is coded on the Session Monitor cluster definition (CNMSI201). RESETDB requires the REUSE parameter.

Use the DBAUTO command to perform maintenance on your VSAM databases while the NetView program is running. The DBAUTO command works on the session monitor, hardware monitor, Save/Restore and 4700 Support Facility databases. You can perform the following operations:

- Switch to a secondary database
- Selectively purge entries in the session monitor or hardware monitor databases
- Reorganize a database
- Erase the contents of a database

Refer to the [Tivoli NetView for z/OS Command Reference](#) for a description of the syntax and use of the IDCAMS, RESETDB, DBAUTO, and PURGEDB commands.
Chapter 10. Additional Tuning Considerations

This chapter provides miscellaneous tuning information.

Tuning Recommendations

The tuning recommendations are arranged in order of expected effect on performance, with the most important tuning considerations listed first. These recommendations are described in detail in this chapter.

1. For SNA topology manager, avoid using commands that cause a large number of storage references during peak periods of activity. See “SNA Topology Manager” on page 123.


3. Use the TASKUTIL command to monitor NetView task utilizations, queue lengths, storage use, and active command lists. See “TASKUTIL Command” on page 127.

4. To optimize performance for command security authorization, consider using AUTOSEC=BYPASS and SEC=BY on CMDMDL statements for commands that do not require security checks. See “Command Security” on page 109.

5. Use the OMIT operand of the STATOPT statement to control the storage requirement of the status monitor. See “Status Monitor STATOPT Filtering” on page 124.

6. Use the high performance transport instead of the management services transport for LU 6.2 communication when possible. See “LU 6.2 Transport” on page 112.

7. To improve CPU usage when using installation exits, do not use dummy exits, and optimize the performance of frequently invoked exits. See “Installation Exits” on page 111.

8. Decide whether to use persistent or nonpersistent sessions for NetView-NetView communications. Persistent sessions are highly recommended for all but very low traffic environments. See “Persistent and Nonpersistent LUC Sessions” on page 118.

9. You may be able to shorten the path length required to perform span of control verification by migrating to the NetView span table. For information on implementing the NetView span table, refer to the Tivoli NetView for z/OS Security Reference.

10. Use the DSRBS command to monitor the DSRB allocations for the NetView data services tasks (DSTs). See “Data Services Request Blocks (DSRBs)” on page 110.

11. To save storage, comment out CMDMDL statements relating to NetView functions that you do not use. See “Minimizing Storage Usage” on page 160.

12. Do not specify the MAXSESS keyword on the CNMAUTH statement in DSILUCTD. This enables the NetView program to allocate as many LUC sessions as it needs for alert forwarding, remote database retrieval, and status forwarding. See “MAXSESS Keyword” on page 114.
Address Space Dispatch Priority

Because they interact closely, use the same dispatch priority for the NetView, GMFHS, RODM, and Event Automation Services (IHSAEVNT) address spaces.

Automated Operations Network (AON) Performance Considerations

- The ENVIRON SETUP control file entry identifies attributes of NetView running AON and enables you to tailor the control file to your needs. The ENVIRON SETUP entry is optional. If not included, AON uses all of the defaults. ENVIRON SETUP parameters that may affect performance are:

  GENALERT
  GENALERT=Y is required to generate alerts. The default is N. To update the NetView management console (NMC) and Resource Object Data Manager (RODM) with automation information, specify GENALERT=Y. If you do not require this information, a significant reduction in CPU time can result when GENALERT=N is specified instead of GENALERT=Y.

  RODMAIP
  RODMAIP defines whether the RODM AIP (Automation in Progress) operator status is set, causing the AIP pattern to display for the affected resource in NMC. RODMAIP=NO is the default setting. Specifying RODMAIP=NO in the ENVIRON SETUP entry is recommended when AIP status is not needed; this will cause a reduction in the CPU time used to process the AON workload.

  TRACE
  TRACE=ON enables AON to perform tracing. The default value is NONE, which prevents program entry, exit, or tracing. A setting of TRACE=OFF enables the trace facility, but no tracing is done at this time. If you are not planning to perform tracing, setting TRACE=NONE will result in a reduction in CPU time compared to setting TRACE=OFF.

- The ENVIRON DDF control file entry defines status update characteristics for the environment. The ENVIRON DDF entry is required to implement DDF. The default is DDF=NO. When DDF=STATUS is defined, DDF colors are defined by resource automation or VTAM status in DDF entries of the control file. Specifying DDF=NO will not initialize or log events to DDF. If DDF is not being used, setting DDF=NO will reduce the CPU time used to process the AON workload.

Browse

If you have storage limitations, control the use of the data set member browse function because it reads the entire data set member into storage. See "Estimating Storage Usage" on page 135 for the storage required for the data set member browse function.

Note: This consideration does not apply to browsing the network log.

Using the BROWSE command, you can browse members on a remote NetView system. When a cross-domain browse request is processed, the RMTCMD command is used internally to satisfy the request. The RMTMAXL parameter of the DEFAULTS and OVERRIDE commands specifies the maximum number of lines transferred for a cross-domain member browse request. If the remote member contains more than the maximum number of lines, the BROWSE command
continues with the permitted number of lines, and message CNM2061 is issued. The BROWSE command uses the RMTMAXL setting of the operator issuing the cross-domain browse request. A large value for RMTMAXL allows a cross-domain member browse request to return large amounts of data, but can cause delays with other RMTCMD LU 6.2 communications. The default value of RMTMAXL is 2500 lines.

Command Security

This section discusses certain considerations for achieving maximum performance of NetView when security checks are run against commands. There are two different methods of command security in NetView: the NetView command authorization table and a system authorization facility (SAF) product, such as RACF. Refer to the Tivoli NetView for z/OS Security Reference for more information about command security.

Recommendations:

- Regardless of which command security you use, you can code SEC=BY on the CMDMDL statement in DSICMD for commands that do not require a security check (HELP, for example). This improves performance by eliminating security check processing time on commands you consider harmless and safe.
- For all three methods of security, you can code AUTOSEC=BYPASS on the DEFAULTS command to bypass security checks on all commands originating from the automation table, assuming your automation table member update process is secure. This can eliminate unnecessary checking of commands not entered at an operator console.
- When using RACF, auditing all resources can degrade system performance. Setting RACF auditing to NONE for resources in the NETCMDS class can improve performance. Using RACF AUDIT provides you with an audit trail of attempts to issue unauthorized commands or command lists.
- When using a backup command authorization table with SAF as your primary command security method, you should make sure generic command identifiers are specified to prevent unnecessary searches of the backup command authorization table. For example, in RACF:
  
  RDEFINE NETCMDS *.*.* UACC(READ)

- When migrating to the SAF NETCMDS class or the NetView command authorization table with the SECMIGR command, excess statements may be generated which can be deleted to (slightly) improve performance. When the SECMIGR tool generates statements equivalent to scope KEYCLASS statements, and a VALCLASS statement is not specified for the keyword, a statement is generated to cover any specified values. For keywords that cannot have values, this statement can be safely deleted. For example, since the AUTOTBL OFF keyword has no value, statements generated for a command identifier of netid.luname.AUTOTBL.OFF.* can be safely deleted.
- Processing generic command identifiers is the most performance intensive part of searching the command authorization table. If you are using generic command identifiers (wildcards) in the command authorization table, you can code specific command identifiers on EXEMPT statements for commands that do not need protection. Using specific (not generic) PROTECT statements for frequently used commands should also be helpful.
**Data Services Request Blocks (DSRBs)**

The following types of data services request blocks (DSRBs) are used to store information about a transaction request being processed by a data services task (DST):
- Unsolicited (DSRBU)
- Solicited (DSRBO)

DSRBUds are used to process unsolicited problem determination request units (RUs). DSRBOs are used to process solicited RUs and VSAM requests. If more requests are received than there are DSRBs available, the requests are queued.

The number of DSRBOs and DSRBUds allocated for a DST are defined as parameters on DSTINIT statements in the DSIPARM initialization member for the DST. If you do not specify the parameters, the default allocations are 3 DSRBOs and 5 DSRBUds. See the [Tivoli NetView for z/OS Administration Reference](#) for the syntax of DSTINIT statements.

The DSRBS command displays statistics on the data services request block use for the NetView program and user-written DSTs. The DSRBS output is a snapshot of the current DSRB use, and can be used to determine if the DSRB allocations are sufficient.

Figure 25 shows a sample of output from the DSRBS command.

```
<table>
<thead>
<tr>
<th>DSRBS Data Services Request Block Usage for BNJDSERV 13:35:07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsolicited DSRBs: 5 Used: 0 Free: 5</td>
</tr>
<tr>
<td>Solicited DSRBs: 5 Used: 5 VSAM Redrive: 0 Free: 0</td>
</tr>
<tr>
<td>TOTAL DSRBs: 10 Used: 5 VSAM Redrive: 0 Free: 5</td>
</tr>
<tr>
<td>Current DSRB Usage</td>
</tr>
<tr>
<td>No. DSRB STATUS Taskname Type Request Redrive Serial No. Step No.</td>
</tr>
<tr>
<td>001 Uns1 Inact</td>
</tr>
<tr>
<td>002 Uns1 Inact</td>
</tr>
<tr>
<td>003 Uns1 Inact</td>
</tr>
<tr>
<td>004 Uns1 Inact</td>
</tr>
<tr>
<td>005 Uns1 Inact</td>
</tr>
<tr>
<td>006 Soli Active NCF01PPT VSAM Erase No 5104 5</td>
</tr>
<tr>
<td>007 Soli Active NCF01PPT VSAM Get No 5100 14</td>
</tr>
<tr>
<td>008 Soli Active NCF01PPT VSAM Put Yes 5105 8</td>
</tr>
<tr>
<td>009 Soli Active PCF01PPT VSAM Get No 5102 13</td>
</tr>
<tr>
<td>010 Soli Active PCF01PPT VSAM Put No 5103 12</td>
</tr>
</tbody>
</table>
```

**Figure 25. Sample Output from the DSRBS Command**

The statistics displayed include:

**DSRB Type Statistics**

- **Unsolicited DSRBs (DSRBU)**
  - Number allocated, in use, and available

- **Solicited DSRBs (DSRBO)**
  - Number allocated, in use, being redriven because of VSAM contention, and available
Total DSRBs (DSRBO + DSRBU)

Number allocated, in use, being redriven because of VSAM contention, and available

For each DSRB currently in use, the following information is displayed:

1. Type of DSRB (unsolicited or solicited)
2. Status (active or inactive)
3. Task that initiated the request
4. Type of request (VSAM or CNM)
5. Type of VSAM request (ENDREQ, ERASE, GET, POINT, PUT)
6. Whether the VSAM request is being redriven because of contention (YES or NO)
7. Request serial number
8. Step sequence number

The DSRBS command is useful in determining DSTs that are waiting for requests to complete. DSRBS is also useful in determining the optimum number of DSRBs to allocate. Too few DSRBs can result in DST requests being queued. Too many DSRBs can result in an excessive numbers of VSAM requests being redriven, which causes extra CPU overhead.

The following are recommendations for tuning the DSRB allocations for a DST.

- If DSRBOs are frequently in VSAM redrive status, consider lowering the number of DSRBOs allocated for the DST.
- If a majority of the DSRBs are frequently in use, consider increasing the number of DSRBs allocated for the DST.
- Monitor the DSRB usage for the DSIGDS, AAUTSKLP, and AAUTCNMI tasks closely, because they benefit from additional DSRBs in many environments.
- Performance of the BNJDSERV task (hardware monitor) can degrade if additional DSRBs are added, because its VSAM requests may go into redrive status.

Installation Exits

This topic provides performance considerations when using exits.

- For most OST-type exits (DSIEXxx), a “LOAD FAILED” message is issued if the exit is not found. Using dummy exits would prevent this message, but degrade system performance by causing NetView to execute instructions to set up a call to the dummy exit every time the exit is driven, therefore unnecessarily using CPU.
- Coding exits for frequently called exits also degrades performance on systems with heavy message loads, especially if the exits are coded in C or PL/I. C has a greater initialization overhead than PL/I. If you code very frequently called exits, consider using Assembler.
- If you use the BLOG command list, be aware that DSIEX18 is run under the browse task to match the search arguments specified in the BLOG input. If you have a large log and the search string matches only a small number of records, the forward or backward function key causes the log to be searched until enough records matching the string are found to fill the screen. This could mean
that thousands of records will be searched before enough matches are found to fill the screen, while resulting in a large amount of processing time being used.

### 3270 Java Support

This function provides support for the Java virtual machine in the NetView address space. This feature enables both native programs and customer written Java programs to run under NetView. NetView customization benefits from the platform independent nature of Java. MVS 5.2.2 or above is required for 3270 Java support. Users can view host data from any workstation with this feature. Most components of NetView, including those requiring full screen, are available. In Tivoli NetView for OS/390 V1R3 or later, users will not notice a difference in performance with regard to platform used.

There are several factors that can affect the performance of this function:

- Ensure that the minimum hardware requirements, as outlined in the Tivoli NetView for z/OS Program Directory are met.
- The "disabled" option must be specified when users are viewing non-secure data.

**Note:** Encrypted data transfer is the default. Data encryption will increase response time and CPU usage. Refer to the Tivoli NetView for z/OS User’s Guide for instruction to change the default.

- It is recommended to have only one 3270 Java session active on a workstation at a time to increase stability.

### LOGTSTAT Command

The LOGTSTAT command can be used to write task utilization data to the System Management Facility (SMF) log. You can use the LOGTSTAT command to create a record for one specified task, or for all tasks that were running at the time when the LOGTSTAT command was issued. If LOGTSTAT is used to generate records for all tasks, an SMF record is written for each task that is active.

LOGTSTAT data can be useful in determining task start/end times. It also provides some Resource Limits data similar to that produced via TASKMON. Specific CPU storage usage for a task from startup to termination or for a known time interval can be determined from the LOGTSTAT data as well. For more information, refer to the Tivoli NetView for z/OS Command Reference.

### LU 6.2 Transport

The NetView LU 6.2 transport is a programming interface that implements architected protocols to enable applications in network nodes to communicate using conversations over LU 6.2 sessions.

The NetView LU 6.2 transport consists of two similar application program interfaces: the management services (MS) transport and the high performance transport. For applications that run in the NetView program, each transport provides a high-level programming interface to mask the LU 6.2 complexities. An application registered with the appropriate transport can send data in architected envelopes to a partner application and receive data in return.

Although both transports provide the same functions and mask the LU 6.2 complexities, each transport offers its own advantages.
The high performance transport uses different LU 6.2 protocols that are faster than the protocols used by the MS transport. Because of these protocols, the high performance transport provides general error notification rather than specific error notification about data. In addition to the advantage of speed, the high performance transport enables programmers to define session parameters such as RU size.

The MS transport uses LU 6.2 conversation protocols that generate more network traffic than the high performance transport protocols to transport each piece of data. The advantage of the MS transport is that it guarantees delivery of data or specific error notification about the data.

Refer to the [Tivoli NetView for z/OS Application Programmer’s Guide](#) for information about LU 6.2 transports.

The following tuning recommendations apply to the LU 6.2 transport.

- When designing applications to use the LU 6.2 transport, use the high performance API instead of the management services API when possible.
- Send requests with a reply expected require more processing than send requests without reply expected, because of the extra overhead of time-out checking and processing the reply. When designing applications to use the LU 6.2 transport, use send requests without reply expected where possible.
- If you decide to use send requests with reply expected, create a CMDMDL statement for the reply processor in the DSICMD NetView initialization member. If the reply processor is a command list, use the CMDMDL statement shown as follows:
  
  clistname  CMDMDL  MOD=DSICCP

  The presence of the CMDMDL statement eliminates I/O to the command list data set (DSICLD) to verify the existence of the reply processor before the request is sent.

- The NetView constants module (DSICTMOD) contains an entry for the LU 6.2 transport support. This entry specifies the number of LUs with which you expect to have sessions. This value is used to optimize control block access. The default value is 2000. If you expect to have more partner LUs than the default, change this value in DSICTMOD. The number you specify does not need to be exact, but too small a number hinders access to control blocks. A value in excess of the expected number of partner LUs can result in unused virtual storage, but can improve access to control blocks. In general, it is better to overestimate than underestimate. Refer to the [Tivoli NetView for z/OS Installation: Getting Started](#) for information about changing and relinking DSICTMOD.

- The high performance transport enables applications to specify their logmode when they register. Sample logmode definitions are contained in member CNMS0001. Logmodes have an RUSIZES parameter that you can use to specify the maximum size of data in bytes that the LUs can send. The default RUSIZES parameter for LU 6.2 applications is 8585. The first two numbers are for the primary LU, and the second two numbers are for the secondary LU. Each pair of numbers represents a mantissa and an exponent, as follows:

  \[ M \times 2^N \]

  The default 8585 means that both the primary and secondary LU can send a maximum of \( 8 \times 2^5 \), or 256 bytes. Adjust the RUSIZES parameter appropriately for your LU 6.2 applications. The RMTCMD command uses the PARALLEL logmode (in CNMS0001), which uses RUSIZES=8787 (or 1024 bytes). Consider
increasing the RUSIZES parameter for the PARALLEL logmode to higher values, so that the RMTCMD can send larger data buffers.

- If you plan to forward messages or MSUs to the generic automation receiver (NVAUTO) for automation table processing, modify the CMDMDL statement for DSINVGRP to make the generic automation receiver command processor resident (by using RES=Y or deleting RES=N). Making this command processor resident avoids the I/O needed to load the command processor every time the generic automation receiver processes a message or MSU.

**MAXSESS Keyword**

You can use the MAXSESS keyword on the CNMAUTH statement in DSILUCTD to restrict the number of cross-domain sessions the NetView program can set up to an adjacent domain. The value for MAXSESS, if specified, can be in the range 1-65535. The value in the samples is 10.

If you do not specify the MAXSESS keyword, the NetView program allocates as many sessions as it needs for alert forwarding and remote database retrieval for the hardware monitor and session monitor. Do not specify a value for MAXSESS unless you need to restrict the number of cross-domain sessions the NetView program can set up.

If you use nonpersistent sessions, do not specify a value for MAXSESS, because any idle sessions are brought down.

If you must specify a value for MAXSESS, use the following formula to calculate a value:

\[
\text{MAXSESS value} = 2 \times \text{(Number of session monitor sessions)} + 1 \times \text{(Alert forwarding, if NPDA.ALERTFWD=NV-UNIQ in CNMSTYLE)} + xx \times \text{(Number of concurrent NetView operators doing hardware monitor remote data retrieval)}
\]

If the MAXSESS value is exceeded, an SDOMAIN (set domain) command from the session monitor and hardware monitor can fail and alerts might not be forwarded.

If an SDOMAIN command fails, message DSI784 is issued.

**MultiSystem Manager (MSM) Performance Considerations**

- Plan to contain as much of the virtual storage demand as possible in central (real) and expanded storage. This keeps data movement to auxiliary storage (disk) to a minimum. This method is especially important for the GETTOPO scenario where many objects are created in RODM.

- If the networks that are being managed by MultiSystem Manager are large (more than 50,000 resources), consider staging the issuance of GETTOPO requests.

  This can be done at the managing agent level (LN, IP, Open, NetFinity, TMR), then at the subgroup level (service points for the various managing agents).

  Over time, this will spread the large CPU and storage demand required for GETTOPO.

**Note:** Do not attempt the GETTOPO command for MSM and MONITOR requests for SNATM at the same time.
Reducing the number of resources to be monitored or designated as critical can make a major difference in the number of RODM objects, which affects the initial topology processing and RODM storage.

For example, consider reducing the number of unmanaged resources by IP agent (UNMANAGED=NO for the IPRES command) or restricting the number of critical adapters identified to the LNM agent.

When exception views are used, the number of exception view table entries in the exception view file (specified using the MSM COMMON.FLC_EXCEPTION_VIEW_FILE statement in CNMSTYLE) affects the amount of CPU used during the GETTOPO processing.

You may want to specify only entries that will be needed for the agents collecting topology. For example, IP is needed, but LNM is not.

Performance of MultiSystem Manager can be improved by choosing to run several autotasks in support of multiple concurrent RUNCMDs.

Choosing the correct number of autotasks is dependent on the number of managing agents (LNM, IP, Open, NetFinity) you are supporting and the multiprogramming capability of your processor.

---

**NetView Access from the Web Browser**

This function provides the capability for users in NetView to convert host data to HTML and then made available to web browsers via the IBM Internet Connection Server. Users can logon to the internet from a workstation, go to a URL for their NetView system, and request/receive command-line data responses.

Beginning in Tivoli NetView for OS/390 V1R3, there is a hard coded 1000 line limit for allowable data display when using the web browser function. If more than 1000 lines of data are requested, users will see a message at the bottom of the requested data stating how many lines were truncated. Users should not knowingly attempt to browse members or datasets that will far exceed this limit. The entire file must be read in, causing an increase in response time and CPU, both at the host NetView and at the workstation. Performance testing showed no significant difference in response time to commands with regard to both platform and specific web browser used.

---

**NetView Constants Module (DSICTMOD)**

DSICTMOD is the NetView constants module. The sample is CNMS0055. This sample assembles and link-edits the module. Changes in the module require an assembly and link-edit.

All of the values are described in the NetView for z/OS Installation: Getting Started. The following constants are described in this book:

- The expected number of task global variables is described in “Global Variables” on page 42.
- The expected number of common global variables is described in “Global Variables” on page 42.
- Sense code filtering is described in “DASD Filtering” on page 73.
- LU 6.2 transport values are described in “LU 6.2 Transport” on page 112.
- Nonpersistent session time-out values are described in “Persistent and Nonpersistent LUC Sessions” on page 118.
- The management of below-the-line storage is described in “Minimizing Storage Usage” on page 160.

---

Chapter 10. Additional Tuning Considerations  115
NetView-NetView Communication

You can use the RMTCMD command to send commands to another NetView program and receive responses, and to query details about RMTCMD associations. The RMTCMD command offers the following advantages over NetView-NetView (NNT) sessions and the ROUTE command.

- The RMTCMD command consolidates the LU-LU communications from multiple operators by sending commands to a remote NetView program using a pair of LU 6.2 sessions. This saves storage and processing time for setting up sessions, and for maintaining session awareness in VTAM and the session monitor. A separate session is not needed for each operator.
- The RMTCMD command uses large buffers and RU sizes, improving performance for operators receiving large multiline messages.

NetView Program-to-Program Interface

The NetView program-to-program interface enables application programs to send network management vector transport (NMVT) or control point management services unit (CP-MSU) requests to the NetView program and enables application programs to send data buffers to, or receive data buffers from, other application programs. An application program can be a sender program, a receiver program, or both. Application programs can send data buffers to, or receive data buffers from, other applications running in the same host as the NetView program, or in a different host.

Each receiver program has a buffer queue for temporary storage of incoming data buffers. These buffer queues reside in the program-to-program interface. A sender program sends a data buffer to a receiver buffer queue, and the receiver program retrieves the data buffer from the buffer queue.

When you define a program as a receiver, you also define the buffer queue limit, the maximum number of outstanding buffers that can be stored in the receiver buffer queue. When the receiver buffer queue is full, and other buffers are sent, the sender programs receive a return code of 35.

You can use the DISPPI command to display information about PPI buffer queues, including buffer limits, buffer queue lengths, total buffers sent, and buffer storage usage. This command displays information for the current receiver or receivers defined to the program-to-program interface. Figure 26 contains sample output from the DISPPI command.

<table>
<thead>
<tr>
<th>Receiver</th>
<th>Identity</th>
<th>Status</th>
<th>Limit</th>
<th>Queued Buffers</th>
<th>Total Buffers</th>
<th>Storage Allocated</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETVALRT</td>
<td>ACTIVE</td>
<td>1000</td>
<td>0</td>
<td>18</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>DUIFSSCO</td>
<td>ACTIVE</td>
<td>100</td>
<td>0</td>
<td>10049</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>PPIMDID</td>
<td>ACTIVE</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>DUIATMGR</td>
<td>ACTIVE</td>
<td>1000</td>
<td>0</td>
<td>3287</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>FNMRCVQ</td>
<td>ACTIVE</td>
<td>99</td>
<td>0</td>
<td>20111</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>NPM60AP</td>
<td>ACTIVE</td>
<td>99</td>
<td>0</td>
<td>20111</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>DUIFIO00</td>
<td>ACTIVE</td>
<td>1000</td>
<td>0</td>
<td>15027</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 26. Sample DISPPI Command Output
Another command, DISBQL, displays buffer queue limits and the number of buffers currently available on the buffer queue. You can use the SETBQL command to reset the buffer queue limit of a receiver.

The program-to-program interface trace facility enables you to set up a trace in the program-to-program interface for either an individual receiver or all current and future receivers. The program-to-program interface trace facility writes a trace record each time a user defines, deactivates, or deletes a receiver to the program-to-program interface and each time a buffer is sent or received. You control the program-to-program interface trace facility with the TRACEPPI command. You can use the TRACEPPI command to start, stop, modify, or end the program-to-program interface trace facility. Refer to \textit{Tivoli NetView for z/OS Command Reference} for more information about the TRACEPPI command.

Refer to the \textit{Tivoli NetView for z/OS Application Programmer's Guide} for detailed information about the NetView program-to-program interface.

**Network Asset Management Facility**

The network asset management facility enables you to collect vital product data (VPD) from the network. Refer to \textit{Tivoli NetView for z/OS Command Reference} for information about these commands: VPDALL, VPDPU, VPDDCE, VPDCMD, and VPDLOG. The nature of those commands and the functions they perform extend their elapsed time for execution.

A request for VPD is sent serially to the physical units (PUs) and data circuit-terminating equipment (DCE) in the network. The PUs and DCEs must receive the request and send a response. The elapsed time for this collection varies depending on the configuration of the network and the traffic rate.

Consider dividing your network into logical sections and scheduling the collection of VPD from a section at a time during off-peak hours. Refer to the \textit{Tivoli NetView for z/OS Application Programmer's Guide} for interpretation of messages returned from the VPDCMD command.

**Partitioned Data Set (PDS) Allocation**

The following are the non-VSAM data sets used by the NetView program.

- **BNJPNL1**
  - NetView hardware monitor panel data sets
- **BNJPNL2**
  - NetView hardware monitor panel data sets
- **CNMPNL1**
  - NetView panel data sets
- **DSICLD**
  - NetView command list data sets
- **DSILIST**
  - NetView listing data sets
- **DSIMSG**
  - NetView messages data sets
- **DSIPARM**
  - NetView definitions, automation table, and CMDMDL statements
- **DSIPRF**
  - NetView operator profile definitions
- **DSIVTAM**
  - VTAMLST data sets
The command list data set (DSICLD) is the most critical data set from a performance standpoint. Each time a command list is selected for execution, it is located and read from that data set (unless it is preloaded by LOADCL). A normal BLDL search of the partitioned data set directory is done to find the command list, enabling you to make command list changes dynamically. Usually, DSICLD is a concatenation of several data sets that may be on different direct access storage device (DASD) volumes.

Define the command list data sets with one extent and ensure that the extent will not fill up over the time the NetView program is running. If a data set expands to multiple extents after the NetView program is up and running, attempts to access members newly placed in the additional extents can cause I/O errors.

Retrieval time for command lists is an important consideration in command list performance. You can avoid physical I/O either by preloading command lists using the LOADCL command or by using a virtual I/O data set for DSICLD. If you cannot use either of those options, you can minimize physical I/O response time by placing DSICLD on a cached DASD device. If some of your command lists are on cached DASD and you have multiple volumes, put the cached volumes first in the DD definition list for DSICLD.

You can create temporary VIO data sets by placing an IEBCOPY statement in the NetView startup procedure, which copies the command lists to a VIO data set and passes that data set on to the NetView program. When you use these temporary data sets, you cannot dynamically change the command lists. You must restart the NetView program to include the changes.

If you use panels extensively, you can reduce I/O delays by placing the panel data sets in a VIO data set. However, you cannot change such panels dynamically.

**Persistent and Nonpersistent LUC Sessions**

If you have a multiple-domain environment with copies of the NetView program on different hosts communicating with one another, you need to decide whether to use persistent or nonpersistent sessions for the NetView-NetView communication used by the session monitor, the hardware monitor.

Persistent sessions are most commonly used with leased lines, while nonpersistent sessions are most commonly used with dialed (switched) lines. Using nonpersistent sessions involves the additional overhead of establishing the dialed connection. Therefore, use nonpersistent sessions only when you expect very low traffic over the session. In deciding between persistent and nonpersistent sessions, examine the trade-off between leased line cost, the host processor cost of establishing the dialed connection, and the nonpersistent session.

**Using Nonpersistent Sessions over Dialed Lines**

If you decide to use nonpersistent sessions over dialed lines, you should understand the relationship between the timers maintained by the NetView program and NCP.

- For every nonpersistent session that it maintains, the NetView program has an activity timer, which it resets every time traffic is sent over the session. The value for this timer is defined by a constant in DSICTMOD. Refer to the [Tivoli NetView for z/OS Installation: Getting Started](#) for more information. If there is no activity over the session within the timer interval, the NetView program ends the nonpersistent session.
For dialed connections involving an NCP (VTAM-NCP or NCP-NCP), NCP has an activity timer, which it resets every time traffic is sent over the dialed line. This NCP timer is controlled by setting BRKCON=CONNECTO on a line or PU statement in the NCP definition. The value for this timer is set by the ACTIVTO parameter on the GROUP statement. Refer to the appropriate NCP, SSP, and EP publication for more information about activity timers. If no traffic occurs over the dialed line within the timer interval, NCP ends the dialed connection.

For VTAM-VTAM dialed connections, if VTAM detects that a dialed connection is active but no sessions are occurring over it, VTAM ends the dialed connection.

For a nonpersistent session over a VTAM-VTAM dialed connection, the session times out according to the NetView timer. The amount of time before the line drops is therefore determined by the NetView timer.

If NCPs are involved in the dialed connection, the nonpersistent session times out as follows:

- If the NCP timer value is less than the NetView timer value, the NCP ends the dialed connection, causing the nonpersistent session to end. In this case, the line drop time is based on the NCP timer.
- If the NetView timer value is less than the NCP timer value, the NetView program brings down the nonpersistent session. Because the session termination causes traffic over the dialed connection, NCP resets its activity timer. If no other activity occurs within the NCP timer interval, NCP ends the dialed connection. In this case, the line drop time is based on the NetView timer plus the NCP timer.

**RESOURCE Command**

The RESOURCE command displays statistics about NetView system resource use. This information is helpful in determining the amount of system resources used by the NetView program. Figure 27 contains sample output from the RESOURCE command.

```
DSI386I NETVIEW RESOURCE UTILIZATION 09:58:07
1 TOTAL CPU % = 6.65
2 E140EGNV CPU % = 0.81
3 E140EGNV CPU TIME USED = 14.06 SEC.
4 REAL STORAGE IN USE = 7664K
5 PRIVATE ALLOCATED < 16M = 612K
6 PRIVATE ALLOCATED > 16M = 17596K
7 PRIVATE REGION < 16M = 4160K
8 PRIVATE REGION > 16M = 32768K
END OF DISPLAY
```

**Figure 27. Sample Output from the RESOURCE Command**

The following information is displayed in the command output (see Figure 27).

1. **Total CPU Utilization**
   
   Total complex CPU utilization based on a maximum of 100%. This utilization is calculated over the most recent 1-second interval.

2. **NetView CPU Utilization**

   NetView CPU utilization based on a maximum of 100%. This utilization is calculated over the most recent 1-second interval.

3. **NetView CPU Time Used**
The combination of task control block (TCB) and service request block (SRB) CPU time used. This field is cumulative from when the NetView program was first started.

4 Real Storage In Use
The amount of real storage currently allocated to the NetView program (central plus expanded) shown in KB.

5 Private Allocated Below 16 M
The amount of virtual storage allocated below the 16-MB line.

6 Private Allocated Above 16 M
The amount of virtual storage allocated above the 16-MB line.

7 Private Region Below 16 M
The total amount of virtual storage below the 16-MB.

8 Private Region Above 16 M
The total amount of virtual storage above the 16-MB.

The NetView program’s CPU use depends on message and network traffic levels, which can appear in bursts. Therefore, the NetView program’s CPU utilization can appear to spike over the small 1-second interval that the RESOURCE command uses. The cumulative CPU time, monitored over time, is a better indicator of the NetView program’s level of CPU use.

The NetView program’s real (processor) storage use includes central plus expanded storage frames allocated to the NetView program. On systems with little or no storage contention, a program’s real storage use can be inflated and may not represent the true working set size.

Resource Limits

The resource limits function in NetView provides users with the ability to prioritize the use of system resources by essential and non-essential tasks. With this function, users can monitor and limit the resource usage for various NetView tasks. Users can obtain information that helps you plan for and tune NetView. The following tasks can then be adjusted according to the amount of storage consumed, CPU used, and other factors.

MAXSTG (Maximum Storage for a task)
Specifies the maximum amount of storage in Kilobytes that a task can use. When MAXSTG is reached and further storage requests are made, the DSIGET macro will return an “out of storage” return code.

SLWSTG (When to start limiting a task from getting storage)
Specifies the maximum amount of storage in Kilobytes that a task can use before slowdown measures are used. When a task exceeds SLWSTG, all storage requests (using DSIGET) will have a time delay.

AVLMAX
Specifies a percentage that determines, for any task, at which value the DSIGET macro will return an “OUT OF STORAGE” return code. In addition, queuing a message to a task that is over its AVLMAX limit will result in the “task not active” condition, and the message will not be transferred.
The percentage is specified as a decimal number in the range of 0–99. NetView computes the ratio of the amount of storage a task is using compared to the sum of that amount and the amount of storage left in the NetView address space. If the task usage is above the specified limit, the DSIGET rejects the request.

**Note:** The specified limit enables users to have the tasks that are under the most storage stress to fail instead of letting the entire region deplete storage, which can result in the failing of all tasks. For example, when using the default of AVLMAX=90, a task using 45 MB of storage will get a storage failure if the amount of free space falls below 5 MB. Specifying a lower value for AVLMAX leaves more of the free address space for tasks that do not overuse storage. Specifying a higher or no limit value for AVLMAX increases the risk for storage failures on all tasks, regardless of whether they are in storage stress.

When NetView is started, the AVLMAX default is set to 90. This setting can be modified by using the DEFAULTS command.

**AVLSLOW**

Specifies a percentage that determines, for any task, at which point slowdown measures are used. In addition, queueing a message to a task that is over its AVLSLOW limit will result in that task having the same slowdown measures applied based on how much over the limit the receiving task gets.

When NetView is started the AVLSLOW default is set to 85. This setting can be modified by using the DEFAULTS command.

Slowdown measures will be used when a task exceeds AVLSLOW. All storage requests (using DSIGET) and message queuing to the affected task will have a time delay. The time delay will be one microsecond for each byte of storage requested over the SLOWSTG limit value, and quadrupled for each 1% the task is beyond the AVLSLOW value thereafter. As the task’s use of storage grows, this produces a slowdown effect that is proportional to the size of the request.

The initial slowdown rate is calculated to allow storage to grow at a rate of 1 megabyte per second, in the range 0–1% beyond the limit value. When the AVLSLOW value is lowered, the point at which slowdown occurs is lowered. For example, SLOWSTG=85 will trigger if the task is using 42.5 MB of storage and there is only 7.5 MB of good memory left in the region. SLOWSTG=80 will trigger if the task is using 40 MB of storage and there is only 10 MB left in the region.

**MAXCPU (Maximum CPU for a task)**

Specifies the maximum CPU utilization allowed for a task. When a task exceeds the limit, automatic task suspension measures will be used to bring the task back into the specified range. The task will be suspended until enough time passes for the CPU to be below the specified limit.

**MAXIO (Maximum input/output transactions for a task)**

Specifies the maximum number of logical VSAM I/O requests per minute allowed for a task. When a task exceeds the limit, automatic task suspension measures will be used to bring the task back into the specified range.
**MAXMQIN (Maximum number of messages received by a task)**

Specifies the number of message KB per minute that is allowed to be sent to the task from other tasks. When a task exceeds the limit, automatic task suspension measures will be used to bring the task back into the specified range. The intent is to add enough of a delay to each request so that the measured rate over a one minute or longer period of time falls to near or below the specified value.

**MAXMQOUT (Maximum number of messages a task can send to a task)**

Specifies the number of message Kilobytes per minute allowed for a task to send to another task. When a task exceeds the limit, automatic task suspension measures will be used to bring the task back into the specified range. If the task attempts to queue a message to another task, it will be slowed down until the rate is under the limit. The intent is to add enough of a delay to each request so that the measured rate over a minute or longer period falls to near or below the specified value.

Changes can be made dynamically to all resource limits settings. Old values are cancelled within one second of the OVERRIDE command being processed. The tasks continue with the new limit in force. Coding a value of 0 will turn off a specified resource limit setting. Refer to the Tivoli NetView for z/OS Command Reference for more information about using the OVERRIDE parameter.

**LOGTSTAT (Whether to log the data to SMF)**

Specifies whether resource utilization data is logged to the external log (SMF).

### Using Resource Limits

Use extreme care when you restrict a task with resource limits. Performance data should be gathered using the V5R1PERF clist sample in Figure 7 on page 4 and then analyzed. Gather data over time for peak, normal, and off-shift workload to ensure that a complete analysis can be made.

There are certain areas within NetView that use a large amount of storage, high CPU during certain operations or at initialization. Limiting these tasks beyond the default settings is not recommended. For example, the Session Monitor main task, AAUTSKLP, can use high CPU during purge processing and can use a large amount of storage during a network outage. The SNA Topology Manager task FLBTOPO will use a large amount of the CPU, especially during TOPOSNA MONITOR requests. Do not limit resource utilization for tasks, such as these, at peak workload.

The resource limit function can prevent an operator task or autotask from looping and saturating a processor. When the task CPU limit is exceeded, BNH161I is issued and the CPU usage, for the OST, will be throttled. User intervention can terminate the application. The same result is true for a user-written application that is issuing DSIGET macros to obtain storage, but never using DSIFRE to release the storage. Resource limits provide the ability to put a cap on the task storage usage and will abend the task if the storage limit exceeds the maximum allowed (MAXSTG).
SNA Topology Manager

The amount of virtual storage and host processor usage by SNA topology manager on your system depends on the size of your network and the number of RODM objects that SNA topology manager creates. This topic describes how to use SNA topology manager efficiently.

Due to virtual storage and processor demands to collect and store the status and topology data, consider establishing most of your TOPOSNA MONITOR requests during off-peak periods for your system, for example, after network activation.

You can display statistics, traffic levels, storage usage, and other information with the TOPOSNA LISTRODM and TOPOSNA LISTSTOR commands. Refer to the Tivoli NetView for z/OS Command Reference for examples on using these commands.

Some TOPOSNA commands result in references to large numbers of objects in RODM and should be avoided during peak periods of activity for your system.

- The TOPOSNA STOP command stops the monitoring of the specified topology (network, local, or LU collection) associated with a specific node. As part of command processing, SNA topology manager marks each affected RODM object with unknown status. If there are a large number of affected objects, and storage is constrained on the system, this command can result in a large number of page faults. Therefore, if you have a large number of SNA topology manager objects in RODM, consider activating your TOPOSNA MONITOR requests throughout the peak periods of activity.

- The TOPOSNA STOPMGR command stops the topology manager in an orderly fashion. As part of this command processing, SNA topology manager marks all objects it has created in RODM with unknown status. If there are a large number of SNA topology manager objects in RODM, and storage is constrained on the system, this command can result in a large number of page faults. Therefore, if you have a large number of SNA topology manager objects in RODM, consider activating SNA topology manager throughout peak periods of activity.

- The TOPOSNA REFRESH command requests a refresh of values for the status resolution table, the OSI-display status table, and the Exception View table. When the Exception View table is refreshed with either TOPOSNA REFRESH EXVIEW,CLASS=xxxx or TOPOSNA REFRESH ALLTABLES,CLASS=xxxx, SNA topology manager refers to all objects in the classes listed in the CLASS=xxxx parameter to query their values and change them appropriately. If there are a large number of SNA topology manager objects in RODM, and storage is constrained on the system, refreshing the Exception View table can result in a large number of page faults. For this reason, if you have a large number of SNA topology manager objects in RODM, avoid refreshing the Exception View table during peak periods of activity.

- The TOPOSNA PURGE command deletes expired unreachable objects from the RODM data cache. As part of this command processing, SNA topology manager references all objects it has created in RODM to determine if they need to be purged. If there is a large number of SNA topology manager objects in RODM, and storage is constrained on the system, this command can result in a large number of page faults. For this reason, if you have a large number of SNA topology manager objects in RODM, you should avoid using the TOPOSNA PURGE command during peak periods of activity.
Warm Starts, Cold Starts, and Checkpointing

Performance is an important consideration in determining whether to warm start or cold start SNA topology manager and how often to checkpoint RODM.

There is no performance advantage in starting SNA topology manager with SNA topology data already residing in RODM; in fact, there is a small penalty in doing so. For performance, it is best to initialize SNA topology manager with no SNA topology manager objects in RODM, and then establish monitor requests for your network. Also, there is no performance advantage from checkpointing RODM with SNA topology manager objects residing in RODM.

Status Monitor STATOPT Filtering

The status monitor component of the NetView program collects and summarizes information on the status of resources defined in a VTAM domain. The status monitor can handle a maximum of 999999 resources. The status monitor preprocessor uses the resource definitions in the VTAMLST data sets to create a DSIPARM member named DSINDEF. DSINDEF contains the network resource information that the status monitor reads during initialization. Refer to *Tivoli NetView for z/OS Installation: Configuring Additional Components* for information on the preprocessor operation.

The status monitor requires approximately 120 bytes of storage for each resource that it monitors. By default, the status monitor keeps status for every resource in the network. If you would like to reduce the storage requirement for the status monitor, you can use the OMIT operand of the STATOPT statement. By specifying OMIT in the VTAMLST data sets, you can omit a node, plus all of the dependent lower nodes that follow, from the status monitor’s view of the network definition.

Refer to the *Tivoli NetView for z/OS Administration Reference* for more information about coding the STATOPT statement.

STEPLIB DD Statements

Do not use a STEPLIB DD statement in your production NetView job control language (JCL). The presence of the STEPLIB DD statement causes its directory to be searched for each LOAD, LINK, or XCTL system macro executed during normal operation. That directory search degrades the performance of command procedures.

Place the NetView load libraries on the system’s LINKLST. Refer to *Tivoli NetView for z/OS Installation: Getting Started* for recommendations on placing the HLL run-time libraries, see “Command Processors” on page 39.

TASKMON Command

The TASKMON command is a REXX procedure that provides color-coded monitoring of all NetView tasks. The output under each group is sorted by the severity index. The first column on the left represents a percentage of the maximum value allowed. Usage of TASKMON is similar to TASKUTIL usage. You can use the data shown in the TASKMON output to view current and historical data on Resource Limits.

Color codes used by the TASKMON command:

- **White**  SLOWSTG limit exceeded
Yellow
  70% of limit for this line exceeded

Pink  80% of limit for this line exceeded

Red   90% of limit for this line exceeded

TASKMON provides statistics based on CPU percentage used on a single processor. TASKUTIL provides statistics based on CPU percentage for the sum of all defined processors, for example, a sysplex with six CPUs dedicated to NetView.

To issue a TASKMON command, enter:

```
TASKMON * *
```

A response similar to the following is displayed:
## TASKMON ---- START OF REPORT ----

### Severity Index | OPID | Current | Session | Maximum | Limit
--- | --- | --- | --- | --- | ---
| -CPU-- | | | | | |
| 152.00% | OPER3 | 1.52 % | 0.19 % | 33.85 % | 1.00 % |
| 23.34% | AUTO2 | 11.67 % | 1.53 % | 67.24 % | 50.00 % |
| 0.03% | NTV98BPT | 0.03 % | 0.04 % | 4.16 % | 99.99 % |
| 0.02% | DSIMONIT | 0.02 % | 0.01 % | 0.04 % | 99.99 % |

### Severity Index | OPID | Current | Session | Seconds
--- | --- | --- | --- | ---
| -Penalty Time-- | | | | |
| 81.89% | OPER3 | 81.89 % | 2.97 % | 81.66 S |

**Memory= 46.26% 24-Bit= 16.63% 31-Bit= 50.02% Left= 19844 K**

### Severity Index | OPID | Current | Maximum | Limit | Slowdown
--- | --- | --- | --- | --- | ---
| ----Storage---- | | | | | |
| 7.15% | AAUTSKLP | 1528 K | 1544 K | 999999 K | 999999 K |
| 5.86% | AUTO2 | 1172 K | 3605 K | 20000 K | 4000 K |
| 4.93% | MAINTASK | 1028 K | 1028 K | 26214 K | 23592 K |
| 1.96% | CMNTAMEL | 397 K | 401 K | 999999 K | 999999 K |
| 1.21% | AAUTCNMI | 244 K | 295 K | 999999 K | 999999 K |
| 0.82% | NTV98BPT | 164 K | 265 K | 999999 K | 999999 K |
| 0.81% | AUTO1 | 162 K | 251 K | 999999 K | 999999 K |
| 0.81% | DSISVRT | 162 K | 213 K | 999999 K | 999999 K |
| 0.47% | DSIAL2WS | 93 K | 121 K | 999999 K | 999999 K |
| 0.42% | BNJDSERV | 84 K | 85 K | 999999 K | 999999 K |
| 0.31% | NTV98VMT | 62 K | 81 K | 999999 K | 999999 K |
| 0.30% | DSICLICR | 59 K | 114 K | 999999 K | 999999 K |
| 0.29% | OPER3 | 57 K | 60 K | 999999 K | 999999 K |
| 0.28% | ALIASAPL | 55 K | 102 K | 999999 K | 999999 K |
| 0.27% | DSIS6DST | 54 K | 112 K | 999999 K | 999999 K |
| 0.23% | DSIGDS | 45 K | 89 K | 999999 K | 999999 K |
| 0.22% | VPDTASK | 44 K | 75 K | 999999 K | 999999 K |
| 0.21% | NTV98BLUC | 42 K | 93 K | 999999 K | 999999 K |
| 0.21% | DSIPDST | 42 K | 93 K | 999999 K | 999999 K |
| 0.20% | DSIQTSK | 39 K | 94 K | 999999 K | 999999 K |
| 0.18% | BNJDE36 | 35 K | 89 K | 999999 K | 999999 K |
| 0.18% | DSISQRAA | 36 K | 36 K | 999999 K | 999999 K |
| 0.18% | DSISQVAC | 36 K | 36 K | 999999 K | 999999 K |
| 0.17% | DSUDST | 34 K | 85 K | 999999 K | 999999 K |
| 0.17% | DSICRTR | 33 K | 85 K | 999999 K | 999999 K |
| 0.15% | DSIALUT | 30 K | 73 K | 999999 K | 999999 K |
| 0.14% | BNJMNPD | 28 K | 28 K | 999999 K | 999999 K |
| 0.13% | DSILog | 26 K | 73 K | 999999 K | 999999 K |
| 0.13% | DSICREM | 25 K | 73 K | 999999 K | 999999 K |
| 0.12% | DSICORS | 24 K | 77 K | 999999 K | 999999 K |
| 0.11% | DSICLTSG | 22 K | 73 K | 999999 K | 999999 K |
| 0.11% | OPER1 | 22 K | 73 K | 999999 K | 999999 K |
| 0.10% | OPER2 | 20 K | 20 K | 999999 K | 999999 K |
| 0.09% | DSITRACE | 18 K | 73 K | 999999 K | 999999 K |
| 0.04% | DUTSSCO | 8 K | 8 K | 999999 K | 999999 K |
| 0.04% | NTV98BRW | 8 K | 8 K | 999999 K | 999999 K |
| 0.03% | NTV98 | 6 K | 6 K | 999999 K | 999999 K |

Figure 28. Sample TASKMON Output (Part 1 of 2)
The output under each group is sorted by the severity index. The first column on the left (Severity Index) represents percentage of the maximum value allowed, or if no limit, the maximum value measured for any task.

The command WINDOW TASKMON * * (TAKE 4 produces a panel that displays the top four tasks in each of the measured categories. The WINDOW refresh PF key can be used to see updated values. TASKMON output is color coded based on severity.

For more information on using the TASKMON command and its operands, refer to Tivoli NetView for z/OS Command Reference.

## TASKUTIL Command

The TASKUTIL command displays task performance information, including central processing unit (CPU) utilization, queue lengths, storage use, and active command lists. This command is for NetView diagnosis and tuning purposes only.

The TASKUTIL command has three parameters:
TYPE
Specifies the type of NetView task:

ALL  All active NetView tasks. ALL is the default.
AUTO  NetView automation operator station tasks started with the AUTOTASK command. This does not include operator station tasks (OSTs) or distributed automation tasks (DISTs).
DIST  NetView distributed automation tasks started with the RMTCMD command. This does not include OSTs or autotasks.
DST   NetView data services tasks (DSTs). This does not include optional tasks (OPTs).
HCT   NetView hard-copy log tasks.
MNT   NetView main task.
NNT   NetView-NetView tasks.
OPT   NetView optional tasks. This does not include DSTs.
OST   NetView operator station tasks. This does not include autotasks or DISTs.
PPT   NetView primary program operator interface task (PPT).
VOST  Virtual Operator Station Tasks (VOSTs). A VOST is created during the ATTACH phase of each full screen automation request.

TYPE can also specify an individual task name.

DURATION
Specifies the length of the measurement, in seconds, over which utilizations are to be calculated. Valid values are from 1–60 seconds. The default is 2 seconds.

SORT
Specifies how the output should be sorted. For example, by NAME, TYPE, or CPU percentage (CPUP). CPUP is the default.

Refer to Tivoli NetView for z/OS Command Reference for a complete description of the TASKUTIL command and its parameters.

TASKUTIL Command Output
To display CPU utilization and storage used for NetView DSTs, enter the following command:

    TASKUTIL TYPE=DST

You receive a response similar to the output in Figure 29 on page 129.
This output was created using the SORT parameter default CPUP and the DURATION default of 2 seconds. Refer to the Tivoli NetView for z/OS Command Reference for a full explanation of the output fields.

The following events occur when the TASKUTIL command is invoked:

1. CPU time readings are taken for each NetView task. These are the cumulative CPU time values shown under the heading CPU-TIME.

2. The task processing the TASKUTIL command waits for the amount of time equal to the value of the DURATION parameter. The task is unable to process commands or messages during this time.

3. When the wait is over, a second set of CPU time readings is taken for each NetView task.

The two fields most important to tuning and diagnosis are N-CPU% (NetView program CPU utilization) and S-CPU% (system CPU utilization).

The N-CPU% and S-CPU% results reflect utilization over the measurement interval specified by the DURATION parameter of the TASKUTIL command. For short intervals, such as the default of 2 seconds, these utilizations are only snapshots, and can be subject to wide variation as the NetView program’s workload fluctuates. If you use a longer measurement duration, you will get more meaningful utilization results. The DURATION parameter has a limit of 60 seconds, because suspending a task for a longer period could cause problems if the task is receiving messages.

It is helpful to understand the N-CPU% and S-CPU% fields and how their values are calculated. The CPU utilization percentage values are calculated from the two sets of CPU time readings.

**Task NetView CPU Utilization**

N-CPU% is the task’s relative contribution to the NetView program’s CPU utilization, based on a maximum of 100%. The formula is:
N-CPU% = \frac{(\text{Task 2nd reading} - \text{Task 1st reading})}{(\text{NetView total 2nd reading} - \text{NetView total 1st reading})} \times 100\% \text{ x 100}

The numerator represents the task control block (TCB) time used by the task during the measurement. The denominator represents the total CPU time (TCB time + system request block (SRB) time) used by the NetView program during the measurement. Multiplying the result by 100% expresses the utilization as a percentage.

**Task System CPU Utilization**

S-CPU% is the task’s contribution to the total system CPU utilization, based on a maximum of 100%. The formula is:

\[
S-CPU\% = \frac{(\text{Task 2nd reading} - \text{Task 1st reading})}{(\text{Measurement duration} \times \text{Number of online host processors})} \times 100\%
\]

The numerator represents the TCB time used by the task during the measurement. The denominator represents the total CPU time that was available during the measurement (the total capacity of the host processors). The measurement duration must be expressed in seconds. Multiplying the result by 100% expresses the utilization as a percentage.

**Other CPU Utilization**

The NETVIEW OTHR category represents TCB utilization that cannot be attributed to active tasks. The following formula is used to calculate the value in the NETVIEW OTHR CPU-TIME field is:

\[
\text{OTHR CPU TIME} = \text{TOTAL CPU TIME} - \text{SRB CPU TIME} - (\text{Sum of CPU TIME foreach active task})
\]

Once the NETVIEW OTHR CPU-TIME value is calculated, it is used in the numerators of the N-CPU% and S-CPU% formulas above to calculate the NETVIEW OTHR N-CPU% and S-CPU% utilization values.

**SRB Utilization**

To calculate the NetView SRB N-CPU% and S-CPU% utilization values, two readings of the NetView address space cumulative SRB time are used in the numerators of the N-CPU% and S-CPU% formulas above.

**NetView Program Total CPU Utilization**

For the NETVIEW TOTL utilization results, two readings of the NetView address space cumulative CPU time (TCB + SRB) are used in the numerator of the N-CPU% and S-CPU% formulas. For the N-CPU% field, the result is always 100%. The S-CPU% field for NETVIEW TOTL is the same result as the NetView CPU % field reported by the RESOURCE command, except that the RESOURCE command uses a 1-second measurement duration. Refer to "RESOURCE Command" on page 119 for more information.

**System Total CPU Utilization**

The SYSTEM TOTL utilization is the average processor utilization percentage for all processors currently online. This result is calculated using the wait time for each of the host processors as follows:

\[
\text{System Total CPU Utilization} = \left(1 - \frac{\text{Sum of wait time for all processors during measurement}}{(\text{Measurement duration} \times \text{Number of online host processors})}\right) \times 100\%
\]
The SYSTEM TOTL CPU utilization is the same result as the TOTAL CPU % field reported by the RESOURCE command, except that the RESOURCE command uses a 1-second measurement duration. Refer to "RESOURCE Command" on page 119 for more information.

Calculating Task Utilizations with Two Observations of TASKUTIL

You can calculate task utilization values over longer measurement intervals using two observations of the TASKUTIL command. You can use the same N-CPU% and S-CPU% formulas to calculate task utilization over an hour, an 8-hour shift, a day, or a week.

The output in Figure 30 is from a TASKUTIL command invocation taken from the same system as in Figure 29 on page 129, two hours later.

It is evident that AAUTSKLP, the session monitor data services task, is the biggest contributor to NetView CPU use. Over the 2-hour period, AAUTSKLP used 1888.04 CPU seconds (23907.17 - 22019.13), while the NetView program as a whole used 4250.92 CPU seconds (59017.88 - 54766.96). For the 2-hour period, AAUTSKLP’s relative contribution to the NetView program’s total CPU utilization according to the N-CPU% formula is:

\[
\frac{23907.17 - 22019.13}{59017.88 - 54766.96} \times 100\% = 44.41\%
\]

Assume that there are six online host processors. AAUTSKLP’s contribution to the total system CPU utilization according to the S-CPU% formula is:

\[
\frac{23907.17 - 22019.13}{(2 \times 60 \times 60) \times 6} \times 100\% = 4.37\%
\]
Remember that you need to convert the measurement duration to seconds for these utilization formulas (2 hours × 60 minutes per hour × 60 seconds per minute = 7200 seconds). This technique makes the assumption that the task did not stop and restart during the measurement interval.

According to the S-CPU% formula, the NetView program’s contribution to the total system CPU utilization for the 2-hour period is:

\[
\frac{(59017.88 - 54766.96)}{2 \times 60 \times 60} \times 100\% = 9.84\% 
\]

**Suggestions for Using TASKUTIL**

Suggestions for using TASKUTIL are:

- Although an individual invocation of TASKUTIL provides a valuable picture of the level of activity in the NetView program, remember that it is only a snapshot. Use multiple invocations of TASKUTIL to spot trends in task CPU usage and storage growth.

- Consider setting an EVERY timer under an autotask to invoke TASKUTIL every 15 minutes or every hour. By default, output from the TASKUTIL command (DW0022I) goes to the network log. The TASKUTIL output can be used as important historical data in diagnosing performance or storage problems. You can use the BLOG command to browse only messages in the network log generated by the autotask invoking TASKUTIL.

- You can write applications to invoke TASKUTIL and trap the output for analysis:
  - A command list can warn of threshold boundaries for task CPU or storage usage being reached.
  - A command list/View panel pair can invoke TASKUTIL and display the data dynamically.
  - A REXX command list can reformat the CPU and storage data into a history file using EXECIO.
  - A command processor can build SMF records from the TASKUTIL output and write the records to the external log task (DSIELTSK).

- TASKUTIL output can aid in tuning the NetView program, for example:
  - If a command or component is showing constantly high CPU usage, the command or component should receive focus for tuning or performance improvements.
  - If a subset of the autotasks show heavy CPU usage while others have light CPU usage, you could probably improve performance by redistributing work among the autotasks.
  - If the NetView program contributes more to the total system CPU utilization than you would like, TASKUTIL output could be used to justify component tuning activity or possibly hardware upgrades.

- TASKUTIL output can aid in diagnosis of problems. For example:
  - If the task’s storage allocation of a task continues to rise, it could indicate that the task is getting storage, but not freeing it properly.
  - If an operator task, autotask, distributed autotask, or NNT shows continually high CPU usage, this could indicate an endless loop condition in a command list or command. TASKUTIL displays the active command list for these task types.
If an operator task, autotask, distributed autotask, or NNT shows the same command list active, a message queue build-up, and low CPU usage, this could indicate that the command list is stuck in a WAIT.

If the message queues for tasks continue to grow during a steady state period when the NetView program’s workload activity should be fairly uniform, and if the total system CPU utilization is near 100%, then this could indicate that the NetView program is not getting dispatched frequently enough to do its work. Continued message queue growth results in continued NetView storage growth, which can lead to storage abends. If you detect such a condition, consider ending low-priority CPU-intensive applications to relieve the system CPU constraint. If the NetView program regularly experiences message queue growth, consider making the MVS dispatching priority for the NetView address space more favorable.

**Tuning NetView Bridge**

NetView Bridge provides an effective means of connecting your current NetView installation to other databases or transaction processors. NetView Bridge can help you track problems you have identified in your resources, network, or configuration by creating, updating, and retrieving problem records in a non-NetView problem tracking database. A bridge dispatcher, running as a NetView autotask, establishes the NetView Bridge environment and passes transactions between user-written command procedures and database servers. The flow for transactions through NetView Bridge is:

1. A user-written command procedure in the NetView command list language, REXX, or high-level language (HLL) uses the NetView Bridge Requester application programming interface (API) to request that a transaction be sent to a database server.
2. The bridge dispatcher passes this request to an available database server.
3. The database server uses the server support API to retrieve the transaction. The server then performs the requested transaction by invoking the appropriate transaction processor, which works directly with the target database.
4. If the transaction processor creates a reply for this request, it uses the server support API to pass the reply back to the user-written command procedure using the bridge dispatcher. The reply is in the form of a message.
5. The user-written command procedure uses the NetView Bridge Requester API to extract the reply data from the message.

NetView Bridge is an automation tool, but is not intended to support vast quantities of data transport or migration. You might want to reduce the amount of data in a given transaction and the number of transactions.

To limit the traffic across the bridge, use discretion in choosing what is sent. Avoid sending large amounts of data, such as repetitious data, single transactions that approach 31 K in size, and low-priority data. You can add copies of database servers to a NetView Bridge server set. The number of servers you should define for your network depends on the CPU you are using and the number of transactions sent to the database. Adding too many servers can begin to degrade NetView Bridge performance because of limited database access. Experiment with the number of servers to achieve optimum performance in your environment.

To determine the queue limit at the receiver and the number of buffers currently available, use the DISBQL command. The DISBQL command enables you to display one or all receivers. If no more buffers are available for the receivers you
have defined, you might want to add database servers or increase the buffer queue
limit with the SETBQL command. Refer to the DISBQL command in the Tivoli
NetView for z/OS Command Reference for more information.
Chapter 11. Storage Considerations

This chapter provides techniques for accomplishing the following tasks:

- Estimating storage required for NetView, RODM, and GMFHS at the host
- Minimizing storage requirements
- Setting REGION sizes

Estimating Storage Usage

The diskette accompanying this book contains the following files:

1. **V5R1HOST.WK1**
   - A spreadsheet containing formulas for estimating minimum virtual storage requirements for the NetView, NetView Subsystem, GMFHS, RODM, and Event/Automation Service address spaces, the RODM data space, and minimum DASD requirements for important NetView, GMFHS, and RODM data sets.

2. **README.1ST**
   - A file containing instructions and notes for using the spreadsheet.

The spreadsheet can be used with spreadsheet programs that support the Lotus® WK1 format, for example, Lotus 1-2-3, Microsoft® Excel, and Borland QuattroPro.

If you do not have access to a spreadsheet program, use the formulas in the following tables to estimate the storage requirements for Tivoli NetView for z/OS.

If you have access to a spreadsheet program, the following tables may be useful for reference since they contain additional information not contained in the spreadsheets on how to pick appropriate values for the input parameters.

**Table 8. Base NetView Formulas.** All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETVTYPE</td>
<td>3</td>
<td>19124</td>
<td>19124</td>
<td>NetView type specified in member CNMSTYLE. The base NetView storage amount is 19124 KB.</td>
</tr>
<tr>
<td>Autotasks</td>
<td>25</td>
<td>265</td>
<td></td>
<td>The number of autotasks you expect to have active on your system. Use the TASKUTIL command to display the active NetView tasks on your system. Add the number of tasks with TYPE=AUTO and TYPE=DIST (distributed autotasks used by the RMTCMD command) shown in the TASKUTIL display.</td>
</tr>
</tbody>
</table>
Table 8. Base NetView Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NNTs</td>
<td>5</td>
<td>___ * 150</td>
<td></td>
<td>The number of NetView-NetView tasks (NNTs) you expect to have active on your system. NetView-NetView tasks are started when an operator from another NetView domain starts a session to this NetView domain. For the number of NNTs, use the TASKUTIL command (NetView V2R3 or later) to display the active NNTs on your system. Add the number of tasks with TYPE=NNT shown in the TASKUTIL display.</td>
</tr>
<tr>
<td>BrowseLines</td>
<td>10000</td>
<td>___ / 50 * 4</td>
<td></td>
<td>Number of lines (or records) in the largest data set you will browse. Member browse reads the entire member into storage.</td>
</tr>
<tr>
<td>ExtTrace</td>
<td>0</td>
<td>___ * 44</td>
<td></td>
<td>Will you be using the external trace function? Enter 1 if yes or 0 if no. If yes, this adds in the extra storage required. To determine if the external trace function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see whether task DSITRACE is active. The default is 1 for yes.</td>
</tr>
<tr>
<td>IntTracePgs</td>
<td>0</td>
<td>___ * 4</td>
<td></td>
<td>How many pages will you be using for NetView internal trace? The default is 0. If you start NetView internal trace and do not specify the SIZE parameter, a default of 250 pages is used. The LIST TRACE command (available in NetView V2R4 or later) can be used to determine this value.</td>
</tr>
<tr>
<td>SaveRestore</td>
<td>1</td>
<td>___ * 232</td>
<td></td>
<td>Will you be using the save/restore function? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if the save/restore function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSISVRT is active. The default is 1 for yes.</td>
</tr>
<tr>
<td>StatMon</td>
<td>1</td>
<td>___ * 240</td>
<td></td>
<td>Will you be using the status monitor? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if the status monitor is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if a task is active whose name consists of the five letter NetView domain identifier followed by the letters 'VMT' (for example, CNM01VMT). The default is 1 for yes.</td>
</tr>
<tr>
<td>VTAMnodes</td>
<td>4200</td>
<td>(_ ___ * 130) / 1024</td>
<td></td>
<td>How many VTAM nodes (major and minor) will you be monitoring with the status monitor? A minor node is a uniquely-defined resource within a major node. A major node is a set of resources that can be activated and deactivated as a group. Include all PUs and LUs defined in the VTAMLST. The status monitor main panel (domain status summary) displays this total.</td>
</tr>
</tbody>
</table>
Table 8. Base NetView Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>0</td>
<td>_____ * 92</td>
<td></td>
<td>Will you be using the alias name translation function? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task ALIASAPL is active. The default is 0 for no.</td>
</tr>
<tr>
<td>NtwkProduct</td>
<td>1</td>
<td>_____ * 76</td>
<td></td>
<td>Will you be using the network product support facility? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIGDS is active. The default is 1 for yes.</td>
</tr>
<tr>
<td>GdsDSRBs</td>
<td>6</td>
<td>_____ * 1</td>
<td></td>
<td>Specify the number of DSRBs used by task DSIGDS. This number is the sum of the DSRBO and DSRBU parameters in DSIPARM member DSICPINT. The DSRBS DSIGDS command also displays this number. The default is 6. This is applicable only if the network product support facility is used.</td>
</tr>
<tr>
<td>CSCF</td>
<td>0</td>
<td>_____ * 52</td>
<td></td>
<td>Will you be using the central site control facility (CSCF)? CSCF enables you to run remote online diagnostic tests on 3172 and 3174 devices that support this function. Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIKREM is active. The default is 0 for no.</td>
</tr>
<tr>
<td>ExternalLog</td>
<td>0</td>
<td>_____ * 56</td>
<td></td>
<td>Will you be using the external logging support facility? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIELTSK is active. The default is 0 for no.</td>
</tr>
<tr>
<td>InfoMgmt</td>
<td>0</td>
<td>_____ * 1960</td>
<td></td>
<td>Will you be using the Information/Management Link? Problems can be logged in the information/management database from NetView. Enter 1 for yes or 0 for no. If yes, the required extra storage is added. The default is 0 for no.</td>
</tr>
</tbody>
</table>
Table 8. Base NetView Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA</td>
<td>0</td>
<td>_____ * 48</td>
<td></td>
<td>Will you be using programmable network access (PNA) programs in your network? PNA is an application that runs on a workstation and acts as a gateway through which NetView commands can be issued to physical units downstream. PNA enables communication between SNA and non-SNA communication systems. Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIROVS is active. The default is 0 for no.</td>
</tr>
</tbody>
</table>
| PUCOUNT  | 100           | (_____ * 79) / 1024       |        | How many programmable network addressable (PNA) devices will register to this host? PUCOUNT determines the size of the registration table for PNA. The default is 100. To determine this value, check the PUCOUNT value in initialization member for DSIROVS (default DSIROVSI). This is applicable only if you are using the DSIROVS task (see the PNA value above). |}
| MSTransport | 1            | _____ * 132               |        | Will you be using the management services (MS) transport? The MS transport is a programming interface that enables applications in network nodes to communicate using conversations over LU 6.2 sessions. MS transport is used for focal point support, remote bridge support, and user-written applications. Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see whether task DSI6DST is active. The default is 1 for yes. |
| MSPartners | 20           | (_____ * 124) / 1024      |        | Using MS transport, how many partner LUs will communicate with NetView? Specify the total number of LUs communicating with applications using MS transport services. This is applicable only if you are using the MS transport. |
| HPTransport | 1            | _____ * 80                |        | Will you be using the NetView high performance transport? The high performance transport offers higher performance over the MS transport for applications that require a high transfer rate, or do not require send confirmations. Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIHPDST is active. The default is 1 for yes. |
Table 8. Base NetView Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPPartners</td>
<td>20</td>
<td>(_____ * 124) / 1024</td>
<td>_____</td>
<td>Using HP transport, how many partner LUs will communicate with NetView? Specify the total number of LUs communicating with applications using high performance transport services. This is applicable only if you are using the HP transport.</td>
</tr>
<tr>
<td>RMTCMD</td>
<td>1</td>
<td>_____ * 76</td>
<td>_____</td>
<td>Will you be using the RMTCMD command? RMTCMD allows you to send system, subsystem, and network commands for execution under other NetView domains. You can send only single or multiline messages, or commands that do not produce full-screen output. RMTCMD uses high performance transport services (DSIHPDST). Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task DSIUDST is active. The default is 1 for yes.</td>
</tr>
<tr>
<td>AutoTblSize</td>
<td>1000</td>
<td>(_____ * 200) / 1024</td>
<td>_____</td>
<td>How many IF-THEN and ALWAYS statements will be in the automation table? The AUTOCNT TYPE=BOTH STATS=DETAIL command (NetView V2R4 or later) has the IF-THEN and ALWAYS statements numbered sequentially for the MSG and MSU statements. Select the highest statement number from the MSG and MSU sections of the detailed report.</td>
</tr>
<tr>
<td>GlobalVars</td>
<td>50</td>
<td>used in next formula</td>
<td>_____</td>
<td>What is the total number of global variables (task and common) you will be using? Command list global variables are stored in dictionaries; one task level dictionary per operator and one common global dictionary for all operators logged onto NetView. Enter the total number of task and common global variables to be stored in all dictionaries. For NetView V2R3 or later releases, use the QRYGLOBL command to determine the number of global variables in use.</td>
</tr>
<tr>
<td>AvgVarSize</td>
<td>40</td>
<td>(GlobalVars_____ * (AvgVarSize_____ + 45)) / 1024</td>
<td>_____</td>
<td>What is the average length (in bytes) of your variables? The length of numeric variables is 4 bytes, while the length of a text variable is simply the length of the text string in bytes. Enter the average length for your common and task global variables. For NetView V2R3 or later releases, the QRYGLOBL command may help in determining this value.</td>
</tr>
<tr>
<td>UserTasks</td>
<td>0</td>
<td>_____ * 50</td>
<td>_____</td>
<td>How many user-written tasks will you have? Enter the number of user-written data services tasks (DSTS) and optional subtasks (OPTs) for your installation.</td>
</tr>
<tr>
<td>Name</td>
<td>Default Value</td>
<td>Formula</td>
<td>Result</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>------------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PreloadClists</td>
<td>0</td>
<td>used in next formula</td>
<td></td>
<td>How many command lists will you preload with the LOADCL command? Use the MAPCL command to obtain this information.</td>
</tr>
<tr>
<td>AvgClistSize</td>
<td>2000</td>
<td>(PreloadClists _____ * AvgClistSize _____) / 1024</td>
<td>_____</td>
<td>What is the average size (in bytes) of your preloaded command lists? Use the MAPCL command to obtain this information.</td>
</tr>
<tr>
<td>ResCmdProcs</td>
<td>0</td>
<td>used in next formula</td>
<td></td>
<td>How many resident user-written command procedures will you have? Command procedures are made resident if you omit the RES=N parameter on the CMDMDL statement for the command procedure in member DSICMD.</td>
</tr>
<tr>
<td>AvgResSize</td>
<td>2000</td>
<td>(ResCmdProcs _____ * AvgResSize _____) / 1024</td>
<td>_____</td>
<td>What is the average size (in bytes) of your resident user-written command procedures?</td>
</tr>
<tr>
<td>ATFs</td>
<td>0</td>
<td>used in next formula</td>
<td></td>
<td>How many user-written automation table functions (ATFs) will you have? ATFs allow function calls from the automation table to a user-written or IBM-supplied module, which can then pass and return information. When the automation table is loaded, all ATFs invoked from the table are also loaded.</td>
</tr>
<tr>
<td>AvgATFSize</td>
<td>2000</td>
<td>(ATFs _____ * AvgATFSize _____) / 1024</td>
<td>_____</td>
<td>What is the average size (in bytes) of your user-written ATFs?</td>
</tr>
<tr>
<td>Exits</td>
<td>0</td>
<td>used in next formula</td>
<td></td>
<td>How many installation exits will you have? Include both DST (XITxx) and OST (DSIEXxx) exits.</td>
</tr>
<tr>
<td>AvgExitSize</td>
<td>2000</td>
<td>(Exits _____ * AvgExitSize _____) / 1024</td>
<td>_____</td>
<td>What is the average size (in bytes) of your installation exits?</td>
</tr>
</tbody>
</table>

Table 8. Base NetView Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetView Virtual Storage</td>
<td></td>
<td>Sum of table values in Results column</td>
<td>_____</td>
<td></td>
</tr>
<tr>
<td>Save/Restore DASD Storage in kilobytes (Kb)</td>
<td>(PNA _____ * PUCOUNT _____ * 68) + (GlobalVars _____ * (52 + AvgVarSize _____)) * 2.1/1024</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9. Hardware Monitor Formulas. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HardMon</td>
<td>0</td>
<td>____ * 330</td>
<td>_____</td>
<td>Will you be using the hardware monitor? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task BNJDSERV is active. The default is 0 for no. The remainder of the parameters in this table are applicable only if hardware monitor is used.</td>
</tr>
</tbody>
</table>
Table 9. Hardware Monitor Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCACHE</td>
<td>10</td>
<td>(_____ * 500) / 1024</td>
<td>_____</td>
<td>Specify the ALCACHE value from CNMSTYLE. If ALCACHE is set to NONE, use a value of 0. If ALCACHE is set to WRAPCNT, use a value equal to the alert recording wrap count, which defaults to 100. The default is 10.</td>
</tr>
<tr>
<td>TotResources</td>
<td>5000</td>
<td>Used in DASD storage formula</td>
<td></td>
<td>Specify the number of resources in your network that could send alerts to hardware monitor.</td>
</tr>
<tr>
<td>AvgRecords</td>
<td>10</td>
<td>Used in DASD storage formula</td>
<td></td>
<td>Specify the average number of records you expect to have in the hardware monitor database, per resource. To get a conservative (high) estimate, you can specify the average wrap count used for events and statistics recording. Wrap counts are specified with the NPDA SWRAP command. The default wrap count for events and statistics recording is 25.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summary</th>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Monitor Virtual Storage</td>
<td>Sum of table values in Results column</td>
<td>_____</td>
</tr>
<tr>
<td>Hardware Monitor DASD Storage in kilobytes (Kb)</td>
<td>(500000 + TotResources_____ * (500 + AvgRecords_____ * 700)) / 1024</td>
<td>_____</td>
</tr>
</tbody>
</table>

Table 10. Session Monitor Formulas. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessMon</td>
<td>0</td>
<td>_____ * 2308</td>
<td>_____</td>
<td>Will you be using the session monitor? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task AAUTSKLP is active. The default is 0 for no. The remainder of the parameters in this table are applicable only if session monitor is used.</td>
</tr>
<tr>
<td>TsklpDSRBs</td>
<td>10</td>
<td>(6 + roundup(.5 * _____) + roundup(.75 * _____)) * 16</td>
<td>_____</td>
<td>Specify the number of DSRBOs used by task AAUTSKLP. This number is specified with the DSRBO parameter in the initialization member for AAUTSKLP (default AAUPRMLP). You can also use the DSRBS AAUTSKLP command to display this number. The default is 10.</td>
</tr>
<tr>
<td>TcnmiDSRBs</td>
<td>11</td>
<td>(_____ + 1) * 16</td>
<td>_____</td>
<td>Specify the total number of DSRBs used by task AAUTCNMI. This number is the sum of the DSRBO and DSRBU parameters in the initialization member for AAUTCNMI (default AAUCNMTD). You can also use the DSRBS AAUTCNMI command to display this number. The default is 11.</td>
</tr>
</tbody>
</table>
Table 10. Session Monitor Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTM</td>
<td>0</td>
<td>Used in session storage formulas below</td>
<td>_____</td>
<td>Is RTM (response time monitor) active? Enter 1 for yes or 0 for no. RTM is a NetView feature available with the 3x74 control unit to measure response times. To determine if RTM is used, check the initialization member for AAUTSKLP (default AAUPRMLP) to see if RTM=YES is specified. The default is 0 for no.</td>
</tr>
<tr>
<td>KEEPRTM</td>
<td>10</td>
<td>Used in session storage formulas below</td>
<td>_____</td>
<td>Specify the KEEPRTM value in the initialization member for AAUTSKLP (default AAUPRMLP), which specifies the number of collection periods retained for sessions which have RTM data collected. If you do not issue the NLD M COLLECT RTM command, RTM data is only received from the 3x74 when the session ends, and you should specify a KEEPRTM value of 1. This value is applicable only if RTM is active. The default is 10.</td>
</tr>
<tr>
<td>KEEPSESS</td>
<td>0</td>
<td>Used in DASD storage formulas below</td>
<td>_____</td>
<td>Specify the KEEPSESS value in the initialization member for AAUTSKLP (default AAUPRMLP), which is used to control DASD session recording and reduce purging session data from the database. The default is 0.</td>
</tr>
<tr>
<td>SSCP-SSCP</td>
<td>0</td>
<td>(_____ * 370) / 1024</td>
<td>_____</td>
<td>Specify the number of active SSCP-SSCP sessions. SSCP s (system services control points) activate, control, and deactivate network resources. To determine this value, use the SESSMDIS command.</td>
</tr>
<tr>
<td>SSCP-PU</td>
<td>0</td>
<td>(_____ * 370) / 1024</td>
<td>_____</td>
<td>Specify the number of active SSCP-PU sessions. To determine this value, use the SESSMDIS command.</td>
</tr>
<tr>
<td>SSCP-LU</td>
<td>0</td>
<td>(_____ * 370) / 1024</td>
<td>_____</td>
<td>Specify the number of active SSCP-LU sessions. To determine this value, use the SESSMDIS command.</td>
</tr>
<tr>
<td>LU-LU</td>
<td>0</td>
<td>(_____ * 260) / 1024</td>
<td>_____</td>
<td>Specify the number of active LU-LU sessions. To determine this value, use the SESSMDIS command.</td>
</tr>
<tr>
<td>CP-CP</td>
<td>0</td>
<td>(_____ * 370) / 1024</td>
<td>_____</td>
<td>Specify the number of active CP-CP sessions. To determine this value, use the SESSMDIS command.</td>
</tr>
<tr>
<td>Accounting</td>
<td>0</td>
<td>Accounting * 48 * (SSCP-SSCP + SSCP-PU + SSCP-LU + LU-LU + CP-CP) / 1024</td>
<td>_____</td>
<td>Is Accounting active? Enter 1 for yes or 0 for no. Session accounting data consists of session start and end times, and traffic counters for the session. If SESSSTATS=YES is specified in the initialization member for AAUTSKLP (default AAUPRMLP), session accounting data is collected for all sessions processed by the session monitor. The default is 0 for no.</td>
</tr>
</tbody>
</table>
Table 10. Session Monitor Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>0</td>
<td>Availability___ * 16 * (SSCP-SSCP___ + SSCP-PU___ + SSCP-LU___ + LU-LU___ + CP-CP___) / 1024</td>
<td></td>
<td>Is Availability active? Enter 1 for yes or 0 for no. Session availability data consists of session start and end times for the session. (Availability data is a subset of the data that would be collected if Accounting was active.) If SESSTATS=AVAIL is specified in the initialization member for AAUTSKLP (default AAUPRMLP), session availability data is collected for all sessions processed by the session monitor. The default is 0 for no. This is applicable only if Accounting is not active.</td>
</tr>
<tr>
<td>PctCrossDmn</td>
<td>80</td>
<td>(PctCrossDmn___ / 100) * LU-LU___ * 50 / 1024</td>
<td></td>
<td>Specify the percentage (between 0 and 100 inclusive) of LU-LU sessions that will be cross-domain. A cross-domain LU-LU session connects logical units in different domains. When the two LUs reside in different domains, each is owned by a different SSCP.</td>
</tr>
<tr>
<td>PctCrossNet</td>
<td>10</td>
<td>(PctCrossNet___ / 100) * LU-LU___ * 150 / 1024</td>
<td></td>
<td>Specify the percentage (between 0 and 100 inclusive) of LU-LU sessions that will be cross-network. A cross-network LU-LU session connects LUs in different networks.</td>
</tr>
<tr>
<td>PctRTM</td>
<td>50</td>
<td>(PctRTM___ / 100) * LU-LU___ * (80 + KEEPRTM___ * 20) / 1024</td>
<td></td>
<td>Specify the percentage (between 0 and 100 inclusive) of LU-LU sessions which have response time data collected. An LU-LU session will have response time data if the terminal is attached to a control unit that supports RTM and the terminal is owned by the VTAM host in which the NetView is installed. This function is not applicable if RTM is not active (see the RTM parameter above).</td>
</tr>
<tr>
<td>PctThruAPPN</td>
<td>50</td>
<td>(PctThruAPPN___ / 100) * LU-LU___ * 100 / 1024</td>
<td></td>
<td>Specify the percentage (between 0 and 100 inclusive) of LU-LU sessions which have APPN nodes in the session path. Route Selection Control Vector data is kept for those LU-LU sessions using a session path containing APPN nodes.</td>
</tr>
<tr>
<td>PctTraced</td>
<td>5</td>
<td>Used in KEEPPIU formula below</td>
<td></td>
<td>Specify the percentage (between 0 and 100 inclusive) of the total number of sessions which have PIU trace storage kept.</td>
</tr>
<tr>
<td>AvgKEEPPIU</td>
<td>7</td>
<td>(PctTraced___ / 100) * (96 + KEEPPIU___ * 48) * (SSCP-SSCP___ + SSCP-PU___ + SSCP-LU___ + LU-LU___ + CP-CP___) / 1024</td>
<td></td>
<td>Specify the average KEEPPIU value for sessions that have PIU trace data kept. A global KEEPPIU value is specified in AAUPRMLP, and KEEPPIU values can be specified for individual keep classes on the KCLASS statements in the keep member (default AAUKEEP1).</td>
</tr>
</tbody>
</table>
Table 10. Session Monitor Formulas (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessRecorded</td>
<td>25000</td>
<td>Used in DASD storage formulas below</td>
<td>Specify the total number of sessions that are recorded to the session monitor VSAM database during a typical 24-hour day. To determine this value, you can monitor the “Sessions Recorded” counter displayed by the SESSMDIS command over a 24-hour period. The total number of sessions recorded will be influenced by the amount of DASD filtering that you set up.</td>
<td></td>
</tr>
<tr>
<td>DaysKept</td>
<td>3</td>
<td>Used in DASD storage formulas below</td>
<td>Specify the number of days of activity that are retained in the database when a purge operation is performed with either the DBAUTO or NLDPM PURGE command. If you use RESETDB to clear the database, specify a value of 0.</td>
<td></td>
</tr>
<tr>
<td>DaysBetween</td>
<td>7</td>
<td>Used in DASD storage formulas below</td>
<td>Specify the maximum number of days that will pass before a database purge operation is performed or the database is redefined.</td>
<td></td>
</tr>
</tbody>
</table>

Summary Formula Result

<table>
<thead>
<tr>
<th>Session Monitor Virtual Storage</th>
<th>Sum of table values in Results column</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Session Monitor DASD Storage in kilobytes (Kb)</td>
<td>maximum of either (1100 * SessRecorded * (DaysKept + DaysBetween) * 2.1) / 1024 or (1100 * KEEPSESS * (SSCP-SSCP + SSCP-PU + SSCP-LU + LU-LU + CP-CP)) / 1024</td>
<td></td>
</tr>
</tbody>
</table>

Table 11. VSAM LSR Buffer Storage. Modify the buffer sizes and number of buffers for each size to match the values you have in CNMSJM01 (DSIZVLSR). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Buffer Size and Number</th>
<th>Default Values</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Buffers:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>7168, 4</td>
<td>CINV * BUFNO</td>
<td></td>
<td>For each buffer size in the output of the VSAMPOOL command, specify the buffer size (CINV) and number of buffers (BUFNO). See “Definitions for LSR and DFR” on page 98 for more information.</td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>8192, 20</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>16384, 4</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>18432, 20</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>20480, 20</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>24576, 20</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index Buffers:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>512, 3</td>
<td>CINV * BUFNO</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 11. VSAM LSR Buffer Storage (continued). Modify the buffer sizes and number of buffers for each size to match the values you have in CNMSJM01 (DSIZVLSR). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Buffer Size and Number</th>
<th>Default Values</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CINV, BUFNO</td>
<td>1536, 30</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>2048, 30</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>2560, 30</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>3072, 10</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>4096, 30</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CINV, BUFNO</td>
<td>4608, 30</td>
<td>CINV____ * BUFNO____</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Summary

<table>
<thead>
<tr>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSAM LSR Buffer Virtual Storage</td>
<td>Sum of table values in Results column/1024</td>
</tr>
</tbody>
</table>

Table 12. NetView Subsystem Storage. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSIBuffers</td>
<td>4200</td>
<td>(____ * 256) / 1024 + 200</td>
<td></td>
<td>Specify the total number of buffers that will be defined in your procedure to start the subsystem address space. The NetView subsystem address space is used in sending messages and commands between MVS and NetView. Enter the total of the following two statements found in sample procedure CNMPSSI: MBUF= and CBUF=.</td>
</tr>
<tr>
<td>PPIreceivers</td>
<td>1</td>
<td>Used in CSA storage formula below</td>
<td></td>
<td>Specify the number of program-to-program interface (PPI) receivers that you will have. PPI receivers are NetView-related programs and user-written application programs to route data such as generic alerts to NetView for processing by the hardware monitor, automation facilities, or installation exits. To determine this value, use the DISPPI command to display the list of PPI receivers. The storage for PPI receivers is allocated in the Common Storage Area (CSA) in 24-bit storage (below the 16 MB line).</td>
</tr>
<tr>
<td>TotalLimits</td>
<td>1000</td>
<td>(____ * 880) / 1024</td>
<td></td>
<td>Specify the total of the buffer limits defined for each PPI receiver. To determine this value, use the DISPPI command to display the list of PPI receivers and their buffer limits.</td>
</tr>
<tr>
<td>TracePPIpgs</td>
<td>0</td>
<td>____ * 4</td>
<td></td>
<td>Specify the number of pages of storage that you will use for PPI internal trace data. The storage required for the PPI internal trace is based on the user-specified size of the trace table (SIZE parameter on TRACEPPI command). The default is 10 pages.</td>
</tr>
</tbody>
</table>

Summary

<table>
<thead>
<tr>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 13. Additional NetView Storage. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AON</td>
<td>0</td>
<td>____ * 9692</td>
<td>Will you be using the Automated Operations Network (AON) feature? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS = TASKS or TASKUTIL command and see if there are some AUTxxxx or EZLxxxx tasks active. The default is 0 for no.</td>
</tr>
<tr>
<td>Controllers</td>
<td>5</td>
<td>Used in DASD storage formula below</td>
<td>Specify the number of 3600/4700 system controllers that will be defined in your network.</td>
</tr>
<tr>
<td>Counters</td>
<td>10</td>
<td>Used in DASD storage formula below</td>
<td>Specify the average number of extended statistical counters per loop. The first counter listed for each date/time is loop basic counter 2. All subsequent entries represent the extended statistical counters you define.</td>
</tr>
<tr>
<td>LOOPERR</td>
<td>24</td>
<td>Used in DASD storage formula below</td>
<td>Specify the loop error wrap count coded with the TARAWRP LOOPERR statement from the initialization member for BNJDSE36 (default BNJ36DST). Base the size for error wrap count on the anticipated solicitation interval of the data.</td>
</tr>
<tr>
<td>Loops</td>
<td>20</td>
<td>Used in DASD storage formula below</td>
<td>Specify the average number of communication loops that will be attached to each 3600/4700 controller.</td>
</tr>
<tr>
<td>LOOPSTAT</td>
<td>20</td>
<td>Used in DASD storage formula below</td>
<td>Specify the loop status wrap count coded with the TARAWRP LOOPSTAT statement from the initialization member for BNJDSE36 (default BNJ36DST). A wrap count defines the number of records kept for a certain record type. All wrap counts must be within the range of 1–9999.</td>
</tr>
<tr>
<td>MSM</td>
<td>0</td>
<td>____ * 7824</td>
<td>Will you be using the MultiSystem Manager? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS = TASKS or TASKUTIL command and see if task AUTOMSM or AUTOMSMD is active. The default is 0 for no.</td>
</tr>
</tbody>
</table>
Table 13. Additional NetView Storage (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSTs</td>
<td>5</td>
<td>____ * (340 + 50 (if session monitor active) + 58 (if hardware monitor active) + 58 (if 4700 support facility active) + 8 (if status monitor active) + 14 (if TAF used))</td>
<td></td>
<td>How many operators will be concurrently logged on to your NetView system? An operator station task (OST) is the subtask that establishes and maintains an online session with the network operator. Use the TASKUTIL command (available in NetView V2R3 or later) to display the active NetView operators on your system. Add the number of tasks with TYPE=OST shown in the TASKUTIL display.</td>
</tr>
<tr>
<td>RESPTIME</td>
<td>24</td>
<td>Used in DASD storage formula below</td>
<td></td>
<td>Specify the response time wrap count coded with the TARAWRP RESPTIME statement from the initialization member for BNJ36DSE36 (default BNJ36DST). Base the size for response time wrap count on the anticipated solicitation interval of the data.</td>
</tr>
<tr>
<td>StatFocalPt</td>
<td>0</td>
<td>____ * 852</td>
<td></td>
<td>Will this host be a status focal point for the NetView management console? Enter 1 for yes or 0 for no. If yes, additional storage is added. Check the initialization member for CNMTAMEL task specified in member DSISTASK. If an AMELINIT statement is enabled, this host is a status focal point. The default is 0 for no.</td>
</tr>
<tr>
<td>SNATM</td>
<td>0</td>
<td>____ * 572</td>
<td></td>
<td>Will you be using the SNA Topology Manager (SNATM)? SNATM obtains the status and topology information of SNA resources from the VTAM topology agent (introduced in VTAM V4R3) for graphical display using NetView management console. Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task FLBTOPO is active. The default is 0 for no.</td>
</tr>
<tr>
<td>SnatmObjects</td>
<td>4000</td>
<td>____ * 1.3</td>
<td></td>
<td>Specify the number of objects managed by SNATM in your network. To determine this value, use the TOPOSNA LISTRODM command to determine the total number of objects that SNATM is managing in RODM. This is not applicable if you are not using SNATM. Storage for SNATM objects in the RODM data space is included in Table 17 on page 151.</td>
</tr>
</tbody>
</table>

**Note:** If you do not have SNATM active for using the TOPOSNA LISTRODM command, see “SNA Topology Manager” on page 123 for more information.
Table 13. Additional NetView Storage (continued). All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAF</td>
<td>1</td>
<td>Used in OST formula above</td>
<td></td>
<td>Will your operators be using the terminal access facility (TAF)? TAF is a facility that allows a network operator to control several subsystems from one NetView terminal. Enter 1 for yes or 0 for no. If yes, additional storage is added for each OST. The default is 1 for yes.</td>
</tr>
<tr>
<td>TARA</td>
<td>0</td>
<td>____ * 84</td>
<td></td>
<td>Will you be using the 4700 support facility (TARA)? Enter 1 for yes or 0 for no. If yes, this adds in the extra storage required. To determine if this function is used, use either the LIST STATUS=TASKS or TASKUTIL command and see if task BNJDSE36 is active. The default is 0 for no. The remainder of the parameters in this table are applicable only if the 4700 support facility is used.</td>
</tr>
<tr>
<td>VOSTs</td>
<td>0</td>
<td>____ * 200</td>
<td></td>
<td>How many Virtual OSTs will be concurrently logged on to your NetView system? A virtual operator station task (VOST) is created as the result of the ATTCH phase during the full screen automation process. Use the TASKUTIL command (available in NetView V2R3 or later) to display the active virtual OSTs on your system. Add the number of tasks with TYPE=VOST shown in the TASKUTIL display.</td>
</tr>
<tr>
<td>Workstat</td>
<td>20</td>
<td>Used in DASD storage formula below</td>
<td></td>
<td>Specify the average number of workstations that will be physically attached to each financial system controller.</td>
</tr>
</tbody>
</table>

**Summary**

<table>
<thead>
<tr>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional NetView Virtual Storage</td>
<td>Sum of table values in Results column</td>
</tr>
<tr>
<td>4700 Support Facility DASD Storage in kilobytes (Kb)</td>
<td>Controllers* (Loops* (LOOPSTAT* 72 + 224 + LOOPERR* (Counters* 23 + 76) + Workstat* (RESPTIME* 91 + 127) + 54)) / 1024</td>
</tr>
</tbody>
</table>
**Table 14. GMFHS Address Space.** GMFHS requires that RODM be active. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMFHS</td>
<td>0</td>
<td>_____ * 6783</td>
<td>_____</td>
<td>Will you be using the Graphic Monitor Facility Host Subsystem (GMFHS)? Enter 1 for yes or 0 for no. GMFHS is an advanced network management system which employs task-oriented dialogs in an intelligent workstation providing a graphics base for IBM systems and network management products. The default is 0 for no. If you will not be using GMFHS, the remainder of the parameters in this table are not applicable.</td>
</tr>
<tr>
<td>GmfhsObjects</td>
<td>1000</td>
<td>Used in RODM data space virtual storage formula. See Table 17 on page 151</td>
<td></td>
<td>Specify the number of GMFHS-managed objects that you expect to have in RODM.</td>
</tr>
<tr>
<td>GmfhsTrcPgs</td>
<td>0</td>
<td>_____ * 4</td>
<td>_____</td>
<td>Specify the number of pages of storage used for GMFHS internal trace. Trace pages are used to record GMFHS data for test and debug purposes. The TRACEPAGES parameter is specified in the DUIGINIT member and works in conjunction with TRACE=ON.</td>
</tr>
</tbody>
</table>

**Summary**  
GMFHS Address Space Virtual Storage above the 16 MB Line  
Sum of table values in Results column  

**Table 15. Event/Automation Service Address Space.** All numbers in the results column are in kilobytes (KB).

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ev/Aut</td>
<td>0</td>
<td>_____ * 5852</td>
<td>_____</td>
<td>Will you be using the Event/Automation Service (IHSAEVNT)? Enter 1 for yes or 0 for no. The Event/Automation Service enables you to effectively manage both distributed and z/OS events from a single interface on a single console. You can exploit the event correlation and automation capabilities of the Tivoli Enterprise Console® for all events in the enterprise. The default is 0 for no.</td>
</tr>
</tbody>
</table>

**Summary**  
Event/Automation Service Address Space Virtual Storage  
Sum of table values in Results column  

Chapter 11. Storage Considerations  149
Table 16. RODM Address Space Storage. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RODM</td>
<td>0</td>
<td>_____ * 49055</td>
<td>_____</td>
<td>Will you be using the Resource Object Data Manager (RODM)? Enter 1 for yes or 0 for no. RODM is an object-oriented data store which supports on-line, real-time collection and access of data, mainly network and system management data. RODM is required in order to use the Graphic Monitor Facility Host Subsystem and the SNA Topology Manager. The default is 0 for no. If you will not be using RODM, the remainder of the parameters in this table are not applicable.</td>
</tr>
<tr>
<td>ConcurUsers</td>
<td>10</td>
<td>Used in formulas below</td>
<td>_____</td>
<td>Specify the number of RODM users connected concurrently. To determine this value, check member EKGCUST for the CONCURRENT_USERS value. Allow room for growth, but do not waste storage unnecessarily.</td>
</tr>
<tr>
<td>AsyncTasks</td>
<td>5</td>
<td>Used in formulas below</td>
<td>_____</td>
<td>Specify the number of asynchronous RODM tasks. To determine this value, check member EKGCUST for the ASYNC_TASKS value. Allow room for growth but do not waste storage unnecessarily.</td>
</tr>
<tr>
<td>PLI_ISA</td>
<td>40</td>
<td>PLI_ISA_____ *</td>
<td>_____</td>
<td>Specify the amount of stack storage in KB that will be allocated for each RODM user. To determine this value, check member EKGCUST for the PLI_ISA value. This is storage below the 16 MB line set aside for PL/I intermodule communication. It is recommended that 40 K be used for faster run time with minimum wasted space.</td>
</tr>
<tr>
<td>PrimaryHeap</td>
<td>64</td>
<td>PrimaryHeap_____ *</td>
<td>_____</td>
<td>Specify the amount of primary PL/I heap storage in KB that will be allocated for each RODM user. To determine this value, check member EKGCUST for the PRIMARY_HEAP_SIZE value. This is temporary storage allocated by RODM modules for work area. It is recommended that 64 K be used for faster run time with minimum wasted space.</td>
</tr>
<tr>
<td>LogRecords</td>
<td>30000</td>
<td>Used in the following DASD storage formula.</td>
<td>_____</td>
<td>Specify the number of records you want to keep in each log (primary and secondary).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summary</th>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>RODM Address Space Virtual Storage</td>
<td>Sum of table values in Results column</td>
<td>_____</td>
</tr>
<tr>
<td>RODM Storage in CSA below the line</td>
<td>64 * (ConcurUsers_____ + AsyncTasks_____ ) / 1024</td>
<td>_____</td>
</tr>
<tr>
<td>RODM Log DASD Storage in kilobytes (Kb)</td>
<td>LogRecords_____ / 2</td>
<td>_____</td>
</tr>
</tbody>
</table>
Table 17. RODM Object Storage. All numbers in the results column will be in kilobytes (KB). These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserObjects</td>
<td>0</td>
<td>Used in formula below</td>
<td></td>
<td>Specify the number of RODM objects created by user-written RODM applications. To determine this value, use the RODMUNLD facility provided with Tivoli NetView for z/OS with the REPORTONLY=YES option to get a detailed report of objects residing in RODM. For prior NetView releases, use the Create Object API counters in the STATAPI record. Create a STATAPI record prior to loading the user objects, and a second STATAPI record after loading the user objects. The differences in the Create Object counters between the two readings can be used to estimate the total number of user objects created. See &quot;RODM API Statistics&quot; on page 89 for more detail on the STATAPI record.</td>
</tr>
<tr>
<td>UserObjSize</td>
<td>3072</td>
<td>(UserObjects * UserObjSize) / 1024</td>
<td></td>
<td>Specify the average size (in bytes) for the RODM objects created by user-written RODM applications. To determine this value, use cell pool usage information in the STATCELL record. Create a STATCELL record prior to loading the user objects, and a second STATCELL record after loading the user objects. The differences in cell pool usage between the two readings can be used to calculate the total storage used by the user objects.</td>
</tr>
</tbody>
</table>

Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>RODM Address Space</td>
<td>13005 + (SnatmObjects * 0.3) + MSM-managed address space storage (see Table 18)</td>
<td></td>
</tr>
<tr>
<td>Virtual Storage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RODM Data Space</td>
<td>UserObject storage (see previous formula) + 3686 + (GmfhsObjects * 3) + (SnatmObjects * 3.4)</td>
<td></td>
</tr>
<tr>
<td>Virtual Storage</td>
<td>(see Table 13 on page 146) + MSM-managed data space storage (see Table 18)</td>
<td></td>
</tr>
<tr>
<td>RODM Checkpoint Data</td>
<td>RODM Data Space Virtual Storage for objects + 16384</td>
<td></td>
</tr>
<tr>
<td>Set DASD Storage in</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kilobytes (KB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RODM Translation Data</td>
<td>RODM Address Space Virtual Storage for objects + 16384</td>
<td></td>
</tr>
<tr>
<td>Set DASD Storage in</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kilobytes (KB)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 18. RODM Data and Address Space Storage for MSM-managed Resources. All numbers in the Results column will be in kilobytes (KB).

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Data Space Result</th>
<th>Addr Space Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LNM Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine which network resources will be managed by LNM and complete the following:</td>
</tr>
</tbody>
</table>

Chapter 11. Storage Considerations  151
<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Data Space Result</th>
<th>Addr Space Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LNM Agent</td>
<td>0</td>
<td>((_____ * 11120) + 18156)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of LNM agents.</td>
</tr>
<tr>
<td>Bridges</td>
<td>0</td>
<td>((_____ * 588) + 1950028)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of bridges managed by LNM.</td>
</tr>
<tr>
<td>Segments</td>
<td>0</td>
<td>(_____ * 9100)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of segments managed by LNM.</td>
</tr>
<tr>
<td>Adapters</td>
<td>0</td>
<td>((_____ * 7005) + 13713)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of adapters managed by LNM.</td>
</tr>
<tr>
<td>CAUs</td>
<td>0</td>
<td>((_____ * 8296) + 12548)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of CAUs managed by LNM.</td>
</tr>
<tr>
<td>IP Networks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine which network resources will be managed by NetView for UNIX (IP) and complete the following:</td>
</tr>
<tr>
<td>IP Agents</td>
<td>0</td>
<td>((_____ * 16076) + 27356)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of IP agents.</td>
</tr>
<tr>
<td>IP Resources</td>
<td>0</td>
<td>((_____ * 4139) + 426164)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of resources managed by NetView for UNIX. Resources include bridges, routers, hubs, hosts, IP links, IP addresses.</td>
</tr>
<tr>
<td>NetFinity networks:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine which network resources will be managed by NetFinity and complete the following:</td>
</tr>
<tr>
<td>NF Agents</td>
<td>0</td>
<td>((_____ * 15744) + 7892)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of NetFinity agents.</td>
</tr>
<tr>
<td>NF workstations</td>
<td>0</td>
<td>((_____ * 5780) + 128087)/1024</td>
<td>____</td>
<td></td>
<td>Specify the number of workstations managed by NetFinity.</td>
</tr>
</tbody>
</table>

Table 18. RODM Data and Address Space Storage for MSM-managed Resources (continued). All numbers in the Results column will be in kilobytes (KB).
Table 18. RODM Data and Address Space Storage for MSM-managed Resources (continued). All numbers in the Results column will be in kilobytes (KB).

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Formula</th>
<th>Data Space Result</th>
<th>Addr Space Result</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Process appls</td>
<td>0</td>
<td>(_ * 4748) + 90593/1024</td>
<td></td>
<td></td>
<td>Specify the number of process applications running on workstations managed by NetFinity.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 394) + 506695/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security appls</td>
<td>0</td>
<td>(_ * 5262) + 41347/1024</td>
<td></td>
<td></td>
<td>Specify the number of security applications running on workstations managed by NetFinity.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 392) + 231257/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sys_mon appls</td>
<td>0</td>
<td>(_ * 4050) + 31034/1024</td>
<td></td>
<td></td>
<td>Specify the number of system monitoring applications running on workstations managed by NetFinity.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 336) + 557072/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TME networks:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine which network resources are managed by TME and complete the following:</td>
<td></td>
</tr>
<tr>
<td>TME Agents</td>
<td>0</td>
<td>(_ * 15216) + 13608/1024</td>
<td></td>
<td></td>
<td>Specify the number of TME agents.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 740) + 1671540/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMR’s</td>
<td>0</td>
<td>(_ * 5280) + 48592/1024</td>
<td></td>
<td></td>
<td>Specify the number of Tivoli Management Regions (TMR).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 872) + 1115256/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Policy Regions</td>
<td>0</td>
<td>(_ * 2128) + 8484/1024</td>
<td></td>
<td></td>
<td>Specify the number of Tivoli Policy Regions.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 176) + 278496/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Managed Nodes</td>
<td>0</td>
<td>(_ * 5714)/1024</td>
<td></td>
<td></td>
<td>Specify the number of TME Managed Nodes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 136)/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitors</td>
<td>0</td>
<td>(_ * 3214)/1024</td>
<td></td>
<td></td>
<td>Specify the number of Monitors at TME Managed Nodes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(_ * 192)/1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sum of table values in Results column</td>
<td></td>
</tr>
</tbody>
</table>

Chapter 11. Storage Considerations 153
Table 19. Minimum Virtual Storage Requirements Summary. All numbers in the Results column are in kilobytes (KB). To convert values to megabytes (MB), divide by 1024. These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Storage Type</th>
<th>Table Location/Formula/Amount</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetView Address Space Above the 16 MB Line</td>
<td>See Table 8 on page 135, Table 9 on page 140, Table 10 on page 141, Table 11 on page 144, and Table 13 on page 146.</td>
<td></td>
</tr>
<tr>
<td>NetView Address Space Below the 16 MB Line</td>
<td>((#Autotasks + #NNTs + #UserTasks + #OSTs + #VOSTs) * 2) + 538</td>
<td></td>
</tr>
<tr>
<td>NetView Subsystem Address Space Above the 16 MB Line</td>
<td>See Table 12 on page 145.</td>
<td></td>
</tr>
<tr>
<td>NetView Subsystem Address Space Below the 16 MB Line</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>RODM Address Space Above the 16 MB Line</td>
<td>Sum of results in Table 16 on page 150 + Address Space object storage (see Table 17 on page 151).</td>
<td></td>
</tr>
<tr>
<td>RODM Address Space Below the 16 MB Line</td>
<td>Environment Dependent</td>
<td></td>
</tr>
<tr>
<td>RODM Data Space Above the 16 MB Line</td>
<td>Data Space object storage (see Table 17 on page 151).</td>
<td></td>
</tr>
<tr>
<td>RODM Data Space Below the 16 MB Line</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>GMFHS Address Space Above the 16 MB Line</td>
<td>692 + See (GMFHSTrcPgs in Table 14 on page 149 * 4).</td>
<td></td>
</tr>
<tr>
<td>GMFHS Address Space Below the 16 MB Line</td>
<td>7084</td>
<td></td>
</tr>
<tr>
<td>Event Automation Service Above the 16 MB Line</td>
<td>4800</td>
<td></td>
</tr>
<tr>
<td>Event Automation Service Below the 16 MB Line</td>
<td>2400</td>
<td></td>
</tr>
<tr>
<td>CSA Storage Below the 16 MB Line</td>
<td>See Table 16 on page 150 and Table 12 on page 145.</td>
<td></td>
</tr>
</tbody>
</table>

Table 20. DASD Storage Conversion Table. Use to convert kilobytes of storage into Cylinders or Blocks. These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>DASD Type</th>
<th>Kilobytes per unit</th>
<th>Unit Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>3390</td>
<td>720</td>
<td>Cylinders</td>
</tr>
<tr>
<td>3375</td>
<td>384</td>
<td>Cylinders</td>
</tr>
<tr>
<td>3330</td>
<td>228</td>
<td>Cylinders</td>
</tr>
<tr>
<td>9335</td>
<td>0.5</td>
<td>Blocks</td>
</tr>
<tr>
<td>9332</td>
<td>0.5</td>
<td>Blocks</td>
</tr>
<tr>
<td>3370</td>
<td>0.5</td>
<td>Blocks</td>
</tr>
<tr>
<td>3310</td>
<td>0.5</td>
<td>Blocks</td>
</tr>
</tbody>
</table>
Table 21. Minimum DASD Storage Requirements Summary. Use the DASD conversion table to convert storage in KB into blocks or cylinders. These formulas are implemented in the V5R1HOST.WK1 spreadsheet, which is on the diskette accompanying this book.

<table>
<thead>
<tr>
<th>Storage Type</th>
<th>Table Location</th>
<th>Device Type</th>
<th>Formula</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session Monitor</td>
<td>See Table 10 on page 141</td>
<td>primary</td>
<td>Round up (StorageInKb / KbPerUnit) and use minimum of 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>secondary</td>
<td>Round up (0.2 * Session Monitor primary result)</td>
<td></td>
</tr>
<tr>
<td>Hardware Monitor</td>
<td>See Table 9 on page 140</td>
<td>primary</td>
<td>Round up (StorageInKb / KbPerUnit) and use minimum of 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>secondary</td>
<td>Round up (0.2 * Hardware Monitor primary result)</td>
<td></td>
</tr>
<tr>
<td>4700 Support</td>
<td>See Table 13 on page 146</td>
<td>Facility primary</td>
<td>Round up (StorageInKb / KbPerUnit) and use minimum of 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Facility secondary</td>
<td>Round up (0.2 * 4700 Support Facility primary result)</td>
<td></td>
</tr>
<tr>
<td>Save/Restore</td>
<td>See Table 8 on page 135</td>
<td>primary</td>
<td>Round up (StorageInKb / KbPerUnit) and use minimum of 2</td>
<td></td>
</tr>
<tr>
<td>RODM Log</td>
<td>See Table 16 on page 150</td>
<td>primary</td>
<td>Round up (StorageInKb / KbPerUnit) and use minimum of 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>secondary</td>
<td>Same as RODM Log primary</td>
<td></td>
</tr>
<tr>
<td>RODM Translation Data Set</td>
<td>See Table 17 on page 151</td>
<td></td>
<td>Round up (Address Space StorageInKb / 1024 / 16 * 16 * 1024 / KbPerUnit)</td>
<td></td>
</tr>
<tr>
<td>RODM Checkpoint Data Set 1</td>
<td>See Table 17 on page 151</td>
<td></td>
<td>Round up (round up (Data Space StorageInKb / 1024 / 16) / 2) * 16 * 1024 / KbPerUnit</td>
<td></td>
</tr>
<tr>
<td>RODM Checkpoint Data Set 2</td>
<td></td>
<td></td>
<td>Same as RODM Checkpoint Data Set 1</td>
<td></td>
</tr>
</tbody>
</table>

Region Size

The REGION parameter on the EXEC statement in the startup procedure for a program specifies how much virtual storage the program is enabled to allocate. Specifying various values for the region size has the following results (unless your installation changes IBM-supplied default limits in MVS):

- A value equal to 0 MB allows the program to allocate all of the storage it requires below and above the 16 MB line.
- A value greater than 0 MB and less than or equal to 16 MB establishes the size of the private area below 16 MB. The extended region size (above the 16 MB line) is the default value of 32 MB.
- A value greater than 16 MB and less than or equal to 32 MB gives the program all the storage available below 16 MB. The extended region size is the default value of 32 MB.
A value greater than 32 MB gives the program all the storage available below 16 MB. The extended region size is the specified value.

The default REGION size in the NetView startup procedure is 32768 K or 32 MB. Consider changing the REGION size to 0 MB, which enables the NetView program to allocate the storage it needs and minimizes the risk of out-of-storage failures. If you do not want to use a value of 0 MB, use no less than 32 MB so that you do not restrict the use of NetView storage usage below the 16 MB line. Restricting NetView’s storage below the 16 MB line does not help other jobs in the system, and could cause out-of-storage failures unnecessarily.

See “Estimating Storage Usage” on page 135 to estimate the amount of virtual storage the NetView program requires for your environment.

For more information on the REGION parameter, Refer to the appropriate MVS JCL publication.

Estimating the Number of RODM Objects Created by SNA Topology Manager

The information below should help you estimate the number of RODM objects that will be created by SNA topology manager for your environment. The estimate will not be exact, but it should be useful to help you plan the additional storage requirements that SNA topology manager will introduce.

Two sections follow, one to estimate objects created by local and network topology monitor requests, and one to estimate LU objects created by critical LU and LU collection monitor requests. Sum the number of objects that you calculate from these two sections.

The two sections that follow assume that the requests are issued to a VTAM agent. If you are issuing local topology requests to a non-VTAM agent such as the APPN Topology and Accounting Agent (APPNTAA). In general, use 10 to 15 objects per locally monitored node.

Objects Created by Local and Network Topology Monitor Requests

The formula below is based on the following assumptions:

• J-lines are suppressed and only CDRSCs that represent adjacent CPs are reported (consistent with the default VTAM options ILUCRDCS and NOLLINES).
• There is a single TG to PU 2.1 nodes (appnENs and lenNodes).
• All nodes are connected through NCPs.
• There are 2 subarea TGs between subarea nodes.
• There are 2 APPN TG circuits between network nodes.

Specify values for the following variables in the formula:

TotalNNs

The value of network nodes to be monitored. Estimate this using the following:

D NET,STATS,TYPE=VTAM ID 109, NETWORK NODES IN THE NETWORK

Note: If you receive message IST1315I indicating that the output has been truncated, use the NUM parameter to specify more lines of output.
for the DISPLAY STATS command. For a description of the DISPLAY
STATS command, refer to the appropriate VTAM publication.

Subareas
The number of value type 4 and 5 nodes in the network. Estimate this
using the following:
D NET,STATS,TYPE=VTAM ID 22, DESTINATION SUBAREAS

If local and network topology monitor requests will be sent to multiple
VTAMs in the network, combine the DESTINATION SUBAREAS values for
each VTAM. However, because multiply-owned T4s are represented by
a single object in RODM, do not count multiply-owned T4s more than once.

TotalLines
The number of lines in the network. Estimate this using the following:
D NET,STATS,TYPE=VTAM ID 65, NUMBER OF LINES DEFINED

If local and network topology monitor requests will be sent to multiple
VTAMs in the network, combine the NUMBER OF LINES DEFINED values
for each VTAM. However, because multiply-owned lines are represented
by a single object in RODM, do not count multiply-owned lines more than
once. Refer to "Tivoli NetView for z/OS SNA Topology Manager Implementation
Guide" for more information on multiply-owned lines.

NTRILines
The number of NTRI lines (J-lines) in the network. You should be able to
get a count of these lines using the following command:
D NET,RSCLIST,ID=J*,IDTYPE=LINES

This command produces a separate message for each line, so if you think
you have a large number, do not issue this command during a peak period
of activity on your system.

TotalPUs
The number of PUs in the network. Estimate this using the following:
D NET,STATS,TYPE=VTAM ID 48, DEFINED PU TOTAL

If local and network topology monitor requests will be sent to multiple
VTAMs in the network, the DEFINED PU TOTAL values for each VTAM
should be combined. However, because multiply-owned PUs are
represented by a single object in RODM, be sure not to count
multiply-owned PUs more than once. Refer to "Tivoli NetView for z/OS SNA
Topology Manager Implementation Guide" for more information on multiply
owned PUs.

NTRILogicalPUs
The number of NTRI logical links PUs in the network. Because there is a
one logical PU per NTRI line, this value should be equal to the value for
NTRILines above.

PctPU2.1
The percentage of the total PUs (not including NTRI logical PUs) that are
type 2.1. Estimate this value, because it is not reported. The following
command can provide information to help you with an estimate:
D NET,RSCLIST

Using values for the above variables, estimate the number of RODM objects
created by local and network topology requests to VTAM agents as follows:
LU Objects Created Critical LU and LU Collection Monitor Requests

The following variables are used to estimate the number of objects created by LU collection requests.

If you plan to issue LU collection requests to VTAM hosts, specify values for the next three variables; otherwise, use values of zero for these variables.

LocalNonSNA
The number of local non-SNA terminals defined on VTAMs targeted for LU collection requests. Estimate this value using the following:
D NET,STATS,TYPE=VTAM ID 16, LOCAL NON-SNA TERMINALS

If LU collection requests will be sent to multiple VTAMs in the network, and VTAM is the target of an LU collection request, the LOCAL NON-SNA TERMINALS values for each VTAM should be summed together.

IndLUs
The number of independent LUs in the network for which VTAM provides boundary function services, and where VTAM is the target of an LU collection request. Estimate this using the following:
D NET,STATS,TYPE=VTAM ID 46, INDEPENDENT LU TOTAL

If you want to send LU collection requests to multiple VTAMs in the network, the INDEPENDENT LU TOTAL values for each VTAM should be summed together.

APPLs
The number of application programs defined on the VTAMs specified for LU collection requests. To get this value, use the following command:
D NET,APPLS

The last message in the D NET,APPLS display (IST1454I) contains the number of resources displayed. If you want to send LU collection requests to multiple VTAMs in the network, and VTAM is the target of an LU collection request, add the number of applications for each VTAM.

If you plan to issue LU collection requests to logical links (PUs) in the network, supply a value for the next variable; otherwise, use a zero value.

LogLinkLUs
The expected number of LUs that will be in concurrent LU collection requests issued to logical links (PUs) in the network. In most environments, this value represents a small percentage of LUs in the network.

If you plan to issue TOPOSNA CRITICAL requests for specific LUs in the network, supply a value for the next variable; otherwise, use a zero value.

CriticalLUs
The expected number of LUs in the network that will be the target of concurrent TOPOSNA CRITICAL monitor requests. In most environments, this value represents a small percentage of LUs in the network.
Using values for the above variables, estimate the number of LU objects created in RODM using the following formula:

\[ \text{LocalNonSNA} + \text{IndLUs} + \text{APPLs} + \text{LogLinkLUs} + \text{CriticalLUs} \]

Add the estimates for the number of LU objects and the number of objects created by local and network topology requests to determine the number of SNA topology manager objects in RODM.

**Keeping Track of Virtual Storage and Other System Resource Usage**

To collect statistics that help you tune your systems’ performance and avoid costly system shutdowns, periodically check the amount of virtual storage, CPU, and other resources your tasks are using. For an accurate picture of the resource usage, you probably need to track usage for an entire business processing cycle. For example, you might want to track usage for a full month to see the usage peaks for applications and tasks that are not run daily.

To collect statistics on virtual storage, CPU usage, message queuing, and input/output activity for a system, issue the following timed LOGTSTAT command from a NetView command line:

```
EVERY 00:30:00,PPT,LOGTSTAT
```

The example command writes the results to the system monitoring facility (SMF) log. When you have collected data for the number of days you want, turn off the EVERY command by issuing a PURGE TIMER command with the timer ID of the original EVERY command. (If you do not know the timer ID, you can find it by issuing a LIST TIMER command.)

After the statistics are collected in the SMF log, use that data to determine the peak resource usage for various tasks. Use the NetView function that enables you to issue the DEFAULTS and OVERRIDE commands to set limits on resource usage for each critical task. When limits are set, NetView sends a warning message when one of the limits is reached or exceeded by a task. You will have time to react before the task can use enough virtual storage or CPU time to slow down or halt your system. You can even automate your responses to the warning messages.

To do a quick check of resource usage (for example, for a day or two), use the following command instead of setting an EVERY timer:

```
WINDOW TASKMON * *
```

This command sends the TASKMON output to a window at the operator’s console where the command is issued. It monitors virtual storage, CPU, message queuing, and I/O activity for every task for which you have set a limit. The results are color-coded, with tasks that are at 90% of their limits shown in red.

To monitor only the virtual storage your tasks are using, issue the following TASKUTIL command each morning:

```
EVERY 30, TASKUTIL
```

This command checks the virtual storage usage every 30 minutes for all tasks. You can vary the number of minutes.

The TASKMON and TASKUTIL outputs provide a baseline from which to compare future data when you suspect a task may be using an unusual amount of virtual storage. For example, suppose a normal amount of virtual storage usage for one of
your NetView operator tasks is about 2 MB. If the operator ROLLs from one NetView window to another, the operator task could quickly take up to 20 MB of storage. Also, the IBM NCP NTuneMon product can use a large amount of virtual storage. If an operator is checking out two or three NCPs at the same time, it could use in the range of 12–15 MB of virtual storage.

When you have collected data for the specified number of days, turn off the EVERY command by issuing a PURGE TIMER command with the timer ID of the original EVERY command. (If you do not know the timer ID, you can find it by issuing a LIST TIMER command.)

**Hint:** If your TASKUTIL output at 8 a.m. says 70 MB and your REGION size is 75 MB, attempt to determine immediately what is using the most storage. The amount of virtual storage being used will surely increase when operators start logging on for the day.

Messages warn you when the NetView region is running out of space. You can automate responses to the following messages:

```plaintext
BNH162I THE domainid BELOW 16M STORAGE IS nn% USED, mmmK IS LEFT
BNH163I THE domainid ABOVE 16M STORAGE IS nn% USED, mmmK IS LEFT
```

**Minimizing Storage Usage**

If you need to save storage, you have several alternatives. When you optimize for storage, you almost always have a trade-off between storage and host processor utilization.

The constants module (DSICTMOD) enables you to choose whether the first allocation of below-the-line storage for an individual subpool and size is freed when it is no longer in use. You can use the RESOURCE command to display the amount of storage in use below-the-line. If the amount of below-the-line storage used by the NetView program is a problem, set DSICTMOD to free unused storage, although performance for subsequent uses will be slowed.

If you have fewer than 300 users logged on at any one time, or if you have a large amount of user-written code that runs in below-the-line storage, choose the option that keeps the first allocation of below-the-line storage (the default).

**Coding RES=N on Command Model Statements**

If you are operating a severely storage-constrained system, you can save small amounts of storage by coding RES=N on specific CMDMDL statements in DSICMD. When a module has RES=N specified in DSICMD, the module must be dynamically loaded each time it is used. If you use the module infrequently, the trade-off between saved storage and added processor time is minimal. For frequently used modules, coding RES=N is not recommended because it can result in excessive CPU use and I/O activity.

This section lists the command model (CMDMDL) statements that you can code as RES=N to save storage.

**Alert Network Operations Support**

```plaintext
DSIREGGR CMDMDL MOD=DSIREGGR,TYPE=R,RES=N
DSILOGGR CMDMDL MOD=DSILOGGR,TYPE=R,RES=N
```
Coding RES=N on the CMDMDL statements for alert network operations support saves 2 KB.

**ALLOCATE, FREE, and LISTA**

ALLOCATE  CMDMDL  MOD=DSIALLOC,RES=N
FREE     CMDMDL  MOD=DSIUNALL,RES=N
LISTA    CMDMDL  MOD=DSILIISTA,RES=N

Coding RES=N on the CMDMDL statements for these functions saves 19 KB.

**Automated Operations**

AUTOTASK  CMDMDL  MOD=DSIAUTOT,TYPE=R,CTL=N,RES=N
GENALERT CMDMDL  MOD=BNJGENAL,TYPE=R,CTL=N,RES=N
DEFAULTS CMDMDL  MOD=DSIDEFAU,TYPE=R,CTL=N,RES=N
OVERRIDE  CMDMDL  MOD=DSIOVERR,TYPE=R,CTL=N,RES=N
GETMSIZE  CMDMDL  MOD=DSIGTMSZ,TYPE=R,CTL=N,RES=N
GETMTYPE  CMDMDL  MOD=DSIGTMTY,TYPE=R,CTL=N,RES=N
GETMLINE  CMDMDL  MOD=DSIGTMLN,TYPE=R,CTL=N,RES=N
PARSEL2R  CMDMDL  MOD=DSIPRL2R,TYPE=R,CTL=N,RES=N
EXCMD    CMDMDL  MOD=DSIEXCMD,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for automated operations saves 24 KB.

**Automatic Operations (MVS only)**

MVS     CMDMDL  MOD=CNMCMVS,TYPE=R,CTL=N,RES=N
WTO    CMDMDL  MOD=DSIWTC,TYPE=R,CTL=N,RES=N
WTOCR CMDMDL  MOD=DSIWTCR,TYPE=R,CTL=N,RES=N
DOM    CMDMDL  MOD=DSIWTCD,TYPE=R,CTL=N,RES=N
RELCONID CMDMDL  MOD=CNMCRCID,TYPE=R,CTL=N,RES=N
DISCONID CMDMDL  MOD=CNMCDCID,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for automated operations (MVS) saves 18 KB.

**Automation Table Listing**

DSIDTEND CMDMDL  MOD=DSIDTEND,TYPE=RD,PARSE=N,RES=N

Coding RES=N on the CMDMDL statements for this function saves 0.7 KB.

**CNM Router**

DSICRUSP CMDMDL  MOD=DSICRUSP,TYPE=D,PARSE=N,RES=N
CHNGFP CMDMDL  MOD=DSICSDN,TYPE=R,RES=N
FPDLCMD CMDMDL  MOD=DSICDLCP,TYPE=R,PARSE=Y,RES=N

Coding RES=N on the CMDMDL statements for the CNM router saves 5 KB.

**Commands**

LIST      CMDMDL  MOD=DSISHP,RES=N
MSG       CMDMDL  MOD=DSIMG,RES=N
HOLD      CMDMDL  MOD=DSIHHP,RES=N,TYPE=B,ECHO=N
INPUT     CMDMDL  MOD=DSIINP,RES=N
PURGE     CMDMDL  MOD=DSIPRP,RES=N
ROUTE     CMDMDL  MOD=DSIRTP,RES=N

Coding RES=N on these CMDMDL statements saves 12 KB.

**Cross-Domain CNM Data Transfer**
Coding RES=N on the CMDMDL statements for data transfer saves 11 KB.

Cross-Domain Logon Facilities

Coding RES=N on the CMDMDL statements for cross-domain logon saves 4 KB.

DCNM Router Function

Coding RES=N on the CMDMDL statements for the DCNM router saves 13 KB.

External Logging

Coding RES=N on the CMDMDL statements for external logging saves 1 KB.

Focal Point Support Over LU 6.2

Coding RES=N on the CMDMDL statements for focal point support saves 2 KB.

GLOBALV and Save/Restore Focal Point

Coding RES=N on the CMDMDL statements for GLOBALV and save/restore saves 6.5 KB.

HLL and REXX

Coding RES=N on the CMDMDL statements for HLL and REXX saves 7 KB.

HLL Only

Coding RES=N on the CMDMDL statements for HLL saves 6 KB.
LU 6.2 Transport and Operations Management Support

DSI6OSCP CMDMDL MOD=DSI6OSCP,TYPE=D,PARSE=N,RES=N
DSI6LOGM CMDMDL MOD=DSI6LOGM,TYPE=D,RES=N
DSIOSRCP CMDMDL MOD=DSIOSRCP,TYPE=RD,PARSE=N,RES=N
DSIOARCP CMDMDL MOD=DSIOARCP,TYPE=RD,PARSE=N,RES=N
DSIOURCP CMDMDL MOD=DSIOURCP,TYPE=D,PARSE=N,RES=N
DSIOLGFP CMDMDL MOD=DSIOLGFP,TYPE=RD,PARSE=N,RES=N

Coding RES=N on the CMDMDL statements for LU 6.2 saves 63 KB.

Message Forwarding, Uppercase Translation

SET CMDMDL MOD=DSISET,TYPE=R,CTL=N,RES=N
SWITCH CMDMDL MOD=DSISWCP,TYPE=RD,RES=N
TRACE CMDMDL MOD=DSIITP,RES=N
UPPER CMDMDL MOD=CNMCUPPR,ECHO=N,RES=N

Coding RES=N on the CMDMDL statements for message forwarding saves 10 KB.

NetView Bridge

RTRQUEUE CMDMDL MOD=DSINBQUE,TYPE=R,RES=N
TRANRCV CMDMDL MOD=DSINBRCV,TYPE=R,RES=N
TRANSND CMDMDL MOD=DSINBSND,TYPE=R,RES=N
DSINBRSM CMDMDL MOD=DSINBRSM,TYPE=R,RES=N
DSINBTRM CMDMDL MOD=DSINBTRM,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for NetView Bridge saves 23 KB.

NetView Bridge Remote Access

DSINBR62 CMDMDL MOD=DSINBR62,TYPE=R,PARSE=N,RES=N
DSINBRLG CMDMDL MOD=DSINBRLG,TYPE=R,PARSE=Y,RES=N

Coding RES=N on the CMDMDL statements for NetView Bridge remote access saves 2 KB.

NetView Command List Language and REXX

DROPCL CMDMDL MOD=DSIRXDRP,TYPE=R,RES=N
LOADCL CMDMDL MOD=DSIRXLDP,TYPE=R,RES=N
MAPCL CMDMDL MOD=DSIRXMAP,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for NetView command list language and REXX saves 9 KB.

NetView management console

DUIATERM CMDMDL MOD=DUIATERM,TYPE=D,RES=N

Coding RES=N on the DUIATERM statement for the NetView management console saves 3.5 KB.

Network Product Support (NPS)

DISPCMD CMDMDL MOD=DSIYOCDCP,TYPE=R,RES=N
CANCMD CMDMDL MOD=DSIYOCNCP,TYPE=R,RES=N

LPDA-2 Modem Support

MMDCNTL CMDMDL MOD=DSIYOMLP,TYPE=R,RES=N
MMDCNFG CMDMDL MOD=DSIYOMCP,TYPE=R,ECHO=N,RES=N

Service Point
RUNCMD CMDMDL MOD=DSIIYORNP,TYPE=R,RES=N
LINKPD CMDMDL MOD=DSIIYOPD,TYPE=R,RES=N
LINKTEST CMDMDL MOD=DSIIYOTST,TYPE=R,RES=N
LINKDATA CMDMDL MOD=DSIIYODAT,TYPE=R,RES=N

3710, 3708, and 386X Modem Support
LINESTAT CMDMDL MOD=DSIIYOLSP,TYPE=R,RES=N
DISPCNFG CMDMDL MOD=DSIIYODSP,TYPE=R,RES=N
RUNDIAG CMDMDL MOD=DSIIYORDP,TYPE=R,RES=N
LPDA CMDMDL MOD=DSIIYOLPP,TYPE=R,RES=N
THRESH CMDMDL MOD=DSIIYOTHP,TYPE=R,RES=N
CCPLOADI CMDMDL MOD=DSIIYOLIP,TYPE=R,RES=N FROM COMMAND LIST ONLY
CCPLOADT CMDMDL MOD=DSIIYOLTP,TYPE=R,RES=N FROM COMMAND LIST ONLY
CCPLOADF CMDMDL MOD=DSIIYOLF,TYPE=R,RES=N FROM COMMAND LIST ONLY
CCPDOR CMDMDL MOD=DSIIYODR,TYPE=R,RES=N FROM COMMAND LIST ONLY

Coding RES=N on the CMDMDL statements for NPS saves 43 KB.

Other Commands and Command Lists
AFTER CMDMDL MOD=DSITIMER,RES=N
AT CMDMDL MOD=DSITIMER,RES=N
UNIQUE CMDMDL MOD=DSIUNIQ,TYPE=R,RES=N - CANCEL COMMAND LIST COPY
EVERY CMDMDL MOD=DSITIMER,RES=N

Coding RES=N on the CMDMDL statements for commands and command lists saves 4 KB.

Programmable Network Access (PNA) Support
DSIROVS CMDMDL MOD=DSIROVS,TYPE=D,PARSE=N,RES=N
DISPREG CMDMDL MOD=DSIDPREG,RES=N

Coding RES=N on the CMDMDL statements for PNA support saves 14 KB.

Remote Operations Over the LU 6.2 Transport
DSIUSNDM CMDMDL MOD=DSIUSNDM,PARSE=Y,TYPE=RD,RES=N
ENDTASK CMDMDL MOD=DSIUDISM,PARSE=N,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for remote operations saves 8 KB.

Save/Restore
DSITIRTR CMDMDL MOD=DSITIRTR,TYPE=RD,PARSE=N,RES=N
RESTORE CMDMDL MOD=DSIRSTP,TYPE=R,PARSE=Y,RES=N
AAUDRSRT CMDMDL MOD=AAUDRSRT,TYPE=D,PARSE=N,ECHO=N,RES=N

Coding RES=N on the CMDMDL statements for save/restore saves 15 KB.

Sequential Log
DSIBSWCP CMDMDL MOD=DSIBSWCP,TYPE=D,RES=N
DSIZBSQW CMDMDL MOD=DSIZBSQW,TYPE=RD,PARSE=N,RES=N

Coding RES=N on the CMDMDL statements for the sequential log saves 5 KB.

Service and Tuning Command Processors
IDCAMS CMDMDL MOD=DSIDCAMS,RES=N
SUBMIT CMDMDL MOD=DSISUBMT,RES=N
VSAMPOOL CMDMDL MOD=DSIVPOOL,RES=N
SESSMDIS CMDMDL MOD=DSINDISP,RES=N
DSRBS CMDMDL MOD=DSIDSRBS,RES=N
LISTCAT CMDMDL MOD=DSILCAT,RES=N
DISPMOD CMDMDL MOD=DSIDDMOD,RES=N
MSGROUTE CMDMDL MOD=DSIMSGRT,RES=N
RESOURCE CMDMDL MOD=DSIRESRC,RES=N
RESETOB CMDMDL MOD=DSIRSTOB,RES=N
TASKUTIL CMDMDL MOD=DSITASKU,RES=N

Coding RES=N on the CMDMDL statements for service and tuning saves 63 KB.

Session Monitor
AAUSLGEX CMDMDL MOD=AAUSLGEX,TYPE=D,RES=N
AAUSRTEA CMDMDL MOD=AAUSRTEA,TYPE=D,RES=N,PARSE=N

Coding RES=N on the CMDMDL statements for the session monitor saves 3 KB.

Status Monitor
MONIT CMDMDL MOD=CNMCBARR,RES=N REACTIVATE CP
STATMON CMDMDL MOD=CNMCACTL,ECHO=N,RES=N FULLSCREEN DISPATCHER CP
CLRSTATS CMDMDL MOD=CNMCRRST,RES=N ZERO COUNTS
PARSE CMDMDL MOD=CNMCSPAR,RES=N

Coding RES=N on the CMDMDL statements for the status monitor saves 15 KB.

Terminal Access Facility (TAF)
LISTSESS CMDMDL MOD=DSILIST1,RES=N LIST STATUS OF TAF SESSIONS
ENDSESS CMDMDL MOD=DSIEND,RES=N TAF STOP PROCESSOR
SENDSESS CMDMDL MOD=DSIRTPX,RES=N TAF SEND PROCESSOR
DSILMEXP CMDMDL MOD=DSILMEXP,TYPE=R,ECHO=Y,RES=N

Coding RES=N on the CMDMDL statements for TAF saves 8 KB.

4700 Support Facility
SOLICIT CMDMDL MOD=BNJNSOLA,TYPE=R,RES=N
SYSMON CMDMDL MOD=BNJNRPTA,TYPE=R,RES=N

Coding RES=N on the CMDMDL statements for the 4700 Support Facility saves 4 KB.
Index

Special characters
?
wildcard character 67
.DEFAULT statement 6
.NO_ENTRY statement 6
* wildcard character 67
%INCLUDE statement 9
&CGLOBAL statement 30
&EXIT statement 30
&PAUSE statement 17
&TGLOBAL statement 30

Numerics
3710, 3708, and 386X modem support command model statements 164
4700 Support Facility 165
A
AAU024I message 64, 66
AAUPRMLP member
defining SAW buffers 61
defining trace buffer 64
keep class member 67
KEEPPIU initialization parameter 70
LUCOUNT parameter 79
PURGE parameter 75
selectivity of keep classes 67
specifying trace mode 63
tracing gateways 67
tracing LU to LU sessions 63
tracing SSCP to SSCP sessions 63
AAUTSKLP task 61, 64
access method services (AMS)
LISTCAT 101
REPRO 105
accessibility information xi
accounting information 77
ALCACHE statement
determining for system 48
using 48
alert 45, 56
alert cache 48
alert network operations support 160
alert recording (AREC) filter 16, 46
alerts database 16
Alerts-Dynamic panel 47, 48
Alerts-Static panel 47
alerts, forwarding over LUC session 49
ALLOCATE statement 161
allocation, data windows 88
ALWAYS statement 9, 10
AMS (access method services)
LISTCAT 101
REPRO 105
APAR OY06920 6
API, RODM
programming recommendations 95
statistics 89
AREC filter 16, 46
assembler language program 27
ASSIGN command 11
AUTOCNT command
description 11
detail report 12, 13
overview 2
preloaded command lists 28
summary report 14, 15
AUTODROP command 29
automating hardware monitor records 16
automation
automation task (AUTOTASK) 17
CMDMDL statement (common) 160
hardware monitor records 16
limiting system messages 6, 9
MSUs 9, 16
network messages 9
subsystem interface 7
system messages 9
table detail reports 12, 13
table listings 161
table summary reports 14, 15
tuning 5
automation tasks (autotasks)
&PAUSE statement 17
GO command 17
multiple 17
automation workload 17
separating from other NetView workloads 19
autotask (automation task)
&PAUSE statement 17
GO command 17
multiple 17
AUTOTBL command 9
AVAIL parameter 70
availability data 70
B
background pictures, GMFHS 85
basic tuning information 1
BEGIN/END sections, automation table 9, 10
bisynchronous LU names 68
BLDL search 118
BLDVRP macro 99
BLOCK parameter 16, 46
BNJ0301 message 16, 46
BNJ461 message
alert forwarding 49
OPER filter 16, 46
BNJDSERV task 48
BNJMBDST member
ALCACHE statement 48
DSRBO value 54
RATIO (R) statement 56
BNJPNL1 data set 117
BNJPNL2 data set 117
books
feedback ix
online ix
ordering ix
Bridge, NetView 133, 163
Bridge, NetView remote access 163
browse, data set 108
buffer
allocation 99
command 7
callout buffer 97
defining 99
DFR 97
header 65
Hiperspace 100
I/O 97
lost 64
message 7
PIU 64
pools 99
queue limit 116
SAW 60
size 100
trace 63, 64
BUFFERS parameter 100
BUFSIZE parameter 62, 66
BUFSIZE parameter 62, 66
C
cached DASD device 118
calculating CPU utilization 129
calculating task utilization 131
CALL statement 30
CBUF (command buffer) 8
CDINIT 64
CDRM names 68
cell pool statistics 91
CEXEC 31
CGLOBAL 30
channel program 97
checking resource limit 124
CHKPT command 87, 88
CI (control interval) size 100
split 105
CICS applications 68
CISIZE values, NetView 98
CLEAR parameter 89
client programmable workstation 84
clock information 102
CMDMDL statement
adding to DSICMD 113
list 161
CNM router 161
CNM493l message 11
CNMAUTH statement 114
CNMPNL1 data set 117
CNMS0055 member 73, 115
CNMS0058 member 73, 115
commands, performance-related
DSRBS 110
LISTCAT 101
RESOURCES 119
REXX 73
STATCELL 91
TASKUTIL 127
VSAMPOOL 103
common control blocks 97
common global variables 42
compiled command list 31
compiled command procedure 39
compiled EXEC (CEXEC) 31
compiler, REXX 31
compiling REXX procedures 31
configuration data 60
consoles, NetView program 6
constants module, NetView program 115
CONTINUE 11
control area split 105
control block common 97
improve access 113
search algorithm 79
control buffer 97
collection, NetView program 115
control interval (CI) size 100
split 105
conversion, REXX command list 30
counters 65
CP-MSU 16, 116
CPU utilization, calculating 129
cross-domain
CNM data transfer 161
communications 54
logon facilities 162
session initiation (CDINIT) 64
cross-network session initiation 64
cross-network session resource allocation 37, 64
Customer Support 42
customization parameters, RODM 95
data set (continued)
VSAM 88, 97
data set considerations
NetView control interval size 98
partitioned 117
VSAM LSR and DFR
buffer pool 99
how LSR and DFR work 97
VSAM REPRO command 105
data set member browse 108
data window allocation 88
database
size 74
VSAM maintenance 105
DBAUTO command
overview 2
VSAM databases 106
DCN (data circuit-terminating equipment) 117
DCNM router function 162
DD definition list 118
DEFAULT statement 6
DEFAULTS command 37
default write (DFR)
buffer pools 99
DSRBO value in BNJMBDST 101
NetView values 98
delete operator message (DOM) 7, 8
dequeue facility 42
detail reports, automation table 12, 13
DFR
buffer pools 99
DSRBO value in BNJMBDST 101
NetView values 98
DGROUP option 72
dialed lines 118
disability information xi
DISBQL command 116, 133
DISKEEP PIU command 68
dispatching priority 19
DISPPI command
overview 2
sample output 116
usage 116
distributed host 6
DOM (delete operator message) 7, 8
DROPCL command 28
dropping preloaded command lists 29
DSIAUTO macro 16
DSICGLOB 10
DSICGLOB automation table function 42
DSICLD data set 28, 117
DSICMD member 113
DSICMD residency option, examining 23
DSICMOD module
overview 115
sense code filter 73
DSIDNMAT (CNMS7003) module 56
DSILIST data set 117
DSILOG task 98
DSIUCMD member 114
DSIMSG 117
DSINDEF 124
DSINVGRP 39, 114
J
job control language (JCL) 7, 88

K
KCLASS AVAIL option 70
KCLASS statement 67, 68
KEEP classes
coding 67
defining 67
member 68
naming convention 67
processing for direct access storage
device (DASD) 71
SAW data 69
selectivity 67
specifications 68
tailoring KEEPPIU values 66
use 67
keep counts, recommended session
monitor PIU 70
KEEPMEM member 70, 72
KEEPPIU option 70
KEEPPIU parameter 63, 66
KEEPRTM parameter 77
KEEPSESS option 72
KEEPSESS parameter 74
keyboard, shortcut keys xi
keyword, MAXSESS 114

L
language processor environment 37
leased line 19, 118
limiting system messages 6, 10
linear data sets 87, 88
LINK macro 39
link speed 85
LISTA statement 161
LISTCAT command
access method services (AMS) 101
NetView 101
sample output 101
LOAD macro 39
LOADCL command
automation tasks (autotasks) 17
command lists 28, 118
loading new members 67
LOC parameter 99
local function package 32
local shared resource (LSR)
buffer pools 99
deinition 97
DSRBO value in BNJMBDST 101
NetView values 98
log data set 97
LOG parameter 70, 98
log record type 8, RODM 91
logical records 103
logical unit (LU)
number of LUs known by session
monitor 79
SSCP-LU session 64, 77
trace 76
lookaside hit ratio 104
LPDA-2 modem support 163
LSR (local shared resource)
buffer pools 99
definition 97
DSRBO value in BNJMBDST 101
NetView values 98
LU (logical unit)
number of LUs known by session
monitor 79
SSCP-LU session 64, 77
trace 76
LU 6.2 sessions
command and message
forwarding 19
communications 85
forwarding alerts 49
NetView constants module 113
NetView-NetView
communication 116
overview 112
LU-LU session
analyzing network problems 71
configuration data 60
hung terminal 64
keep class statements 69
protocol problem 64
SAW data 60
session partners 60
session status 60, 75
SESSMDIS command 77
TRACELU 63
LUC sessions, forwarding alerts over
LU-COUNT parameter 79
message processing facility (MPF) 5, 10
messages, limiting system 6, 10
MPF (message processing facility) 5, 10
MPFLSTxx member 6
MS transport layer 112
MSUs
automating 16
detail report 12
filtering 16
summary report 14
MSUSEG condition 16, 46
multiple autotasks 17
multiple NetView programs
for separating system and network
automation workloads 19
multiple processors 17
multiple programs 19
multitasking 17
MVS
consoles 6
DSRBO values 98
N
naming convention, keep class 67
NetView
automation 19
Bridge 133, 163
Bridge remote access 163
consoles used 6
constants module 115
CPU utilization 129
data set member browse 108
dispatching priority 19
function package 32
functions for REXX 32
Graphic Monitor Facility 81, 163
Graphic Monitor Facility host
subsystem (GMFHS) 86
library 28, 30
local function 32
primary POI task 42
problem determination (PD) 19
program-to-program interface 116
saving storage 160
SESSMDIS command 67, 75
startup JCL 7, 39
startup procedure 118
subsystem address space job
values 8
subsystem job 7
system function 32
system package 32
user function 32
NetView automation table
%INCLUDE 9
ALWAYS statement 9, 10
AUTOCNT command 11
automated message 6
AUTOTBL command 9
BEGIN/END statement 9
NetView automation table (continued)
design guidelines 9
detail report 12
DSICGLOB 10
efficiency 9
END statement 9, 10
IF-THEN statement 9, 10
message automation 9
message CNM493I 11
MSU automation 9
reducing entries 9
related entries 9
search 6
size 9
summary report 14
suppressed message 9
SYN statement 9
NetView command list language 163
NetView programs
multiple
for system and network automation workloads 19
single
using WLM enclaves 20
NetView workloads
separating automation workload from 19
NetView-NetView task (NNT) compared to RMTCMD command 19, 116
message forwarding 19, 49
network asset management 117
network log browse 108
network management vector transport (NMVT) 116
network message automation 9, 19
network product support 163
NIXL index component 103
NLDM RECORD STRCDATA command 65, 75
NLDM TRACE START command 63
NMVT (network management vector transport) 116
NNT (NetView-NetView task) compared to RMTCMD command 19, 116
message forwarding 19, 49
NO_ENTRY statement 6
nonpersistent session 19, 118
number of REXX environments 38
NUMVARS option of DSIVARS macro 42
NVAUTO (generic automation receiver) 39, 114
O
OMIT specification 124
online publications x
OPER filter 16, 46
operations management support
command model statements 163
operator (OPER) filter 16, 46
operator request 65
operator station task (OST) 17, 19
ordering publications x
OST (operator station task) 17, 19
OVERRIDE command 37
P
panels 118
PARALLEL logmode 113
PARSE EXTERNAL statement 17
parse function 17
PARSE PULL statement 17
PARSEL2R 31
partitioned data set (PDS) 117
partitioned data set directory 118
PASS parameter 16
path length, RODM 95
pattern-matching character 67
PAUSE statement 17
PDS (partitioned data set) 117
performance considerations 23
concatenating the DSICLD library 23
DDF tree and panel, customizing 24
NetView 23
node automation, switching 24
operations 24
reordering the automation table 23
performance, enhancing 42, 109
persistent session
LUC sessions 118
status focal point 19
physical unit (PU) 117
PU
buffers
allocation 64
estimate fullness 65
ratio 78
size 65, 78
storage 64
count 65
data space limit factor (RACPIULM) 66
keep counts 70
kept for traced session 66
trace 77
PL/1 programs 40
PLU name 68
PNA (programmable network access) 164
PPT (primary program operator interface task) 42
preinitialized environments, PL/1 40
preinstallation tasks
updating the IRXANCHR table 36
preloading command lists 28
primary program operator interface task (PPT) 42
procedures, REXX 31
processing, global variables 30
program-to-program interface 116
programmable network access (PNA) 164
programmable operator facility (PROP) 10
programmable operator message exchange facility (PMX) 5
programmable workstation client 84
connectivity to status focal point 85
storage estimates 83
programs, multiple 19
protocol problem 64
PU 64, 117
PU names 68
publications
feedback ix
online ix
ordering ix
PURGE command 74
PURGE parameter 75
PURGEDB command 74, 105
Q
QRYGLOBL command
enhancing performance 42
overview 2
R
RACMXBUF (SAW buffer limit) 62
RACPIULM (PIU data space limit factor) 62, 66
RACSAWLM (SAW data space limit factor) 66
RACSAWPK (SAW data space packing factor) 61
RECORD command 79
record filtering 16
region size 8, 155
RELOAD command 67
remote operations over LU 6.2
transport 164
reorganizing database
recovering space lost 105
session monitor 74
REPRO command 105
request units (RUs) 110
RESETDB command 74, 105
RESIDENT entries 31
RESOURCE command 119
Resource Object Data Manager (RODM)
API statistics 89
checkpoint guidelines
EKGD001 data set 88
EKGD002 data set 88
EKGMAST data set 88
EKGTRAN data set 88
customization parameters 95
log record type 8 91
path length 95
programming recommendations 95
storage manager 93
storage requirements 150
resource pool 99
resource status collector
resynchronization 85
response time monitor (RTM) 77
RESTC 42
RESTT 42
REXX
CALL statement 30
command list conversion 30
compiler 31
considerations 30
environments 37
### SAW (session awareness) data

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTM data</td>
<td>69</td>
</tr>
<tr>
<td>session partners</td>
<td>60</td>
</tr>
<tr>
<td>session PIU</td>
<td>59</td>
</tr>
<tr>
<td>session status</td>
<td>60, 75</td>
</tr>
<tr>
<td>SESSM DIS command</td>
<td>61, 75</td>
</tr>
<tr>
<td>size of SAW buffers</td>
<td>60</td>
</tr>
<tr>
<td>SSCP-LU session</td>
<td>60, 77</td>
</tr>
<tr>
<td>SSCP-PU session</td>
<td>60, 76</td>
</tr>
<tr>
<td>SSCP-SSCP session</td>
<td>60, 76</td>
</tr>
<tr>
<td>storage required per session</td>
<td>60</td>
</tr>
<tr>
<td>VTAM SAW filter</td>
<td>60, 67</td>
</tr>
<tr>
<td>SAW parameter</td>
<td>69</td>
</tr>
<tr>
<td>SDLC (synchronous data link control)</td>
<td>85</td>
</tr>
<tr>
<td>SDO MAIN command</td>
<td>114</td>
</tr>
<tr>
<td>search algorithm</td>
<td>79</td>
</tr>
<tr>
<td>search sequence</td>
<td>32</td>
</tr>
<tr>
<td>search table</td>
<td>79</td>
</tr>
<tr>
<td>security</td>
<td>109</td>
</tr>
<tr>
<td>security scenarios</td>
<td>95</td>
</tr>
<tr>
<td>selective tracing</td>
<td>60, 63</td>
</tr>
<tr>
<td>sense code filtering</td>
<td>73</td>
</tr>
<tr>
<td>separating system and network automation workloads</td>
<td></td>
</tr>
</tbody>
</table>
  - using multiple NetView programs | 19 |
  - separating the automation workload from other NetView workloads | 19 |
| sequential log command model statements | 164 |
| server-client configurations | 85 |
| service point command model statements | 163 |
| service request block (SRB) | 119 |
| session accounting | 64, 77 |
| counts | 76 |
| cross-domain | 60 |
| cross-network | 60 |
| ends | 74, 78 |
| initiation | 64 |
| NCP trace data | 59 |
| partner | 60 |
| partner name | 67 |
| PIUs | 59, 66 |
| recording | 72 |
| resource allocation | 64 |
| same domain | 60 |
| starts | 67, 78 |
| status | 60, 75 |
| trace data | 59 |
| session awareness (SAW) data allocating buffers | 61 |
| buffer ratio | 61 |
| configuration data | 60 |
| description | 60 |
| discarded | 69 |
| filtering | 60 |
| filtering by NetView program | 60 |
| filtering by VTAM | 60 |
| keep classes | 69 |
| keeping selectively | 60 |
| LU-LU session | 60, 77 |
| LUSSSCP session | 60, 76 |
| non-RTM data | 69 |
| notifications | 61 |
| RTM data | 69 |
| session partners | 60 |

### Session and Storage Information

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>panel</td>
<td>76</td>
</tr>
<tr>
<td>specific trace mode</td>
<td>63</td>
</tr>
<tr>
<td>storage usage</td>
<td>77</td>
</tr>
<tr>
<td>trace data</td>
<td>65</td>
</tr>
<tr>
<td>gateway</td>
<td>67</td>
</tr>
<tr>
<td>global</td>
<td>63</td>
</tr>
<tr>
<td>NCP</td>
<td>59</td>
</tr>
<tr>
<td>PIU</td>
<td>59</td>
</tr>
<tr>
<td>recording</td>
<td>63</td>
</tr>
<tr>
<td>selective tracing</td>
<td>63</td>
</tr>
<tr>
<td>specific</td>
<td>63</td>
</tr>
<tr>
<td>summarized</td>
<td>63</td>
</tr>
<tr>
<td>viewing</td>
<td>63</td>
</tr>
<tr>
<td>trace mode</td>
<td>63</td>
</tr>
<tr>
<td>traffic data</td>
<td>65</td>
</tr>
<tr>
<td>tuning</td>
<td>59</td>
</tr>
<tr>
<td>warm start</td>
<td>60</td>
</tr>
<tr>
<td>work load traffic counters</td>
<td>78</td>
</tr>
<tr>
<td>session status</td>
<td>60</td>
</tr>
<tr>
<td>session-end notification</td>
<td>65</td>
</tr>
<tr>
<td>sessions, persistent and nonpersistent</td>
<td>118</td>
</tr>
</tbody>
</table>

### Session Awareness (SAW) data

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>session PIU</td>
<td>66</td>
</tr>
<tr>
<td>session status</td>
<td>60</td>
</tr>
<tr>
<td>SESSM DIS command</td>
<td>61, 75</td>
</tr>
<tr>
<td>size of SAW buffers</td>
<td>65</td>
</tr>
<tr>
<td>SSCP-LU session</td>
<td>60, 77</td>
</tr>
<tr>
<td>SSCP-PU session</td>
<td>60, 76</td>
</tr>
<tr>
<td>SSCP-SSCP session</td>
<td>60, 76</td>
</tr>
<tr>
<td>storage required per session</td>
<td>60</td>
</tr>
<tr>
<td>VTAM SAW filter</td>
<td>60, 67</td>
</tr>
<tr>
<td>session block</td>
<td>79</td>
</tr>
<tr>
<td>session initiation (INIT)</td>
<td>64</td>
</tr>
<tr>
<td>session monitor accounting</td>
<td>59, 77</td>
</tr>
<tr>
<td>COLLECT and RECORD commands</td>
<td>79</td>
</tr>
<tr>
<td>command model statements</td>
<td>165</td>
</tr>
<tr>
<td>data recording (SMDR) command</td>
<td>75</td>
</tr>
<tr>
<td>database managing</td>
<td>73</td>
</tr>
<tr>
<td>DISKEEP PIU command</td>
<td>68</td>
</tr>
<tr>
<td>failure</td>
<td>63</td>
</tr>
<tr>
<td>global trace mode</td>
<td>63</td>
</tr>
<tr>
<td>initialization</td>
<td>67</td>
</tr>
<tr>
<td>keep class</td>
<td></td>
</tr>
</tbody>
</table>
  - DASD keep-class processing | 71 |
  - recommend PIU keep counts | 70 |
  - SAW data | 67 |
| KEEPP PIU | 70 |
| LUCOUNT parameter | 79 |
| NCP trace data | 59 |
| PIU | 59 |
| processing | 60 |
| RELOAD command | 67 |
| reorganizing database | 74 |
| RTM (response time monitor) | 59 |
| SAW (session awareness) data | 
  - selective tracing | 60, 63 |
  - session and storage information panel | 76 |
| specific trace mode | 63 |
| storage usage | 77 |

### Samples Library

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>samples library</td>
<td>99</td>
</tr>
</tbody>
</table>
SESSMDIS (continued)
tuning PIU buffer allocation 65
SESSSTATS parameter accounting and availability 64, 70 session and storage information panel 66, 77 set domain (SDOMAIN) command 114 set recording filter (SRF) command 56 SETBQL command 116, 133 SETCV 64 shortcut keys, keyboard xi SHRPOOL parameter 99 single NetView programs using WLM enclaves 20 SLU names 68 SMDR (session monitor data recording) command 75 SNA LU names 68 SNA topology manager 123 SNI session 79 solicited DSRBUs 110 specific trace 63 SRATIO command 55 SRB 119 SRFILTER (SRF) command disabling network resources 55 setting hardware monitor filters 16, 45 SSCP session 63 SSCP trace 76 SSCP-LU session configuration data 60 keep class statements 69 SAW data 60 session initiation (INIT) 64 session partners 60 session status 60 SESSMDIS command 77 TRACESC 63 VTAM application 64 SSCP-PU session configuration data 60 cross-network session resource allocation 64 gateway NCPs 63 SAW data 60 session partners 60 session status 60 SESSMDIS command 76 TRACESC 63 SSCP-SCCP session configuration data 60 cross-domain session initiation 64 SAW data 60 session failure 63 session partners 60 session status 60 SESSMDIS command 76 TRACESC 63 startup JCL 39 STATAPI parameter generating RODM statistics 89 overview 3 STATCELL parameter 91 statistics 45, 91 STATOPT filtering 124 status focal point workstation connectivity 85 status monitor command model statements 165 resource limit 124 STEPIL DD statement 39, 124 STOP FORCE command 98 storage considerations 39, 160 manager, RODM 93 programmable workstation estimates 83 RODM requirements 150 storage usage tracking 159 STRNO parameter 99 subarea dial overhead 19 subpool 64 subroutines 30 subsystem address space job values 8 subsystem allocatable consoles 6 subsystem interface 5 subsystem job 7 summary reports, automation table 14, 15 SVFILTER command 46 SWITCH command 98 switched line 19, 118 SWRAP command 54 SYN statement 9 synchronous data link control (SDLC) 85 system command 7 CPU utilization 130 function call 32 function package 32 library 30 message 6, 9 message traffic 6 system function call 32 T task control block (TCB) 119 task global variables 42 task utilization, calculating 131 TASKMON 159 TASKUTIL 159 command output 129, 131 command processor 127, 164 diagnosis techniques 132 suggestions for using 132 tuning techniques 132 TCB 119 temporary VIO data set 118 terminal access facility (TAF) 165 TGLOBAL 30 time-out facility 17 Tivoli Customer Support xi token-ring connections 85 TRACE command 63 TRACE command specification 63, 64 TRACE START command 63 trace, gateway 67 TRACEGW 67 TRACELU 63 TRACEPPI command 117 traces, global and specific 63 TRACESC 63 TSO applications 68 TSO, reserving REXX environments 36 TSO/E ARXANCHR environment 39 IRXANCHR environment 39 TSO/E local 32 TSO/E system 32 TSO/E user 32 TSO/E IRXANCHR environment 39 tuning additional considerations 107 automated operations 5 basic information 1 Bridge, NetView 133 command procedures 27 definition 1 hardware monitor 45 LU 6.2 transport 112 NetView Graphic Monitor Facility host 86 workstation 81 RODM 87 session monitor 59 status monitor 107 VSAM 97 tuning considerations 23 concatenating the DSICLD library 23 DDF tree and panel, customizing 24 NetView 23 node automation, switching 24 operations 24 reordering the automation table 23 U unsolicited DSRBUs 110 uppercase translation command model statement 163 usage reports, automation table detail report 12 summary report 14 user function package 32 V variable dictionary 42 variables, common global 42 VIO data set 118 virtual I/O (VIO) data set 118 virtual storage access method (VSAM) BLDVRP macro 99 buffer allocation 99 Cl size 97 data set 97 linear data sets (LDSs) 87, 88 local shared resources (LSR) and deferred write (DFR) 97 record queue 77 resource pool 99 session recording 78 tuning 97 VSAMPOOL command 100
virtual storage, tracking 159
virtual storage, unused 79
virtual telecommunications access
method (VTAM)
  application 64
  constants module (ISTRACON) 66
  D NET command 63
display command 63
  MVS/ESA considerations 61
  naming conventions 68
  PIU buffer 64
  private storage 61
  SAW filter 60
  sessions 59
vital product data (VPD) 117
VPD (vital product data) 117
VSAM
  BLDVRP macro 99
  buffer allocation 99
  CI size 97
data set 97
  database maintenance 105
  I/O 97
  linear data sets (LDSs) 87, 88
local shared resources (LSR) and
defered write (DFR) 97
  record queue 77
  REPRO command 105
  resource pool 99
  session recording 78
  tuning 97
  VSAMPOOL command 100
VSAM ACB options 102
VSAM REPRO command 105
VSAMPOOL command
  buffer pool 100
  overview 3
  sample output 104
  usage 103
VSE/Operator Communication Control
Facility (VSE/OCCF) 10
VTAM
  application 64
  constants module (ISTRACON) 66
  D NET command 63
display command 63
  MVS/ESA considerations 61
  naming conventions 68
  PIU buffer 64
  private storage 61
  SAW filter 60, 67
  sessions 59
  VTAMLST 124

workload, automation
  separating from other NetView
  workloads 19
wrap counts 54, 72
wrap counts, initialization 54
write-to-operator (WTO) command 7, 8
write-to-operator with reply (WTOR)
  command 7, 8
WRTBFR macro 97
WTO (write-to-operator) command 7, 8
WTOR (write-to-operator with reply)
  command 7, 8

X
XCTL macro 39, 124

W
WAIT FOR OPINPUT 17
warm start
  RODM 87, 89
  session monitor 60
wildcard characters 67
window allocation 88
WLM (Work Load Manager) 20
Work Load Manager enclaves
  for separating system and network
  automation workloads 20