

Tivoli® NetView® for z/OS™



Command Reference Volume 2

Version 5 Release 1

Tivoli® NetView® for z/OS™



Command Reference Volume 2

Version 5 Release 1

Tivoli NetView for z/OS Command Reference

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, AIX, APPN, BookManager, DB2, DFSMS/MVS, MVS, MVS/DFP, MVS/SP, NetView, OS/2, OS/390, RACF, SecureWay, Tivoli Enterprise, TME, TME 10, VTAM, and z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

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

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

Part 1. NetView, Automated Operations Network, and MultiSystem Manager Commands 1

Chapter 1. Using Commands	3
Tasks	5
Command Types	5
Command Priority	6
Entering Commands	6
Restricting Access to Resources and Commands	7
Syntax Conventions Used in This Document	7

Chapter 2. NetView Commands and Command Descriptions 9

Part 2. Automated Operations Network Commands 11

Chapter 3. AON Base Commands	13
ACTMON	14
AON	15
AONAIP	16
AONENABL	18
AONGW	19
AONHD	20
AONINFO	21
AONINIT	22
AONMAINT	23
AONOIV	24
AONTAF	25
AONTASK	26
AONTRACE	27

AUTOVIEW	29
CDLOG	30
CGED	31
DBMAINT.	32
DDF.	34
DELAUTO	35
DELMONIT	36
DELNTFY	37
DELTHRES	38
DETAIL	39
DISAUTO	40
DISNTFY	41
DISTHRES.	42
DSPCFG	43
DSPSTS.	44
ILOG	46
INFORM (AON).	48
INFORMTB (AON).	49
LOADTBL.	51
MARK	52
NLOG	53
SENDCMD	55
SETAUTO	57
SETMONIT	59
SETNTFY	61
SETTHRES	63
STARTEZL	65
STARTGWY	67
STOPEZL	68
UNMARK.	70

Chapter 4. AON/SNA Commands 71

AONSNA	72
AONX25	73
APPN	74
CHGSNBU	75
CHGSPEED	76
DISPOOL	77
DISSNBU	79
DSPSNBU	80
LISTSNBU.	83
LUDRPOOL	84
NETSTAT	85
QRYSNBU.	87
SETPOOL	88
SETSNBU	89
SNAHD	91
SNAMAP	92
SNAPULST	93
SNBU	94
VTAMCMD	95
VTAMOPT	96
X25MONIT	97

Chapter 5. AON/TCP Commands. 99

AONTCP.	100
FKXESSF	101
IPCMD	103
IPMAN	105
IPMANSSF	106

IPTRACE	108
IPSTAT	109
MVSPING	110
NV6KCMD	112
NV6KLIST	113
NV6KPERF	114
NV6KPING	115
NV6KRPNG	116
NV6KVIEW	117
NVSNMP	118
SNMPVIEW	120
TRACERTE	122

Part 3. MultiSystem Manager Commands 125

Chapter 6. MultiSystem Manager Base Commands 127

DISPTOPO	128
DMCS (Distribution Manager Command Support)	130
FLCACTIP	133
INITTOPO	134
REMOVBJ	135
RESTOPO	139
SETREMV	140
SUSPTOPO	142

Chapter 7. MultiSystem Manager IP Commands 143

CRITRES	144
GETTOPO IPDETAIL	145
GETTOPO IPONLY and IPRES	150

Chapter 8. MultiSystem Manager LNM Commands 157

GETTOPO LNMADP, LNMBRG and LNMCAU	158
GETTOPO LNMRES and LNMONLY	163
GETTOPO LNMSEG	170

Chapter 9. MultiSystem Manager NetFinity Commands 175

DISPMON	176
DISPPRC	177
DISPPRO	178
DISPTHR	179
GETTOPO NFGRP	180
GETTOPO NFOONLY	186
GETTOPO NFRES	192
GETTOPO NFWKST	199

Chapter 10. MultiSystem Manager Open Commands 205

GETTOPO HOSTONLY	206
GETTOPO OPENRES	211

Chapter 11. MultiSystem Manager TMR Commands 217

GETTOPO TMEONLY	218
GETTOPO TMERES	223

Part 4. Command List NetView Commands 229

Chapter 12. Command List Commands 231

DSITSTAT (REXX)	232
DSIVSAM (PIPE)	236

DSIVSAM DEL (PIPE)	239
DSIVSAM GET (PIPE)	241
DSIVSAM GETREV (PIPE)	243
DSIVSAM INQUIRE (PIPE)	244
DSIVSAM PUT (PIPE)	247
DSIVSMX (PIPE)	248
DSIVSMX CLOSE (PIPE)	250
DSIVSMX DEL (PIPE)	251
DSIVSMX GET (PIPE)	253
DSIVSMX GETREV (PIPE)	255
DSIVSMX IDCAMS (PIPE)	256
DSIVSMX INQUIRE (PIPE)	257
DSIVSMX OPEN (PIPE)	260
DSIVSMX PUT (PIPE)	261
DUIFECMV (RODM)	262
FLUSHQ (REXX)	264
GETM Commands.	265
GETMLINE (REXX)	267
GETMPRES (REXX)	269
GETMSIZE (REXX)	272
GETMTFLG (REXX)	274
GETMTYPE (REXX)	276
GLOBALV (REXX)	281
GLOBALV DEF (REXX)	289
GLOBALV GET (REXX)	291
GLOBALV PURGE (REXX)	293
GLOBALV PUT (REXX)	295
GLOBALV RESTORE (REXX)	297
GLOBALV SAVE (REXX)	299
MSGREAD (REXX)	301
MSGROUTE (REXX)	304
PARSEL2R (REXX)	308
TRAP (REXX)	314
WAIT (REXX)	321
Sending Messages to the Operator Console	331
DOM (REXX)	333
WTO (REXX)	339
WTOR (REXX)	349

Preface

This document describes the commands and components of Tivoli® NetView® for z/OS™ Version 5 Release 1 (NetView) that can be used for network operation. You can use many of these commands in command lists or command procedures.

Who Should Read This Document

This document is intended for system console operators, network operators, and system programmers. Before using this document, be familiar with the basic functions presented in the *Tivoli NetView for z/OS User's Guide*. Specific operator procedures are defined by the individual installation to suit local requirements.

What This Document Contains

This document is divided into the following sections:

- “Chapter 1. Using Commands” on page 3 describes the various components of NetView and explains the syntax conventions used in this document.
- “Part 2. Automated Operations Network Commands” on page 11 alphabetically lists all of the Automated Operations Network commands and their functions.
- “Part 3. MultiSystem Manager Commands” on page 125 alphabetically lists all of the MultiSystem Manager commands and their functions.
- “Part 4. Command List NetView Commands” on page 229 alphabetically lists all of the REXX and NetView command list language commands and their functions.

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 1. Resource Web sites

IBM®	http://www.ibm.com/
Tivoli Systems	http://www.tivoli.com/
Tivoli NetView for z/OS	http://www.tivoli.com/nv390

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

Preface

- 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:
<http://www.tivoli.com/support/survey/>

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 newsgroups, 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 .
<i>Italics</i>	Variables and new terms appear like <i>this</i> , in <i>italics</i> . Words and phrases that are emphasized also appear like <i>this</i> , in <i>italics</i> .
Monospace	Code examples, output, and system messages appear like <code>this</code> , 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

Preface

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:

<http://www-1.ibm.com/servers/eserver/zseries/zos/security/racfss.html>

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 for OS/390 are included in the IBM Communications Server for OS/390 element of the OS/390 operating system. Refer to <http://www.software.ibm.com/enetwork/commserver/about/csos390.html>.

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

Element Position	Meaning
On the command line	Required
Above the command line	Default
Below the command line	Optional

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.

CCPLOADF

▶▶—CCPLOADF *resname*—————▶▶

Figure 1. Required Syntax Elements

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.

TRANSMMSG

▶▶—TRANSMMSG MEMBER=*membername*—————▶▶

Figure 2. Syntax for Variables

Optional Keywords and Variables

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

DISPREG

▶▶—DISPREG ———▶▶
 └ ID=*resname* ─┘

Figure 3. Optional Syntax Elements

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 on page xii 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

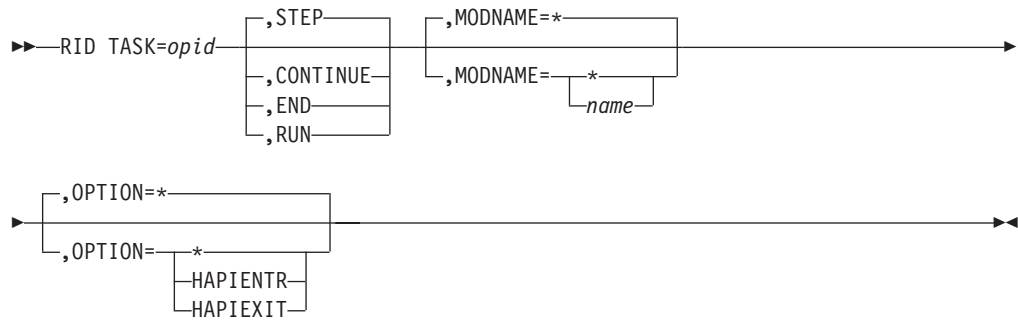


Figure 4. Sample of Defaults Syntax

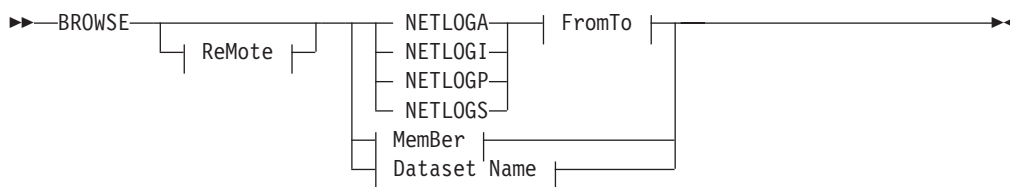
Long Syntax Diagrams

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.

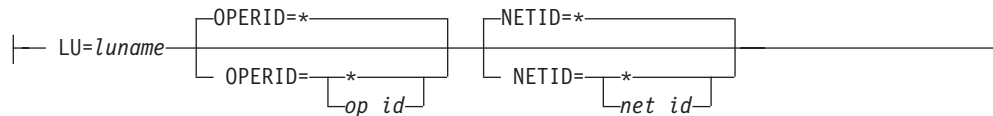
Syntax Fragments

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 xiii for a syntax with the fragments ReMote and FromTo.

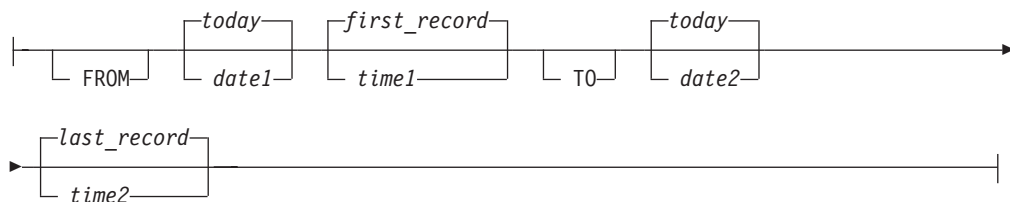
BROWSE



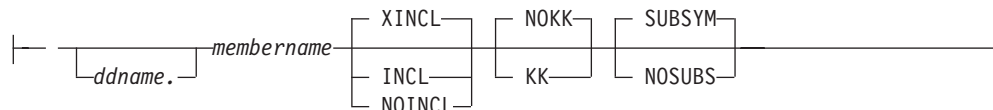
ReMote:



FromTo:



MemBer:



Dataset Name:



Figure 5. Sample Syntax Diagram with Fragments

Commas and Parentheses

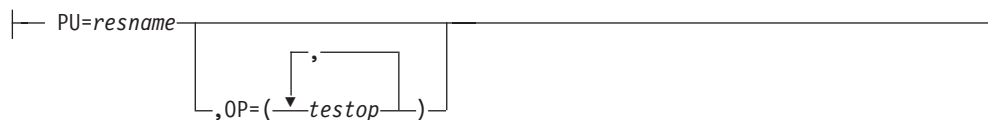
Required commas and parentheses are included in the syntax diagram. When an operand has more than one value, the values are typically enclosed in parentheses and separated by commas. In Figure 6 on page xiv, the OP operand, for example, contains commas to indicate that you can specify multiple values for the *testop* variable.

Preface

CSCF



Pu



PurgeAll



PurgeBefore

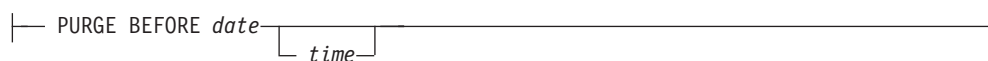


Figure 6. Sample Syntax Diagram with Commas

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 xii.

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

This element...	Looks like this...
Keyword	CCPLOADF
Variable	<i>resname</i>
Operand	MEMBER= <i>membername</i>
Default	<u>today</u> or INCL

Abbreviations

Command and keyword abbreviations are described in synonym tables after each command description.

Part 1. NetView, Automated Operations Network, and MultiSystem Manager Commands

Chapter 1. Using Commands	3
Tasks	5
Command Types	5
Command Priority	6
Entering Commands.	6
Restricting Access to Resources and Commands	7
Syntax Conventions Used in This Document.	7
Chapter 2. NetView Commands and Command Descriptions	9

Chapter 1. Using Commands

When using NetView commands, it is helpful to know that the NetView program is composed of the following components:

Automated Operations Network

Provides a set of programs that can be customized and extended to provide network automation.

Browse facility

Provides the capability to browse the network log or a data set member.

Command facility

Provides base service functions and automated operations. Included are the operator interface and network and trace logging facilities. The command facility can also operate as a subsystem of MVS.

The component identifier for the command facility is NCCF.

Event/Automation Service

Provides translation and forwarding services for alerts, messages, TEC events and SNMP traps. The component identifier for the Event/Automation Service is EAS.

Graphic Monitor Facility host subsystem

Provides a link between RODM and the NetView Management Console to display Systems Network Architecture (SNA) and non-SNA resources.

The component identifier for the Graphic Monitor Facility host subsystem is GMFHS.

Hardware monitor

Provides information about physical network resources. This includes failure information that shows probable cause and recommended actions and information on the 4700 Support Facility. The information can be grouped into the following categories:

- Events
- Statistics
- Alerts

Events are unusual conditions detected by a device about itself or on behalf of a device it controls. Events can be records of permanent errors and other warning and exception conditions. Statistics include information describing the number of transmissions and retransmissions for traffic on a line. An alert is an event that is considered critical and requires operator attention. Whether an event is important enough to be considered an alert is determined by a filter. This filtering decision is made using criteria set in your installation based on how you want to manage and control your network and what information the operators need to see.

The component identifier for the hardware monitor is NPDA.

Help facility

Displays online help information for the NetView components, panels, messages, and commands. This facility includes a procedure that help desk personnel can use to help with problem determination. An index is provided for quick reference.

MultiSystem Manager

Simplifies the task of managing your network resources.

NetView Management Console

Monitors and graphically displays the resources that represent a network, a portion of a network, or a group of networks at various levels of detail.

The component identifier for the NetView Management Console is NMC.

Resource Object Data Manager

An object-oriented, high-speed, multithread in-memory data cache that provides application programs with a means of rapidly accessing or changing the status of data.

The component identifier for the Resource Object Data Manager is RODM.

Session monitor

Provides information about the logical network resources. This includes session-related information such as response time measurement and the components that make up a session. Table 4 shows how this information is grouped:

Table 4. Session Monitor Data Categories

Data Type	Data Description
Session awareness data	Information about session activity within the networks. This data identifies the partners of each session, which can be in the same domain, in different domains, or in different networks.
Session trace data	Consists of session activation parameters, VTAM path information unit (PIU) data, and network control program (NCP) data.
Session response time data	Measured response time broken down into ranges of time that are specified by the performance class definitions.
Route data	Includes a list of PUs and transmission groups (TGs) that make up the explicit route used by a session.
Network accounting and availability measurement data	Network availability data and distribution of use of network resources.

The component identifier for the session monitor is NLDM.

Status monitor

Displays the status of network resources in a hierarchical manner and enables you to browse the network log. This facility also automatically reactivates minor nodes except applications and cross-domain resources (CDRSC).

The component identifier for the status monitor is STATMON.

SNA Topology Manager

Performs dynamic collection and display of APPN[®], subarea, and LU topology and status. This topology and status is stored in RODM for use by NMC.

The component identifier for the SNA Topology Manager is TOPOSNA.

4700 Support Facility

Provides support for the IBM 3600 and 4700 Finance Communication Systems.

The component identifier for the 4700 support facility is TARA.

The NetView program operates as a VTAM application or as a subsystem of MVS. It provides a network log to record information as necessary.

Additional Information

Topic	Refer to
Command Summary	<i>Tivoli NetView for z/OS User's Guide</i>

Tasks

The NetView program can perform many functions. The NetView program controls these functions by defining units of work called *tasks*. The types of tasks are:

- OST** Operator station task. There is one OST for each NetView operator. There are also automated OSTs (autotasks), which perform unattended operations functions related to automation. You can use OSTs to maintain the online sessions with the command facility terminal operators. The OST also analyzes commands as it receives them from the operator and invokes the appropriate command processors.
- NNT** NetView-NetView task. When you have more than one NetView domain, each OST can have secondary sessions across domains with other NetViews. The OST in the remote domain is called a NetView-NetView task (NNT). There is one NNT for each cross-domain NetView with which the local NetView domain communicates. This task controls communication with cross domain NetView programs to issue commands and receive responses. An NNT is not used by the RMTCMD command, which uses LU 6.2 to communicate and uses distributed automated OSTs. RMTCMD is recommended for issuing cross domain commands, although NNTs are still supported.
- PPT** Primary program operator interface task. This task processes commands and command lists that are performed on a system-level basis. NetView uses the PPT to carry out all timer management functions. It runs timer-initiated commands designed to run under the PPT, and it can expand and run command lists. The PPT also routes unsolicited VTAM messages to the authorized receiver.
- DST** Data services task. This task processes requests for communication network management (CNM) or virtual storage access method (VSAM) data. This is the interface to network management data and VTAM.
- MNT** Main task. The NetView main task loads and attaches other NetView tasks.
- HCT** Hardcopy task. The hardcopy task logs messages received from or sent to a specified operator station. The HCT uses a specified printer to accomplish this.
- OPT** Optional task. Optional tasks are user-defined subtasks that can provide increased flexibility beyond the subtasks that the NetView program provides.

Command Types

The NetView program processes different types of commands, including:

- Regular commands
- Immediate commands
- Data services task commands

Most commands and all command lists are **regular commands**. Regular commands can run concurrently with other regular commands. Regular commands can be interrupted by system routines or by immediate commands.

Immediate commands, such as RESET, GO, and AUTOWRAP, can interrupt or preempt regular commands. As their name implies, they run as soon as you enter the command. Only one immediate command runs at a time. You can run immediate commands only in the command facility.

Data services commands run under a data services task (DST). These commands are internal to the NetView program. You cannot enter them at your terminal.

Some commands can run as either regular or immediate commands. If you enter a command at your terminal, it is treated as an immediate command. If the command is in a command list, it is treated as a regular command.

Some commands can run as either immediate or data services commands. If you enter a command at your terminal, it is treated as an immediate command. If the command is run under a DST, it is treated as a data services command.

Command Priority

A NetView OST, PPT, NNT, or autotask's command priority determines the order in which NetView commands and command lists are processed. You can set or modify command priority with the NetView DEFAULTS or OVERRIDE commands, or override it for one command with the CMD command.

Whether a command is entered by the operator or sent to the task, the command is queued on the message queue of that task.

Any command entered at a terminal or sent using an EXCMD command from another task is queued to the target task's message queue corresponding to the target task's command priority.

Additional Information

Topic	Refer to
Command priority	<i>Tivoli NetView for z/OS User's Guide</i>

Entering Commands

You can enter commands from either the NetView console or from the MVS console. From the MVS console, you can use either the MVS MODIFY command or prefix the command with the NetView designator character.

Additional Information

Topic	Refer to
Entering commands from the MVS console	<i>Tivoli NetView for z/OS User's Guide</i>

Restricting Access to Resources and Commands

Both the commands that you can issue and the resources you can access are set for your operator ID. One such restriction is called *span of control*. Span of control restricts your control to select network resources. The span of control for which you are authorized is defined in either your operator profile or in an SAF product, depending on the method used for security authorization.

Another restriction is called *command authorization*. Command authorization restricts the use of commands, keywords, and values. This command authorization is defined in either the DSIPARM data set or in an SAF product.

Additional Information

Topic	Refer to
Restricting command access	<i>Tivoli NetView for z/OS Security Reference</i>
List of commands, keywords, and values which can be restricted	<i>Tivoli NetView for z/OS Security Reference</i>

Syntax Conventions Used in This Document

The commands are listed in alphabetical order for easy reference. The NetView component, the equivalent NetView command list (if any), and any operating system restrictions are provided for each command. If more than one component name is listed next to the command name, you can use the command in more than one NetView component.

The formats and operands of the NetView commands and command lists are described in the same notation throughout the document. Each command description includes the format and description of operands and, where applicable, restrictions, examples, and responses. The command syntax and examples shown assume they are being entered from within the appropriate component.

Chapter 2. NetView Commands and Command Descriptions

This chapter describes the formats of NetView commands and command lists. You can enter these commands from the command facility from any other NetView component.

The commands are listed in alphabetical order. Each command description includes the format and description of operands and, where applicable, usage notes, responses, and examples.

To get online help for a specific NetView component, enter:

HELP *component*

Where *component* is the name of the NetView component. The possible values for *component* are:

AON	Automated Operations Network components
BROWSE	Browse facility
EAS	Event/automation service
GMFHS	Graphic Monitor Facility host subsystem
HELP	Online help
MultiSystem Manager	MultiSystem Manager components
NCCF	Command facility
NLDM	Session monitor
NMC	NetView Management Console
NPDA	Hardware monitor
RODM	Resource Object Data Manager
STATMON	Status monitor
TARA	4700 Support Facility
TOPOSNA	SNA topology manager

For online help on a specific command, enter:

HELP *command*

Where *command* is the name of the command.

For online help on messages, enter:

HELP *msgid*

Where *msgid* is the identifier of the NetView message for which a help panel is to be displayed.

Part 2. Automated Operations Network Commands

Chapter 3. AON Base Commands	13	NETSTAT	85
ACTMON	14	QRYSNBU	87
AON	15	SETPOOL	88
AONAIP	16	SETSNBU	89
AONENABL	18	SNAHD	91
AONGW	19	SNAMAP	92
AONHD	20	SNAPULST	93
AONINFO	21	SNBU	94
AONINIT	22	VTAMCMD	95
AONMAINT	23	VTAMOPT	96
AONOIV	24	X25MONIT	97
AONTAF	25		
AONTASK	26	Chapter 5. AON/TCP Commands	99
AONTRACE	27	AONTCP	100
AUTOVIEW	29	FKXESSF	101
CDLOG	30	IPCMD	103
CGED	31	IPMAN	105
DBMAINT	32	IPMANSSF	106
DDF	34	IPTRACE	108
DELAUTO	35	IPSTAT	109
DELMONIT	36	MVSPING	110
DELNTFY	37	NV6KCMD	112
DELTHRES	38	NV6KLIST	113
DETAIL	39	NV6KPERF	114
DISAUTO	40	NV6KPING	115
DISNTFY	41	NV6KRPNG	116
DISTHRES	42	NV6KVIEW	117
DSPCFG	43	NVSNMP	118
DSPSTS	44	SNMPVIEW	120
ILOG	46	TRACERTE	122
INFORM (AON)	48		
INFORMTB (AON)	49		
LOADTBL	51		
MARK	52		
NLOG	53		
SENDCMD	55		
SETAUTO	57		
SETMONIT	59		
SETNTFY	61		
SETTHRES	63		
STARTEZL	65		
STARTGWY	67		
STOPEZL	68		
UNMARK	70		
Chapter 4. AON/SNA Commands	71		
AONSNA	72		
AONX25	73		
APPN	74		
CHGSNBU	75		
CHGSPEED	76		
DISPOOL	77		
DISSNBU	79		
DSPSNBU	80		
LISTSNBU	83		
LUDRPOOL	84		

Chapter 3. AON Base Commands

This section contains all base Automated Operations Network (AON) commands, which are listed in alphabetical order.

ACTMON

Syntax

ACTMON



Purpose of Command

The ACTMON command is a panel synonym that displays, adds, and changes the Active Monitoring Control file entry for a resource. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Operand Descriptions

resname

The name of the resource for which you are adding or changing active monitoring intervals.

Usage Notes

- This command is full-screen only.
- If you enter a resource name and if the resource has no active monitoring specified, the Add panel is displayed. If there is a definition set, the Change panel is displayed. If you do not specify a resource name, the list panel of resources that have active monitoring set is displayed.

Examples

To add or change active monitoring for the resource TA1P523A, type:

```
ACTMON TA1P523A
```

AON

Syntax

AON

▶▶—AON—▶▶

Purpose of Command

The AON command is a panel synonym that displays the AON: Operator Commands Main Menu panel. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AON command operates in full-screen mode only.

Examples

To display the AON: Operator Commands Main Menu panel, type:

AON

AONAIP

Syntax

AONAIP

```
▶—AONAIP resname,restype,reqtype—————▶  
  
▶, [ other_netid ], [ nqn_netid ], [ gw_flag ], [ sscp_name ]————▶
```

IBM-Defined Synonyms

Command or Operand	Synonym
AONAIP	EZLESAIP

Purpose of Command

The AONAIP command provides the ability to set and reset the RODM Automation in Progress (AIP) operator status for resources being monitored or viewed in NMC. AON only sets the operator status for resources that it can recover. You can also update the Operator Intervention View, using request type *ADD* for the specified resource.

Operand Descriptions

resname

The name of the resource.

restype

The *restype* parameter must be one of the following:

- CDRM
- LINE
- LINKSTA
- NCP
- PU

Note: To enable or disable which resource types are supported by AONAIP, refer to the description of the ENVIRON AIP control file entry in the *Tivoli NetView for z/OS Administration Reference*.

reqtype

The operation to be performed. The following values are supported:

SET Sets the AIP status for the resource.

RESET

Resets the AIP status for the resource.

SET/OIV

Sets the AIP status and removes the resource from the Operator Intervention View.

RESET/OIV

Resets the AIP status and adds the resource to the Operator Intervention View.

Note: To update the Operators Intervention View independently of AIP status, see the AONNOIV command for more information.

other_netid

The NETID of the resource if owned by another domain.

nqn_netid

The NETID of a resource as specified in its network qualified name.

gw_flag

A flag. When set to Y will cause the AIP operator status to be updated for both the NCP and Gateway NCP resources. Otherwise, only the NCP resource will be changed.

sscp_name

The VTAM SNA node name of the reporting VTAM. This is only required when the resource is owned by another VTAM domain.

Usage Notes

- Commas are required between each of the parameters.
- The first three parameters are required, the remaining parameters are optional.
- AONAIP must be issued in or forwarded to the domain where the target RODM is to be updated.
- The AIP and OIV functions will use the RODM user ID `domainid` concatenated with AON when accessing RODM.

Examples

The following example sets the AIP operator status for an NCP:
AONAIP NCP01,NCP,SET

AONENABL

Syntax

AONENABL

▶▶—AONENABL—▶▶

Purpose of Command

The AONENABL command is a panel synonym that provides a full-screen selection list to enable, disable, initialize and display product information about AON and its components. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AONENABL command operates in full-screen mode only.

Examples

To display the Enable/Disable Automation panel, type:

AONENABL

AONGW

Syntax

AONGW

▶▶—AONGW—▶▶

Purpose of Command

The AONGW command displays the Gateway Sessions panel, which you use to issue gateway commands to other NetView domains. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AONGW command operates in full-screen mode only.

Examples

To display the Gateway Sessions panel, type:

AONGW

AONHD

Syntax



IBM-Defined Synonyms

Command or Operand	Synonym
AONHD	AONHED, AONHEAD

Purpose of Command

The AONHD command accesses the help desk, which shows you how resources are connected within the network and helps you perform problem determination for failed resources. To use the AONHD command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource to be searched.

Usage Notes

- Different types of networks require different problem determination tasks, so the automation components have their own help desk components. Some resources may be defined to more than one automation component. Therefore, if AON finds that the resource is defined to more than one automation component, you must choose which automation component's help desk component you want to use.
- Use the AONHD command with no parameters to display the entry panel to the help desk.
- The AONHD command always operates in full-screen mode.

Examples

To display the help desk for the resource named TA1P523A, type:

```
AONHD TA1P523A
```

AONINFO

Syntax

AONINFO

▶▶—AONINFO—▶▶

Purpose of Command

The AONINFO command displays the comprehensive tutorial for the entire AON feature. The comprehensive tutorial includes overview information, the operator commands, and information about the automation components.

Usage Notes

The AONINFO command operates in full-screen mode only.

Examples

To display the comprehensive AON tutorial, type:

AONINFO

AONINIT

Syntax

AONINIT

▶▶—AONINIT—▶▶

Purpose of Command

The AONINIT command is a panel synonym that reloads the automation table and AON control file. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AONINIT command operates in full-screen mode only.

Examples

To display the AON: Reinitialize Automation panel, panel, type:
AONINIT

AONMAINT

Syntax

AONMAINT

▶▶—AONMAINT—▶▶

Purpose of Command

The AONMAINT command is a panel synonym that provides a full-screen interface to AON Task and Log Maintenance. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AONMAINT command operates in full-screen mode only.

Examples

To display the AON: Task and Log Maintenance panel, type:

AONMAINT

AONNOIV

Syntax

AONNOIV

```
▶▶—AONNOIV— rename restype —┬— ADD —▶▶  
                                └— DELETE —▶▶
```

Purpose of Command

The AONNOIV command is used to add and delete a resource from the Operator Intervention View (OIV). The OIV is a network view intended to be used to identify failed resources, which automation has attempted to recover, but cannot. An operator must perform some other task to recover the resource.

Operand Descriptions

rename

The rename to be added to the Operator Intervention View. The resource must already be defined to RODM.

restype

The restype of the resource.

ADD

Adds the resource to the Operator Intervention View.

DELETE

Deletes the resource from the Operator Intervention View.

Usage Notes

- To use the OIV function, the AIP function must be enabled.
- If the OIV timer function is enabled the resource will be automatically removed from the view when its display status is set to satisfactory (even if AON did not detect a resource recovery).
- The AONNOIV function does not create a RODM entry object for the specified resource. Instead, it connects existing objects to the specified view.

Examples

To display the Operator Intervention View, type:

```
AONNOIV NET1CDRM CDRM ADD
```

If the AIP and OIV functions are enabled and the NET1CDRM resource is defined to RODM, the resource will be added to the Operator Intervention View.

AONTAF

Syntax

AONTAF

▶▶—AONTAF—▶▶

Purpose of Command

The AONTAF command is a panel synonym that displays the TAF Sessions panel, which you use to start and stop application sessions with other NetView domains. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Usage Notes

The AONTAF command operates in full-screen mode only.

Examples

To display the TAF Sessions panel, type:

AONTAF

AONTASK

Syntax

AONTASK

▶▶—AONTASK—▶▶

Purpose of Command

The AONTASK command is a panel synonym that provides a full-screen list display of the NetView tasks and operators.

Usage Notes

The AONTASK command operates in full-screen mode only.

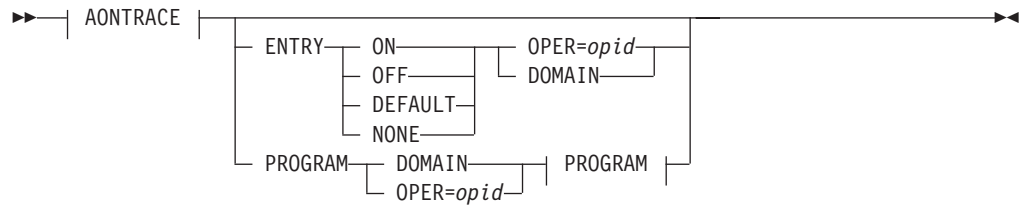
Examples

To display the AON: Task/Operator Display panel, type:

AONTASK

AONTRACE

Syntax



AONTRACE

|—AONTRACE—|

PROGRAM:

|—PROGRAM=(programe, traceops)—|

Purpose of Command

The AONTRACE command sets trace controls to allow product debugging without editing the code. To use the AONTRACE command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

ENTRY

Changes settings for entry and exit tracing. You can use entry and exit tracing for command lists, REXX programs, and command processors. Choose one of the following trace options for entry and exit tracing:

ON

Starts entry and exit tracing.

OFF

Suppresses entry and exit tracing.

DEFAULT

Sets the trace options to the domain-wide default.

NONE

Disables both entry/exit and program tracing for improved performance.

OPER=opid

Runs the trace against the operator ID you specify.

DOMAIN

Runs the trace against your domain.

PROGRAM

Specifies program tracing. Use the following values with the PROGRAM= keyword:

progrname

The name of the program you want to trace. You must know the program name to use this parameter. No defaults are available.

traceops

The interpreted REXX trace options you can use are:

- R** Result option. The result option is useful for general debugging, because it traces all clauses before it runs and traces the final results when it evaluates the expressions.
- I** Intermediate option. This option traces all clauses before it runs and traces any intermediate results during expression evaluation and substitution.
- C** Command option. This option traces all commands before it runs and displays any error return code from the commands.
- E** Error option. This option traces any command that results in error or failure and also displays any error codes that return from the command.
- F** Failure option. This option traces any command that results in failure. This option is the same as the Trace Normal command.
- L** Label option. This option traces all labels passed when it is running. Use this option when you want to know all subroutine calls and signals.
- O** Off option. This option turns tracing off and resets any previous trace settings.

Note: If the program being traced is a command list program, C, E, and O are valid and all other selections results in a trace all.

Usage Notes

- You can run traces against any operator ID on your domain.
- You can run traces against your own domain only.
- Generally, use entry and exit tracing first to help you perform problem determination. If you need more information, run the trace against the program you suspect is the source of the problem.
- You can run these traces against NetView command lists or REXX programs.
- The AONTRACE command can be issued in line-mode. Therefore, you can issue it from within your own routines.

Examples

To set entry and exit tracing for the operator ID, OPER1, type:

```
AONTRACE ENTRY ON OPER=OPER1
```

To set entry and exit tracing for your domain, type:

```
AONTRACE ENTRY ON DOMAIN
```

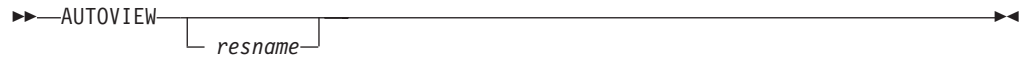
To set program tracing against the EZLEVIEW and FKXEVIEW programs as they affect the operator ID, OPER1, type:

```
AONTRACE PROGRAM OPER=OPER1 PROGRAM=(EZLEVIEW,I),PROGRAM=(FKXEVIEW,R)
```

AUTOVIEW

Syntax

AUTOVIEW



IBM-Defined Synonyms

Command or Operand	Synonym
AUTOVIEW	DISNODE

Purpose of Command

The AUTOVIEW command displays the summary of information for a specified resource. To use the AUTOVIEW command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource to be displayed.

Usage Notes

If you specify a resource name, the AUTOVIEW command searches through all the automation components to determine if the resource is defined. If the resource is defined in only one place, the summary information is displayed immediately.

The information shown for a resource varies according to which automation component is displaying the information. Therefore, if the resource is defined to more than one automation component, AON displays a selection panel so you can select which automation component you want to have display the information.

Examples

To display the AutoView summary for the resource named TA1P523A, type:

```
AUTOVIEW TA1P523A
```

CDLOG

Syntax

▶▶—CDLOG—◀◀

Purpose of Command

The CDLOG command is a panel synonym that displays the Cross-Domain Logon panel, which you use to log on and issue commands to other NetView domains. To use the CDLOG command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Usage Notes

The CDLOG command operates in fullscreen mode only.

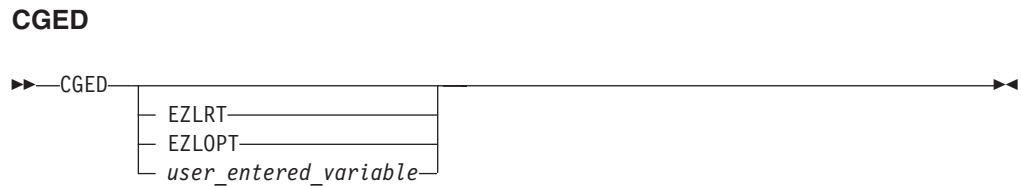
Examples

To display the Cross Domain Logon panel, type:

```
CDLOG  
CGED
```

CGED

Syntax



Purpose of Command

The CGED command displays the common global variables (CGLOBALs) used by AON. From this panel, you can add, change, or delete common global variables. To use the CGED command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

EZLRT

Displays the common global variables for resource types or option definitions. Do not delete these variables.

EZLOPT

Displays the common global variables for option definitions.

user_entered_variable

Displays the common global variables that match the search criteria you enter.

Usage Notes

- The CGED command operates in fullscreen mode only.
- CGED supports the use of an asterisk (*) as a wildcard character.
- Do not delete the common global variables for resource types.

Note: If you delete the common global variables for resource types (EZLRT or option EZLOPT), you must reload the affected loader tables by issuing the LOADTBL command.

Examples

To display the common global variables for resource types, type:

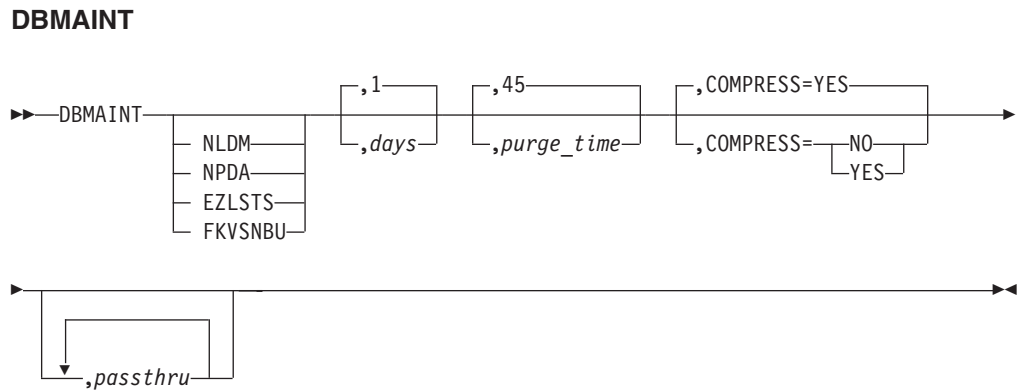
```
CGED EZLRT
```

To display all common global variables for AUTOVIEW routines, type:

```
CGED EZL*.*.AUTOVIEW
```

DBMAINT

Syntax



IBM-Defined Synonyms

Command or Operand	Synonym
DBMAINT	CLEARSTS

Purpose of Command

The DBMAINT command performs VSAM database maintenance for the NetView hardware monitor, the NetView session monitor, or the AON status file. To use the DBMAINT command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

NLDM

Performs maintenance for the NetView session monitor database.

NPDA

Performs maintenance for the NetView hardware monitor database.

EZLSTS

Performs maintenance for the AON status file.

FKVSNBU

Performs maintenance for the switched network backup (SNBU) status file. SNBU automation is a subcomponent of the AON/SNA automation component.

days

The number of days prior to the current date before which data is to be deleted. The default is 1.

purge_time

The number of minutes to wait for the database maintenance to complete. The default is 45 minutes.

COMPRESS

Determines whether the database is compressed after the records are deleted. The default is YES.

passthru

Specifies parameters which are appended unchanged to the internal ALLOCATE command which allocates the temporary database used to hold the contents of the VSAM file being reorganized. This ALLOCATE will default the following parameters if not specified: SPACE(50 50) CYL RECFM(VB) LRECL(8196) BLKSIZE(8200). DBMAINT enables VSAM reorganization in either EXPORT/IMPORT mode or REPRO/REUSE mode. In the EXPORT/IMPORT case, no ALLOCATE parameters other than primary space, secondary space, and space allocation-type are supported.

Usage Notes

- IDCAMS support in NetView is required to use this command. IDCAMS performs the compression function during database maintenance.
- This command can be issued in line mode if the parameters are entered correctly. Therefore, you can issue it from within your own routines. If the command is issued without parameters specified, a fullscreen interface is displayed for you to specify your DBMAINT input.
- DBMAINT runs under the baseoper autotask (AONBASE). When you issue a DBMAINT request from any other task, you will see a message indicating that EZLEAU03 ended with a return code of 9. This indicates that the DBMAINT request has been routed to the baseoper task.

Examples

To delete records from the NetView hardware monitor database that are older than three days old, issue:

```
DBMAINT NPDA,3,30,COMPRESS=YES
```

In this example, the DBMAINT command stops deleting records from the database if it takes longer than 30 minutes to complete the maintenance. The default value, COMPRESS=YES, is explicitly coded in this example.

DDF

Syntax

DDF

```
DDF [panel_name]
```

Purpose of Command

The DDF command shows a color-coded status display for the resources that are currently being acted upon by AON or require operator intervention for recovery. Refer to the *Tivoli NetView for z/OS Automated Operations Network Customization Guide* for more information.

Operand Descriptions

panel_name

The name of the panel as it appears in the upper left hand corner of the panel. Use this parameter to go a specific DDF panel.

Usage Notes

- The workstation from which you enter the DDF command must support 3x79 terminal extended attributes, which are blue, red, pink, green, turquoise, yellow, and white for color and blink, reverse video, and underscore for highlighting.
- If you enter DDF without the *panel_name* parameter, the main DDF panel is displayed. (The panel name for this panel is EZLPNLST).
- The DDF command operates in fullscreen mode only.

Examples

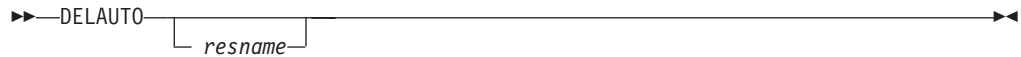
To display the panel named EZLPNL2, type:

```
DDF EZLPNL2
```

DELAUTO

Syntax

DELAUTO



Purpose of Command

The DELAUTO command removes all automation settings for a specific resource. To use the DELAUTO command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource for which you are deleting automation settings. You can delete the resource name. For example, for SNA, the resource name can also be a generic resource type, such as LU, PU, LINE, or NCP.

Usage Notes

- Because AON requires thresholds settings for the DEFAULTS resource, you cannot delete the DEFAULTS settings by using the DELAUTO command. No confirmation panel is displayed when you issue the DELAUTO command, so use this command with caution.
- This command can be issued in line mode if all parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To set all automation off for the resource TA1P523A, type:

```
DELAUTO TA1P523A
```

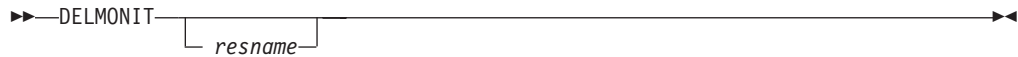
To set all automation off for all the resources that have names beginning with TA1, type:

```
DELAUTO TA1*
```

DELMONIT

Syntax

DELMONIT



Purpose of Command

The DELMONIT command deletes any recovery monitoring intervals defined for a resource. Monitor intervals determine how often AON attempts to reactivate a failing resource. To use the DELMONIT command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource for which you are deleting reactivation settings.

You can delete the resource name. For example, for SNA, the resource name can also be a generic resource type, such as LU, PU, LINE, or NCP.

Usage Notes

- Because AON requires thresholds settings for the DEFAULTS resource, you cannot delete the DEFAULTS settings by using the DELMONIT command.
- This command can be issued in line mode if all parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

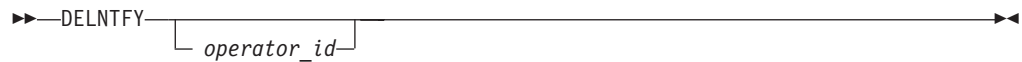
To issue the DELMONIT command against a resource, type:

```
DELMONIT TA1P523A
```

DELNTFY

Syntax

DELNTFY



Purpose of Command

The DELNTFY command removes an operator ID from the list of valid notification operator IDs. To permanently remove the operator ID from the list, your system programmer must edit the control file. Otherwise, the operator ID appears in the list the next time AON initializes. To use the DELNTFY command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

operator_id

The operator ID for the notification operator you want to delete from the list of valid operators.

Usage Notes

- Because AON requires thresholds settings for the DEFAULTS resource, you cannot delete the DEFAULTS settings by using the DELNTFY command. If you want to keep an operator ID in the list of notification operators and turn off notifications temporarily, use the NOTIFY=N option on the SETNTFY command. See SETNTFY in the NetView online help for more information.
- If you have specified operator message classes that you do not want to send to an operator ID, the only way to delete those assigned message classes from the command line is to delete the entire operator ID and then use the SETNTFY command to reset the message classes. You can delete message classes easily from the operator interface.
- The DELNTFY command can be issued in line-mode. Therefore, you can issue it from within your own routines.

Examples

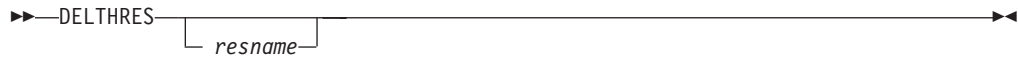
To delete the operator ID, OPER1, type:

```
DELNTFY OPER1
```

DELTHRES

Syntax

DELTHRES



Purpose of Command

The DELTHRES command deletes all threshold settings for a resource or a group of resources. AON uses thresholds to determine when to send messages to notification operators about resource problems detected through passive monitoring and when to stop recovery attempts for those resources. To use the DELTHRES command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

Name of the resource for which you are deleting the automation thresholds. The resource can be a specific resource name or, more frequently, a generic resource type (for example, DEFAULTS, LU, PU, CDRM, APPL, NCP).

Usage Notes

- Because AON requires thresholds settings for the DEFAULTS resource, you cannot delete the DEFAULTS settings by using the DELTHRES command. If you attempt to delete the settings for DEFAULTS, AON issues an error message.
- This command can be issued in line mode if the parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To delete all threshold settings for all resources which have names beginning with TA1, type:

```
DELTHRES TA1*
```

DETAIL

Syntax

DETAIL

▶▶—DETAIL—┐
 └ *resname* ┘—————▶▶

Purpose of Command

The DETAIL command queries DDF storage for details regarding a specified resource and displays the data in a DDF DETAIL panel format.

Operand Descriptions

resname

Specifies the resource name for which to search. If the resource is in a distributed host network, you must specify the resource name as follows:

network_name.resname

where *network_name* is the SYSNAME specified for that network in the EZLTREE member and in the ENVIRON SETUP, SYSNAME= keyword in the control file on the distributed host.

Usage Notes

- If no DDF details exist for the resource, you see an error message.
- Although the panel displayed from this command resembles the DDF DETAIL panel, DDF has not been invoked and DDF commands are not valid from this panel.
- To find the name of the system for the resource, browse the EZLTREE member. You do not need to specify the system name if the resource is on the same system where you are working.
- If you enter the DETAIL command with no resource name, the Operator Command Interface: DETAIL panel is displayed.

Examples

To display the DDF details for TA1P523A, type:

DETAIL TA1P523A

DISAUTO

Syntax

DISAUTO



Purpose of Command

The DISAUTO command displays all automation settings for a specific resource or a group of resources.

Operand Descriptions

resname

The name of the resource for which you are displaying automation settings. The resource name can be the name of one resource, DEFAULTS, or a generic resource type (for example, LU, PU, LINE, NCP). You can also use the wildcard characters * and %.

Usage Notes

This command can be issued in line mode if the parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To view automation settings for all NCPs, issue:

```
DISAUTO NCP
```

DISNTFY

Syntax

DISNTFY



Purpose of Command

The DISNTFY command displays the settings for notification operators.

Operand Descriptions

operator_id

The operator ID for the notification operator.

Usage Notes

This command can be issued in line mode if the parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To display the notification settings for operator ID, OPER1, type:

```
DISNTFY OPER1
```

DISTHRES

Syntax

DISTHRES



Purpose of Command

The DISTHRES command displays all threshold settings for a resource or a group of resources. AON uses thresholds to determine when to send messages to notification operators about resource problems that have been detected through passive monitoring and when to stop recovery attempts for those resources.

Operand Descriptions

resname

The name of the resource for which you are displaying the automation thresholds. The resource can be a specific resource name or, more frequently, a generic resource type (for example, DEFAULTS, LU, PU, CDRM, APPL, NCP).

Usage Notes

This command can be issued in line mode if the parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To display all threshold settings for all resources which have names beginning with TA1, type:

```
DISTHRES TA1*
```

DSPCFG

Syntax

DSPCFG



IBM-Defined Synonyms

Command or Operand	Synonym
DSPCFG	DISCFG

Purpose of Command

The DSPCFG command displays the configuration data for a specific entry and type from the control file. When the data is displayed, you can make additions, changes, and deletions. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Operand Descriptions

entry_name

The name of the entry in the control file.

type_name

The type within the entry in the control file. You can use the following wildcard characters:

- * Multiple character wildcard. For example, PU0* finds the resource, PU01, and all the other resources that have names starting with PU0.
- % Single character wildcard. For example, P%%1 finds the resource PU01.

Usage Notes

The DSPCFG command operates in fullscreen mode only.

Examples

To display the data for all the AUTOOPS control file entries, issue one of the following:

```
DSPCFG AUTOOPS
```

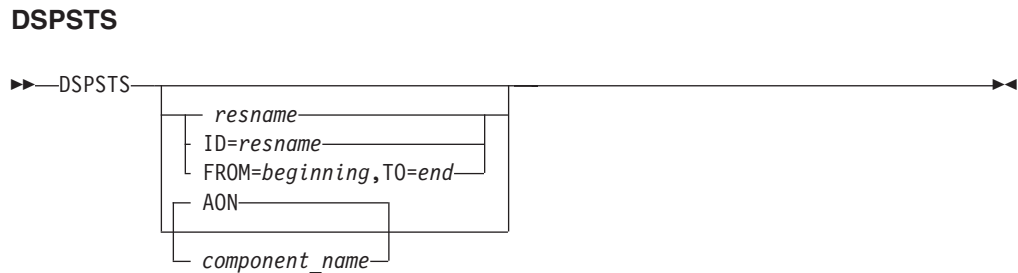
```
DSPCFG AUTOOPS *
```

To display the data for all AUTOOPS control file entries that include a type of NETOPER, type:

```
DSPCFG AUTOOPS NETOPER
```

DSPSTS

Syntax



IBM-Defined Synonyms

Command or Operand	Synonym
DSPSTS	DISSTS

The DSPSTS command displays the information contained in the automation status file. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Operand Descriptions

resname

Any resource identified in the control file. Input parameters can be 32 characters in length.

FROM=beginning,TO=end

Search parameters for the status file. The variables *beginning* and *end* can be any alphabetic character from A to Z or any number from 1 to 9. AON searches for records that begin with the letter or number you specify with FROM=*beginning* and includes all those records through those that start with the letter or number you specify with TO=*end*. Input parameters can be 16 characters in length.

AON

Optional parameter that causes AON to search all the status file records. AON is the default.

component_name

Parameter that causes AON to search for status file records that were created by a particular automation type. The components are:

SNBU Searches for records created by the AON/SNA switched network backup (SNBU) automation type.

Usage Notes

- The FROM=*beginning,TO=end* search parameter searches for alphabetic characters first and searches for numbers. To display all status file records, specify FROM=A,TO=9.
- The DSPSTS command operates in fullscreen mode only.

Examples

To display all the status file records for the resource named TA1P2460, type:

```
DSPSTS TA1P2460
```

To display all the status file records, type:

```
DSPSTS FROM=A,T0=9
```

ILOG

Syntax

▶▶ ILOG [*search_string*] ▶▶

Purpose of Command

This command is used to display the Inform Log operator panel. Any parameters following the command name are used as a search string on the contact name field portion of each inform log entry.

Operand Descriptions

search_string

An optional parameter defining a search string against the inform log name field. Using *search_string* enables the operator to limit the inform log entries displayed.

Usage Notes

The ILOG routine is also known as EZLEINFL or INFORMLG.

The log function must be enabled in the inform policy member, and the inform policy member must be loaded prior to using the ILOG command.

Even if a log file exists, it is not accessible when an inform policy member is not currently active. Only the active file is accessible through ILOG, use the NetView BROWSE command to view previously active logs.

Inform Log Utility

The following are the functions of the Inform Log Utility:

At the top of the help for Inform Log Utility, you will find the following:

1. ACKNOWLEDGE – changes the displayed status to ACKNOWLEDGE.
2. REINFORM – reissues the message against the active inform policy member using the original policy name.
3. REINFORM/NEW – same as REINFORM, but provides a pop-up with the original message text. This text can be edited or replaced prior to confirming the reissue.
4. DELETE – removes the entry from the inform log.

The following are a few of the PF keys that appear at the bottom of the help for Inform Log Utility:

- F5 REFRESH – will reread the current inform log and display this version to you.
- F9 SEARCH – will provide the ability to locate just those entries which might be important to you. For example, searches of the actual message and the name field are planned, as well as a time period-based search.

Note: Multiple actions can occur for one event, depending on the definitions in the inform policy member.

The following are statements regarding the Log Utility function:

1. Only the entry marked with Acknowledged is marked, not any other inform entries generated by the same event.
2. When an entry is reissued, only that entry (not its companions) is deleted. And only one new inform log action is logged (even though multiple actions might have been generated).
3. Reissued inform actions are NOT duplicates. Instead, the inform policy is re-consulted and the appropriate inform actions are taken (based on the same policy, but new time/date).

Note: The policy member might have changed; the policy which drove the initial inform might no longer be defined or active for the current time frame.

4. INFORM (also known as EZLECALL) does not log its actions, since it is intended as an interface that operators will use to page someone directly.

INFORM (AON)

Syntax

INFORM

►►—INFORM *policy_name* —┐
 └ *message* ┘

Purpose of Command

The INFORM command is a REXX routine that generates an immediate inform action based on the INFORM policy member. An operator can enter the name of the individual or group policy to contact, and optionally specify message text. The INFORM command requires that an INFORM/CONTACT policy entry exists for this purpose. By default command actions implemented using INFORM are not logged. Refer to the *Tivoli NetView for z/OS Administration Reference* for more information about how to enable logging of INFORM command actions.

Operand Descriptions

policy_name

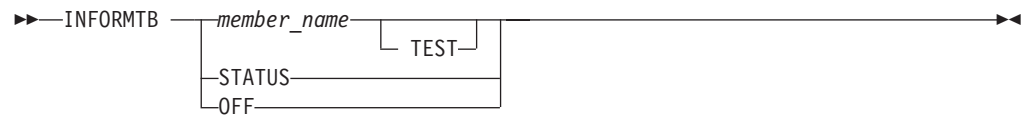
Specifies which INFORM policy or group name to use, which determines which individual(s) to contact.

message

Specifies the message text to be sent to the contact. The message text must be consistent with the CONNECTION types specified in the INFORM policy member. If no message is specified, either the default inform message or the message defined in the INFORM policy is sent.

INFORMTB (AON)

Syntax



Purpose of Command

The INFORMTB is a REXX routine that processes and loads the DSIPARM member containing the INFORM policy. INFORMTB can also be used to verify the INFORM policy member prior to activating it. Refer to the *Tivoli NetView for z/OS Administration Reference* for more information about the INFORM policy member.

Operand Descriptions

member_name

The name of the DSIPARM member that contains the inform policy definitions.

TEST

The inform member will be checked for syntax only, but will not be activated upon successful completion.

STATUS

The current state of the INFORM policy. If the status is active, the status indicates whether a member is active, who activated it, and when.

OFF

Specifies that the INFORM member should be disabled if currently active.

Examples

The following is an example of the INFORMTB command:

```
INFORMTB BEEPER1 TEST
```

This example checks the INFORM policy member BEEPER1 for syntax, but does not activate the INFORM policy member if it completes successfully.

The following command outputs an INFORM policy member named INFNIGHT:

```
INFORMTB INFNIGHT
```

The following is the output for this member:

```
001   INFORM PAUL,SP=NT8D1005;
002   CONTACT ONCALLDAY=WEEKDAY,
003     ONCALLTIME=08:00 to 17:00
004     CONNECTION=EMAIL,
005     ROUTE=OPER4@RTP123.IBM.COM,
006     NAME=PAUL,
007     INTERFACE=EZLENETF,
008     MSG=A %RESTYPE% NAMED &RESNAME% FAILED DUE TO %RESSTAT%
009     PLEASE CALL 2345678;
010   INFORM PAUL,SP=NT6D1005;
011   CONTACT ONCALLDAY=WEEKDAY,
012     ONCALLTIME=17:00 to 24:00;
013     CONNECTION=ALPHAPAGE,
014     ROUTE=1234567,
```

```
015      NAME=PAUL Q,  
016      INTERFACE=EZLENETF,  
017      MSG=PLEASE CALL THE OFFICE IMMEDIATELY,  
018      COMPORT=COM1,  
019      TAPACCESS=918001234567;
```

The results of this member are:

```
EZL454I DUPLICATE POLICY OR GROUP NAME PAUL DETECTED AT LINE 10  
EZL455I PROCESSING FAILED FOR 'INFORMTB INFNIGHT' COMMAND
```

LOADTBL

Syntax

LOADTBL



IBM-Defined Synonyms

Command or Operand	Synonym
LOADTBL	LOAD

Purpose of Command

The LOADTBL command reinitializes all the common global variables in the specified definition table.

Operand Descriptions

table_name

The name of the definition table to reload. The AON base product and each of its automation components have separate definition tables. The definition table names are:

EZLTABLE

AON base definition table

FKVTABLE

AON/SNA if installed

FKXTABLE

AON/TCP if installed

user Any user-created definition table.

Usage Notes

- The LOADTBL command reloads one table per invocation. To reload more than one table, reissue the command.
- The LOADTBL command can be issued in line-mode. Therefore, you can issue it from within your own routines.

Examples

To reinitialize the common global variables in the AON option definition table, type:

```
LOADTBL EZLTABLE
```

MARK

Syntax

MARK

▶▶—MARK *root_comp.rv(opid)*—————▶▶

Purpose of Command

The MARK command assigns a DDF entry to an operator. You must supply the data used to identify the DDF entry as well as the ID of the operator to be assigned to the entry.

Operand Descriptions

root_comp

Defines the root component name as defined in the DDF entry to be assigned.

rv The resource name as it appears in the RefValue field of the DDF entry to be assigned.

opid

This is the operator ID to whom this entry is to be assigned.

Usage Notes

The *root* and *rv* parameters are required to issue this command from a command line. If the *opid* parameter is not specified, DDF assigns the entry to the operator ID issuing the MARK command.

If the entry to be signed out is already signed out then no action is taken. You first need to issue the UNMARK command before attempting to reassign the entry to another operator.

When assigned, a resource will stay assigned during recovery monitoring, as long as the operator who MARKed the resource is logged on. If the operator assigned to the resource logs off, the next recovery monitoring timer will cause the resource to be unassigned.

Examples

To assign the DDF entry for NCP001 under the domain CNM01 to the operator OPER1, type:

```
MARK CNM01.NCP001(OPER1)
```

NLOG

Syntax

NLOG



Purpose of Command

The NLOG command browses the automation log.

Operand Descriptions

ACTIVE

Browses the automation log that is currently active.

PRIMARY

Browses the primary log file.

SECONDARY

Browses the secondary log file.

INACTIVE

Browses the automation log that is currently inactive.

CLEAR

Clears the inactive log if it is not being browsed.

CLEARPRIM

Clears the primary log if not in use.

CLEARSCND

Clears the secondary log if not in use.

ENDPRIM

Forces off operators browsing the primary log (issued by automation operators only).

ENDSCND

Forces off operators browsing the secondary log (issued by automation operators only).

Usage Notes

- The NLOG command displays the automation log and bypasses the operator interface.
- The NLOG command operates in fullscreen mode only.

Examples

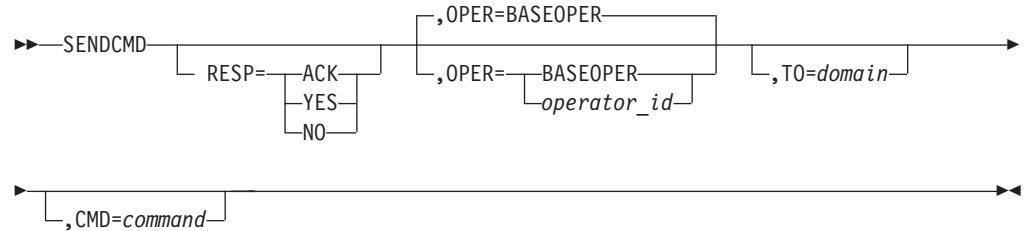
To display the active automation log, type:

```
NLOG
```

SENDCMD

Syntax

SENDCMD



Purpose of Command

The SENDCMD command allows you to route commands to other domains using the gateway sessions. When you use gateway sessions, an automation operator known as a gateway operator logs on to the other domain, so it is unnecessary for you to have your own session with that domain. To use the SENDCMD command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

RESP=

Parameters are defined as follows:

- ACK** Specifies that a notification is required indicating whether the command was run on the target domain or not.
- YES** Specifies that the output from the command being issued is to be displayed on the current domain.
- NO** Specifies that neither a response nor acknowledgement is required for the command that is being issued.

OPER=

The command is run on this operator ID at the target domain. If *operator_id* is not specified, the command is run by the AON automation operator, BASEOPER, on the target domain.

TO=domain

Specifies the domain to which the command is being issued.

CMD=command

Any valid NetView or AON command except LOGOFF.

Usage Notes

- If the command you are sending to another domain using the SENDCMD command contains a comma, you must use the delimiters ' or " for fullscreen mode or / if you are using line mode. For example:
SENDCMD RESP=YES,TO=CNM02,CMD='D NET,CDRMS'
- If you enter SENDCMD without any parameters, a fullscreen panel is displayed.

- Do not use SENDCMD to issue commands that require a confirmation at the target domain (such as REPLY).
- To learn the status of the gateways, use the operator interface.

Examples

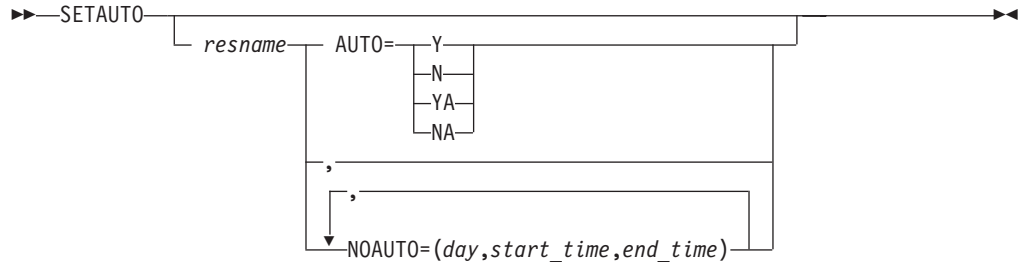
To issue a WHO command to the NetView domain CNM99, type:

```
SENDCMD RESP=YES,OPER=BASEOPER,TO=CNM99,CMD=WHO
```

SETAUTO

Syntax

SETAUTO



Purpose of Command

The SETAUTO command sets recovery automation for a specific resource or a group of resources. To use the SETAUTO command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource for which you are changing automation settings. The resource name can be the name of one resource, DEFAULTS, or a generic resource type. If you specify DEFAULTS, the settings affect all resources that do not have other settings explicitly set for them. You can set automation for entire groups of resources by type such as LU, PU, LINE, and NCP, or for entire days such as MONDAY, HOLIDAY, or JAN/1/2000. You can also use the wildcard characters * and %. The most specific setting coded for a resource issued to control whether automation recovery is on or off for the resource.

AUTO

Defines whether automation is allowed for the resource. Use the following values with the AUTO keyword to determine whether automation is running:

Y Sets recovery on.

N Sets recovery off.

YA

Sets recovery on for the specified resource and its lower nodes.

NA

Sets recovery off for the specified resource and its lower nodes.

Note: YA and NA are valid only for SNA resources that do not contain wildcard characters (* and %). If entered, recovery is set on or off, but the AON ignores the lower nodes.

NOAUTO

Defines specific times when automation does not occur. Contrast with AUTO=N, which sets automation to off all the time. Define all of the following values when you use the NOAUTO keyword:

day

Specifies the days when recovery is set to off and is one of the following:
MONDAY or MON

TUESDAY or TUE
 WEDNESDAY or WED
 THURSDAY or THU
 FRIDAY or FRI
 SATURDAY or SAT
 SUNDAY or SUN
 Month/date_spec/year (examples: JAN/1/2000 or JAN/LAST-30/2000)
 Month/day_spec/year (examples: JAN/SAT/1ST/2000 or
 JAN/SAT/LAST-4/2000) In this case SAT cannot be SATURDAY
 because only three digits are allowed.
 Special_day_name (example: My_Birthday or HOLIDAY or
 NEW_YEARS_DAY)

Use an asterisk (*) to set automation to off during some interval every day.

start_time

Sets automation to off starting at this time. Specify the time in the 24-hour format *hh:mm* where *hh* is a number between 00 and 23 and *mm* is a number between 00 and 59.

end_time

Determines the end of the interval when automation is not active for the resource. Specify the time in the 24-hour format *hh:mm* where *hh* is a number between 00 and 23 and *mm* is a number between 00 and 59.

You can specify up to five settings for NOAUTO to begin. To enter more NOAUTO settings, press **PF10** to save the settings, then press **PF8** to add more settings.

Usage Notes

- You must specify an AUTO or NOAUTO keyword on the SETAUTO command. If you do not specify AUTO, you must schedule at least one NOAUTO interval for the resource.
- It is possible for automation to always be inactive for a resource even if you specify AUTO=Y or AUTO=YA, if you also specify NOAUTO=*,00:00,23:59. Pay close attention to the intervals you specify when you use the NOAUTO keyword.
- This command can be issued in line mode if all parameters are correctly entered. Therefore, you can issue it from within your own routines. If no parameters are specified, a fullscreen interface is displayed.

Examples

To have automation active for the resource, TA1TT167, every day *except*:

- Saturday and Sunday from 8:00 a.m. to 10:00 a.m.
- Every day from 3:00 p.m. (15:00) to 5:00 p.m. (17:00)

issue:

```
SETAUTO TA1TT167 AUTO=Y,NOAUTO=(WEEKEND,08:00,10:00),NOAUTO=(*,15:00,17:00)
```

To set automation on for all NCPs *except* during the interval between 1:00 p.m. and 2:00 p.m. on Saturday and Sunday, issue:

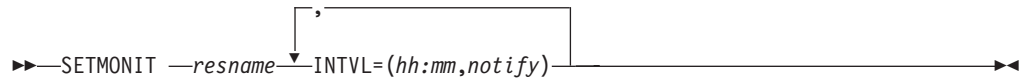
```
SETAUTO NCP NOAUTO=(WEEKEND,13:00,14:00)
```

SETMONIT

Syntax

SETMONIT

▶—SETMONIT —*resname*—INTVL=(*hh:mm,notify*)—▶



Purpose of Command

The SETMONIT command sets the intervals AON uses during recovery monitoring. At these recovery monitoring intervals, AON monitors the failed resource to see if it has recovered. For SNA resources, AON also attempts to reactivate the failed resource at each recovery monitoring interval. Finally, you use the SETMONIT command to specify whether AON sends messages to the notification operators at recovery monitoring intervals. To use the DELTHRES command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource to be recovered.

INTVL

The reactivation interval setting. The variables on this setting are:

hh:mm

The length of the interval between reactivation attempts expressed as hours (*hh*) and minutes (*mm*).

notify

The setting that determines whether messages are sent to the notification operators when AON attempts to reactivate the resource. The settings can be:

Y Send messages to the notification operators.

N Do not send messages to the notification operators.

YF Send messages to the notification operators and repeat recovery monitoring at the last interval specified until the resource recovers or the control file is reloaded.

NF

Repeat recovery monitoring at the last interval specified, but do not send messages to the notification operators.

Usage Notes

- Using the SETMONIT command, you can specify up to 12 recovery monitoring intervals for a resource.
- If the recovery monitoring intervals already exist for the resource, the SETMONIT command replaces the existing intervals.

- Use the YF and NF settings on the last interval only. If you use YF or NF for any interval before the last one, AON ignores that interval.
- The SETMONIT command can be issued in line mode. Therefore, you can issue it from within your own routines.

Examples

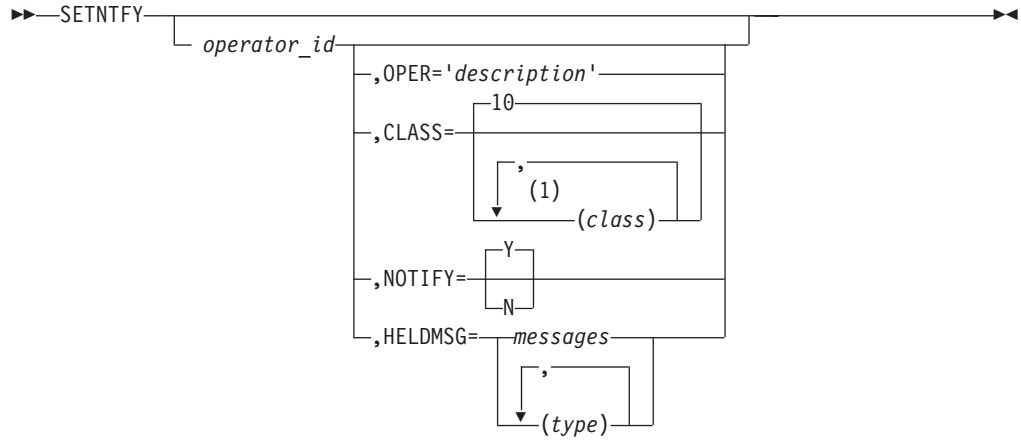
To set recovery monitoring for all CDRMs, type:

```
SETMONIT CDRM INTVL=(02:10,Y),  
              INTVL=(01:10,N),  
              INTVL=(01:20,Y),  
              INTVL=(01:45,YF)
```

SETNTFY

Syntax

SETNTFY



Notes:

- 1 You can specify up to 10 variables.

IBM-Defined Synonyms

Command or Operand	Synonym
SETNTFY	SETALERT

Purpose of Command

The SETNTFY command controls the settings for notification operators. With the SETNTFY command, you can add a notification operator, turn messages to the notification operator on and off without deleting the notification operator, select message classes for the notification operator, and determine whether automation messages sent to the notification operator's command facility are held. To use the SETNTFY command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

operator_id

The operator ID of the notification operator.

CLASS

The classes of messages this operator is to receive. The parentheses are optional if you code one value for *class*, but are required if you code two or more values. You can define up to 10 message classes for an operator. Default class is 10.

NOTIFY

Specifies whether messages are actually sent to this notification operator. You can use this parameter to temporarily stop the notification messages without actually deleting the notification operator. The default is Y.

description

A brief description (20 characters or shorter) for the notification operator, which is usually the name of the operator. You must enclose the description inside single quotes.

HELDMSG

The types of messages that should be held on the notification operator's command facility. Held messages remain on the command facility until you clear them. You can specify multiple types, as follows:

I or INFO

Informational messages

W or WARN

Warning messages

E or ERROR

Error messages

A or ACTION

Action messages

Usage Notes

- You can specify the keywords CLASS=, OPER=, NOTIFY=, and HELDMSG= in any order. Use only the keywords you need.
- You must put the parameters for the HELDMSG keyword in parentheses, for example, HELDMSG=(I).
- The SETNTFY command can be issued in line mode if the resource name is provided. Therefore, you can issue it from within your own routines. If the resource name is provided with no other parameters specified, all defaults are used (CLASS=10, NOTIFY=Y).
- If no parameters are issued with this command, a fullscreen panel is displayed showing all valid notification operators.

Examples

To add the operator ID, OPER1, to the list of valid notification operators using the default settings (CLASS=10, no description, no held messages), type:

```
SETNTFY OPER1
```

To hold all error messages for OPER1, type:

```
SETNTFY OPER1 HELDMSG=(E)
```

To set OPER1 as a valid notification operator, but specify that AON does not send notifications, type:

```
SETNTFY OPER1 CLASS=10,NOTIFY=N,HELDMSG=(I,E,W,A),OPER='J Smith'
```

SETTHRES

Syntax

SETTHRES

```
▶▶—SETTHRES resname—▶▶  
┌, CRIT=(nn,hh:mm)┐  
┌, FREQ=(nn,hh:mm)┐  
└, INFR=(nn,hh:mm)┘
```

Purpose of Command

The SETTHRES command sets thresholds for a specific resource or a group of resources. When AON detects messages that indicate resource problems through passive monitoring, AON attempts to recover the failed resource. You can use the SETTHRES command to specify whether AON sends messages to the notification operators when the infrequent, frequent, or critical thresholds are reached. To use the SETTHRES command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

resname

The name of the resource for which you are setting thresholds. The resource can be a specific resource name or, more frequently, a generic resource type (for example, DEFAULTS, LU, PU, CDRM, APPL, NCP). The three types of thresholds are:

CRIT

The critical threshold. AON stop reactivation attempts for the resource when the critical threshold is reached.

FREQ

The frequent threshold. The frequent threshold indicates that the resource is having errors often enough that action is required.

INFR

The infrequent threshold. The infrequent threshold provides early warning of intermittent resource errors.

The thresholds are defined as a number of errors within a given time span as follows:

nn The number of errors before threshold is reached. The number must be between 0 and 12.

hh:mm

The time span before threshold is reached (hours:minutes), where *hh* is a number between 0 and 99 and *mm* is a number between 0 and 59.

Usage Notes

- AON uses the thresholds coded for the resource name, DEFAULTS, as the default threshold settings.
- AON requires thresholds settings for the resource, DEFAULTS. Therefore, AON does not let you delete the DEFAULTS settings.

- You can code all three thresholds values (CRIT, FREQ and INFR) on one invocation of the SETTHRES command.
- AON keeps date and time stamps for all failures of a resource in the status file, so threshold analysis is based on the actual time span between outages (without rounding off to the whole hour or minute).
- This command can be issued in line mode if all parameters are correctly entered. Therefore, you can issue it from within your own routines. For the command to be considered correct, you must specify at least one of the CRIT, INFR, or FREQ parameters with the resource name.
- If no parameters are specified, a fullscreen interface is displayed.

Examples

To set the default thresholds settings, type:

```
SETTHRES DEFAULTS,CRIT=(2,00:14),FREQ=(2,01:00),INFR=(4,04:00)
```

To replace the critical threshold setting for the resource type, NCP, type:

```
SETTHRES NCP,CRIT=(2,02:00)
```

The values for the frequent and infrequent thresholds remain unchanged.

STARTEZL

Syntax

STARTEZL



Purpose of Command

The STARTEZL command starts AON components, which are tasks that drive the automation log, the control file, the status file, or DDF.

Operand Descriptions

ALL

Starts all of the components.

CONFIG

Starts the EZLTCFG task which is required for any policy function. You can substitute the synonyms CFG or EZLCFG for CONFIG. The CONFIG task loads the control file defined in CNMSTYLE. Any attempt to change the control file name is ignored.

DDF

Starts the Dynamic Display Facility component, the EZLTDDF task. You can substitute the synonym EZLDDF for DDF.

LOG

Starts the automation log component, the EZLTLOG task. You can substitute the synonym EZLLOG for LOG.

STATUS

Starts the status file component, the EZLTSTS task. You can substitute the synonyms STS or EZLSTS for STATUS.

Usage Notes

- Before you issue this command, check with your system programmer to determine its impact on network automation.
- STARTEZL ignores the MEMBER=parameter when specified. The policy file(s) that are loaded are read from CNMSTYLE definitions.
- After using STARTEZL to start a component or task, do not attempt to use STOPEZL to stop the component or task before receiving a message indicating that the component or task is ready and waiting for work.
- After issuing the STOPEZL CONFIG command, wait until the command has completely stopped, then issue STARTEZL CONFIG. Otherwise, all AON stops and remains inactive.
- The STARTEZL command can be issued in line mode. Therefore, you can issue it from within your own routines.

Examples

To restart all tasks, type:

```
STARTEZL ALL
```

Any tasks that were already started before you issue this command are not affected.

STARTGWY

Syntax

STARTGWY

▶▶—STARTGWY—◀◀

Purpose of Command

The STARTGWY command initializes all the gateway sessions for this NetView as defined by the GATEWAY entries in the control file. When gateway sessions are active, you can send commands to other domains without having your own sessions active on those domains. Instead, the automation operators known as gateway operators log on to the other domains and handle communications for you. To use the STARTGWY command from the operator interface, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Usage Notes

- When the gateways start successfully, the following command is displayed on the command facility:

```
EZL652I GATAN001 STARTING GATEWAY SESSIONS WITH TARGET DOMAINS
```

- For the gateway sessions to operate correctly, your system programmer must define the following entries in the control file for systems that have gateway connections:

```
AUTOOPS GATEWAY  
GATEWAY domain_id  
MONITOR domain_id  
FORWARD
```

When the GATEWAY control file entries are correct, gateway sessions are established automatically when NetView and AON initialize.

- The STARTGWY command is useful if you are testing gateway sessions, or if you want to reestablish gateway sessions without waiting for AON automation to detect the broken connection and restart the sessions.

Examples

To start all gateway sessions, type:

```
STARTGWY
```

STOPEZL

Syntax



Purpose of Command

The STOPEZL command stops AON components, which are tasks that drive the automation log, the control file, the status file, or DDF.

Operand Descriptions

ALL

Stops all the components.

CONFIG

Stops the control file component, the EZLTCFG task. You can substitute the synonyms CFG and EZLCFG for CONFIG.

DDF

Stops the Dynamic Display Facility component, the EZLTDDF task. You can substitute the synonym EZLDDF for DDF.

LOG

Stops the automation log component, the EZLTLOG task. You can substitute the synonym EZLLOG for LOG.

STATUS

Stops the status file component, the EZLTSTS tasks. You can substitute the synonyms STS and EZLSTS for STATUS.

Usage Notes

- When the STOPEZL command stops the control file task, the information in the control file is no longer available and AON automation does not operate properly.
- The STOPEZL command does not stop AON activity completely and can cause errors in your NetView log. To stop AON automation activity completely, you must disable the AON base program by using the Enable/Disable panel under the AON: Support Functions panel.
- Before you issue STOPEZL, check with your system programmer to determine its impact on the network.

Note: At some locations, your system programmer might have added authorization protection for this command. Your system programmer can tell you which commands have authorization protection active.

- If you have issued STARTEZL to start a component or task, do not use the STOPEZL command to stop the task until after you receive the message:

DSI530I *taskname*: DST IS READY AND WAITING FOR WORK

- The STOPEZL command can be issued in line mode. Therefore, you can issue it from within your own routines.

Examples

To stop the DDF component, type:

```
STOPEZL DDF
```

UNMARK

Syntax

UNMARK

►►—UNMARK *root_comp.rv(operator_id)*—◄◄

Purpose of Command

The UNMARK command removes a DDF assignment entry from the specified operator. The data used to identify the DDF entry as well as the ID of the operator to be removed must be supplied.

Operand Descriptions

root_comp

Defines the root component name as defined in the DDF entry to be unmarked.

rv The resource name as it is displayed in the RefValue field of the DDF entry is unmarked.

operator_id

The operator ID assigned to this DDF entry.

Usage Notes

- The *root* and *rv* parameters are required. If you do not specify the *operator_id* parameter, the UNMARK command uses your operator ID as the default.
- If you issue the UNMARK command for a DDF entry that is not assigned to an operator, the UNMARK command takes no action.
- Each automation component has its own syntax for specifying the name of the resource to be displayed.

Examples

To remove the operator ID, OPER1, from the DDF entry NCP001 under CNM01, type:

```
UNMARK CNM01.NCP001(OPER1)
```

Chapter 4. AON/SNA Commands

This section contains all AON/SNA commands, which are listed in alphabetical order.

AONSNA

Syntax

▶▶—AONSNA—◀◀

Purpose of Command

The AONSNA command displays the SNA Automation: Menu panel, which enables you to access the functions of AON/SNA.

Restrictions

This command operates in full-screen mode only.

AONX25

Syntax

AONX25

▶▶—AONX25—▶▶

Purpose of Command

The AONX25 command provides access to the X.25 Main Menu to manage X.25 resources.

APPN

Syntax

APPN

▶▶ APPN ◀◀

Purpose of Command

The APPN command displays the SNA Automation: APPN Commands Menu panel.

Restrictions

This command operates in full-screen mode only.

CHGSNBU

Syntax

CHGSNBU

▶▶—CHGSNBU *pu_name*————▶▶

Purpose of Command

The CHGSNBU command lets you switch SNBU lines between leased status and dial backup status. The CHGSNBU command enables you to initiate several SNBU actions independent of automation by using operator commands. You can:

- Switch a communication link to backup speed
- Return a communication link to full speed
- Switch a PU to switched network backup
- Restore a communication link to leased line operation
- Delete erroneous status information

The CHGSNBU command displays the Change Speed or Initiate/Terminate SNBU Operation panel.

Operand Descriptions

pu_name

An optional name for the PU.

Usage Notes

A modem is not a network addressable unit (NAU). You must address the modem using the network name and the address of the remote PU to which it is attached. When you use the PU name as an operand, the relevant information for that PU is automatically filled in on the resulting panel.

If you fail to delete old or erroneous Busy and Down status records, your attempts to select an alternate port might not succeed because SNBU automation cannot use a busy or down port. AON/SNA can issue the FKV850I message that indicates no ports are available, when there is nothing to prevent the dial from occurring. Erroneous status file records prevented the ports from being selected.

Restrictions

This command operates in full-screen mode only.

CHGSPEED

Syntax

CHGSPEED

▶▶—CHGSPEED *pu_name*—————▶▶

Purpose of Command

The CHGSPEED command is a synonym for the CHGSNBU command. The CHGSPEED command controls the speed of SNBU modems. The CHGSPEED command enables you to initiate several SNBU actions independently of automation by using the operator commands. You can use the CHGSPEED command to:

- Switch a communication link to backup speed
- Return a communication link to full speed
- Switch a PU to switched network backup
- Restore a communication link to leased line operation
- Delete erroneous status information

The CHGSPEED command displays the Change Speed or Initiate/Terminate SNBU Operation panel.

Operand Descriptions

pu_name

An optional name for the PU.

Usage Notes

The CHGSNBU and CHGSPEED commands are synonyms. You can use these commands interchangeably. Refer to the CHGSNBU command for usage information.

Restrictions

This command operates in full-screen mode only.

DISPOOL

Syntax

DISPOOL

▶▶—DISPOOL—▶▶

Purpose of Command

The DISPOOL command displays the switched network backup automation (SNBU automation) modem pools. Each POOL entry in the control file has a list of the lines that are available for alternate port backup. The DISPOOL command displays all the POOL entries in the control file and all the entries dynamically added with the SETPOOL command for this use of NetView. The DISPOOL command also displays the status for each of the lines in the modem pools.

The DISPOOL command displays the SNBU Modem Pools panel.

Usage Notes

Using alternate port switched network backup requires that:

- Additional lines with spare modems be defined as alternate ports in the control file.
- These lines are defined as belonging to a pool of alternate ports.

When an alternate port switched network backup is called for, the first alternate port in the pool that does NOT have a busy or down status in the status file is selected. No attempt at dialing occurs for ports marked busy or down in the status file.

The DISPOOL command shows you the status of the modem. A status of Down indicates that a previous attempt was made to use this alternate port and it failed for some reason. The SNBU automation marks the port as down to indicate to the operator that manual intervention might be required to enable switched network backup to occur over this port. After the operator has investigated the problem and fixed the disabling condition, the old and now erroneous status record for that port must be deleted using the CHGSNBU command with the delete option.

A status of Busy indicates that the modem is currently in use as an alternate port for switched network backup. The display from DISPOOL also indicates the name of the (primary) remote PU supported by switched network backup. After a record of this type is written to the status file, no further attempts at switched network backup are made using this alternate port.

As long as you use the switched network backup to return from the PU to leased line operation, either by automation or by the CHGSNBU command, the busy status records are deleted when switched network backup is disconnected. However, if switched network backup is stopped manually by the NetView MDMCNTL command, by the modem keypad, or by recycling VTAM, the busy status record for that alternate port is not deleted. It must be deleted before the port is again selected for switched network backup. The deletion can be achieved using the delete status option with the CHGSNBU command.

If you fail to delete old or erroneous Busy and Down status records, attempts to select an alternate port might not succeed because the SNBU automation cannot use a busy or down port. An EZL827I message that no ports are available can be issued, when in fact there is nothing to prevent the dial from occurring. Erroneous status file records prevent those ports from being selected.

Restrictions

This command operates in full-screen mode only.

DISSNBU

Syntax

DISSNBU

▶▶—DISSNBU—◀◀

Purpose of Command

The DISSNBU command displays the switched network backup automation (SNBU automation) resources. The DISSNBU command reads the control file and finds all the SNBU automation entries, including those for the defaults. The DISSNBU command then displays all of the SNBU automation entries in the control file, plus those dynamically added with the SETSNBU command for this use of NetView.

The DISSNBU command displays the Display SNBU Resource Definitions panel.

Usage Notes

Depending upon whether there are fanout PUs specified, it will display the results in different formats. If there are no modems with fanout PUs specified, then the results will be shown with up to 15 entries per panel. If any modems have fanout PUs attached or the phone number is too long to be completely displayed, tab to that line and press Enter to see another panel that contains the complete entry in the control file for that resource.

Restrictions

This command operates in full-screen mode only.

DSPSNBU

Syntax

DSPSNBU



Purpose of Command

The DSPSNBU command provides status file information about automation status, error thresholds, and time and date information for error conditions. You can retrieve, display, and delete status file information.

Operand Descriptions

id The resource name of the terminal ID.

from
Beginning of range to enable search to be specific.

to End of range to enable search to be specific.

Usage Notes

You must use both the *from* and *to* parameters if you select either the *from* or *to* parameter.

If you issue the DSPSNBU command from a command line without one of the optional parameters, AON displays the Display Status Data panel.

```

EZLK7200          Display Status Data          CNM01

Select an Option

  _ 1. Id _____
     2. From _____
        To _____

Select a Component

  _ 1. AON Base
     2. AON SNA Automation - SNBU Option

Command ==>
F1=Help      F2=End          F3=Return          F6=Roll
              F12=Cancel

```

Figure 7. The Display Status Data Panel

To display status data, you must select one of the options and complete the field by typing in the requested information and pressing Enter. For example:

1. Type **1** in the entry field of Select an Option and type **TA1P523A** in the Id field. Figure 7 shows these fields.
2. Type **3** in the Select a Feature field to select the SNBU option.
3. Press Enter.

AON displays the Display Status Data panel, as shown in Figure 8.

```

EZLK7210          Display Status Data          CNM01

Select one of the following. Then press enter.
1=Delete

- ID= SUTA1P523A      , TYPE= SNBU      , STATUS= ISNBU
  LAST UPDATE BY OPERATOR AUTNET1
  LAST THRESHOLD EXCEEDED - *****
  OPERATOR NOTIFIED: *
  LAST STATUS CHANGE DATE= 03/31/97 , TIME= 11:03 , OPER= AUTNET1
  LAST MONITORED DATE= 03/31/97 , TIME= 11:03
  HIER 1= TA1N500 , 2= TA1L5023 , 3= TA1P523A , 4=      , 5=
  ALTPORT= ***** , POOL= **** , SPEED = FULL , LEVEL=

Command ==>
F1=Help      F2=Main Menu  F3=Return          F5=Refresh  F6=Roll
F7=Backward  F8=Forward   F12=Cancel

```

Figure 8. Displaying the Status Data

4. Type **1** in the entry field to delete the record.

If you issue the DSPSNBU command from the command line with one of the parameters, AON bypasses the Display Status Data panel that Figure 7 on page 81 shows and displays the Display Status Data panel, as shown in Figure 8 on page 81.

Restrictions

This command operates in full-screen mode only.

LISTSNBU

Syntax

LISTSNBU

»—LISTSNBU—«

Purpose of Command

The LISTSNBU command displays the SNBU PU and modem pool entries defined in the AON/SNA control file. It also displays the status of these entries.

The LISTSNBU command displays the AON SNBU Resource List panel.

Usage Notes

You can issue other AON/SNA SNBU commands by pressing F4 when the cursor is on a SNBU resource. The first two lines of the first panel display SNBU defaults and the PUs so that you can access global SNBU automation defaults from LISTSNBU by pressing F4.

The status codes for a PU are:

INIT Initial status for speed selection (operation begun)

ISNBU

Initiated switched network backup (operation begun)

Note: If the LISTSNBU status remains on the panel for an extended time, browse the NETLOG to determine why SNBU failed.

BOTH Transmit and receive lines in backup speed

LOCAL

Transmit side of line in backup speed

RCV Receive side of line in backup speed

SNBU Successful dial connection (operation completed)

The status codes for an alternate port are:

BUSY This alternate port is in use.

DOWN

This alternate port is inoperative.

Restrictions

This command operates in full-screen mode only.

LUDRPOOL

Syntax

LUDRPOOL

►►—LUDRPOOL *ncp interval threshold*—◄◄

Purpose of Command

The LUDRPOOL command assists in X.25 network management. When the NCP is built, the system programmer sets aside a pool of control blocks to be used by dynamically-added LUs. Thus, the name of the command comes from the LU Dynamic Reconfiguration POOL. These pools are used when users dial in to the network. The LUDRPOOL command interrogates the count of available LUs in the LUDRPOOL for PU type 2 for an NCP.

The program returns the message

LUDRPOOL=*xx*

The LUDRPOOL command is also issued at regular intervals if an interval is specified on the command. The LUDRPOOL command is based on the NCP control blocks structure and works with different versions of NCP. The supported versions are NCP V4R2 through V7R6. You must have vital product data (VPD) turned on to issue the LUDRPOOL command.

The LUDRPOOL command displays the SNA Automation: X25 LUDRPOOL panel.

Operand Descriptions

ncp

Specifies the name of the NCP.

interval

Specifies a number between 1 and 59.

threshold

Adds, changes, or deletes the threshold settings. Specify a number between 0 and 999.

Usage Notes

If just *ncp* is passed, AON/SNA displays the following message:

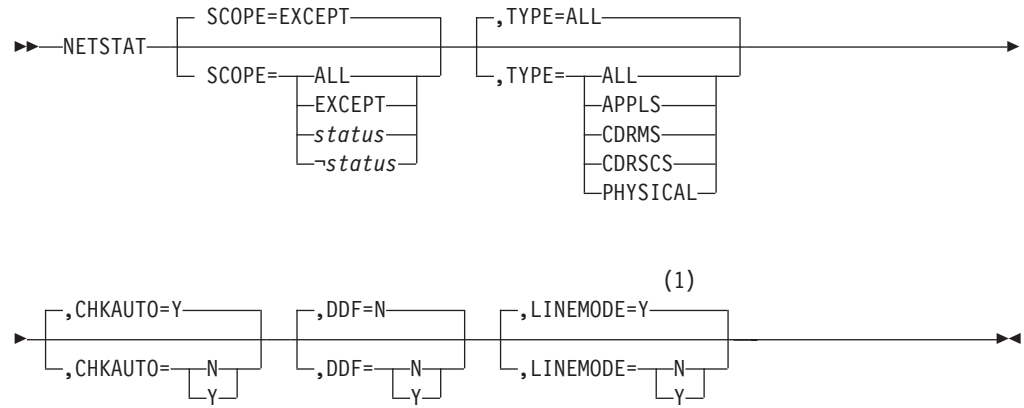
FKV651I LUDRPOOL FOR NCP *ncp=number*

If *ncp* and *interval* are specified, LUDRPOOL is set to issue at the specified intervals. Optionally, a *threshold* value can be specified to define the minimum number as expected.

NETSTAT

Syntax

NETSTAT



Notes:

- 1 AON automatically sets this parameter to YES for automation operators. The default for NetView operators is NO.

Purpose of Command

The NETSTAT command can be used to level-set operators on network status after an IPL or unscheduled outage of VTAM, NetView, or AON. It shows you a display of all the VTAM resources, or all the resources that are currently not active.

Attention: The NETSTAT command is not intended for use as an active monitor. For very large networks, running time for this command can be lengthy.

No recovery is scheduled on the basis of output from this facility.

The NETSTAT command displays the SNA Automation: NetStat panel.

Operand Descriptions

SCOPE

Defines the extent to which the resource is displayed.

EXCEPT

Displays all resources that are currently not in an ACTIVE status. This is the default.

ALL

Displays all resources regardless of status.

status

Displays only resources with this status.

~status

Display all resources that do not have this status.

TYPE

Defines which resources to display.

ALL

Displays all resources. This is the default.

APPLS

Displays only the applications.

CDRMS

Displays only the cross-domain resource managers.

CDRSCS

Displays only the cross-domain resources.

PHYSICAL

Displays only the physical components. This includes NCPs, LUs, PUs, and switched nodes.

CHKAUTO

Determines if the resources displayed should be checked against the recovery statements in the control file. If recovery is off, the resource is not displayed. The default is Y.

DDF

Determines if the information should be passed to the Dynamic Display Facility (DDF). The default is N. Operators should not use this option unless the DDF statuses need to be reset. AON uses this command to initialize DDF statuses.

LINEMODE

When LINEMODE=Y, the output from the command can be sent to the NCCF command line or to a program routine. When this command is issued by an automated operator task or from NetView-NetView sessions, AON automatically sets this parameter to YES for operator tasks. The default for operators is NO.

Usage Notes

Consider the following when using the NETSTAT command:

- This command list is run when DDF initializes if DDFREFRESH=YES is specified in the control file as part of the ENVIRON SETUP statement.
- Refer to FKVESYNC in the *Tivoli NetView for z/OS Automated Operations Network Customization Guide* to provide automation of any resource failures detected by NETSTAT.

QRYSNBU

Syntax

QRYSNBU

▶▶—QRYSNBU—◀◀

Purpose of Command

The QRYSNBU command queries the status file entries for switched network backup automation (SNBU automation) resources. The QRYSNBU command displays the status of all SNBU automation entries in the status file.

The QRYSNBU command displays the SNBU PU Status Display panel.

Usage Notes

The status codes for a PU are:

INIT Initial status for speed selection (operation begun)

ISNBU

Initiated switched network backup (operation begun)

Note: If the ISNBU status remains on the panel for an extended time, browse the NETLOG to determine why SNBU failed.

BOTH Transmit and receive lines in backup speed

LOCAL

Transmit side of line in backup speed

RCV Receive side of line in backup speed

SNBU Successful dial connection (operation completed)

The status codes for an alternate port are:

BUSY This alternate port is in use.

DOWN

This alternate port is inoperative.

Restrictions

This command operates in full-screen mode only.

SETSNBU

Syntax

SETSNBU

```
SETSNBU [ pname ]
```

Purpose of Command

The SETSNBU command enables you to dynamically add, delete, or alter the SNBU entries in the control file for the current execution of NetView.

Operand Descriptions

pname

The name of the physical unit (PU).

Usage Notes

- If *pname* is specified, the CLIST will retrieve the current information about that PU from the control file and display that information. If there is no current information, or if a PU name is not specified, the input panel will contain only the default values for the SNBU operands.
- Entering new information will update the table; F12 will delete a current entry. The operator is notified if a value entered is invalid.
- Following are descriptions of the input fields:

Resource name

The name of the PU to be switched.

Allow automatic speed switch

Set to lower speed for E/T errors.

Allow automatic SNBU backup

Allow dial backup for permanent errors.

Allow automatic reconnection

Automatically restore modem connection.

SNBU when Critical Threshold Exceeded

SNBU when the critical threshold is exceeded.

SNBU When first MONIT Interval Exceeded

SNBU when the first MONIT interval is exceeded.

Modem Link Level

Tailed circuits, modem link level.

Modem Pool Name

The four character modem pool name.

Use Alternate Port Only

Always use the alternate port from this pool.

SNBU Phone Numbers

The first and second phone numbers. Valid characters include: 'F' | 'P' for pause, '-', '(', and ')'.
'

Fanout PUs

A scrollable window that displays fanout PUs, up to 12 per window, attached through a modem to the above PU name.

SNAHD

Syntax

SNAHD



Purpose of Command

The SNAHD command provides access to a full-screen help desk to guide you through problems with SNA resources and NetView Access Services user IDs. Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for more information.

Operand Descriptions

resname

The name of the SNA resource or the NetView Access Services user ID you want to investigate.

option

option can do one of the following:

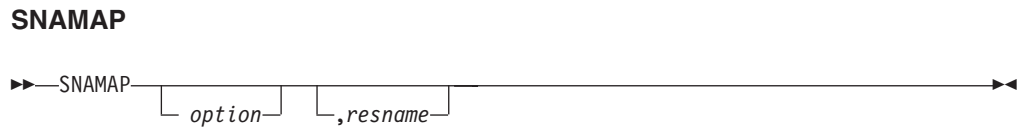
- 1 (Recycle resource)
- 2 (Problem determination)
- 3 (NetView Access Services *userid*)

Restrictions

This command operates in full-screen mode only.

SNAMAP

Syntax



Purpose of Command

The SNAMAP command provides the ability to map a resource to its lower resources.

Operand Descriptions

option

One of the following:

- 1 (MAJNODES)
- 2 (APPLS)
- 3 (CDRMS)
- 4 (CDRSCS)
- 5 (LINKSTA)
- 6 (CLSTRS)
- 7 (TERMS)
- 8 (user-provided *resname*)

resname

The name of the SNA resource. This is used with Option 8.

Restrictions

This command operates in full-screen mode only.

SNAPULST

Syntax

SNAPULST



Purpose of Command

The SNAPULST command provides a list display of the LUs and CPs that belong to a PU.

Operand Descriptions

resname

The name of the PU, LU, or CP. If the name is an LU or CP, a pop-up is displayed with the option to either show the PU list on the PU, or enter the SNA Help Desk on the LU or CP. If *resname* is not specified, a panel is displayed to prompt you to enter the name of the resource.

Restrictions

This command operates in full-screen mode only.

SNBU

Syntax

SNBU

▶▶—SNBU—◀◀

Purpose of Command

The SNBU command displays the SNBU Main Menu where you can select SNBU-related commands (for example, CHGSNBU, SETSNBU).

VTAMCMD

Syntax

VTAMCMD

▶▶—VTAMCMD—◀◀

Purpose of Command

The VTAMCMD command provides a full-screen interface to issue common commands.

Usage Notes

When you enter the VTAMCMD command, a full-screen panel is displayed showing the last commands entered from this panel. Move your cursor to the command you want to issue and press Enter, or you can type a new command to be issued.

Restrictions

This command operates in full-screen mode only.

VTAMOPT

Syntax

VTAMOPT

▶▶—VTAMOPT—◀◀

Purpose of Command

The VTAMOPT command provides a full-screen interface to display and change VTAM options.

Usage Notes

When you enter the VTAMOPT command, a full-screen panel is displayed with the current VTAM option settings. From the panel, you can type over the existing entry and press Enter to make needed changes.

Restrictions

This command operates in full-screen mode only.

X25MONIT

Syntax

X25MONIT

▶▶—X25MONIT—▶▶

Purpose of Command

The X25MONIT command displays all X.25 resources defined in the control file. From the list panel, you can add, change, and delete X.25 resources.

Chapter 5. AON/TCP Commands

This section contains all AON/TCP commands, which are listed in alphabetical order.

AONTCP

Syntax

▶▶—AONTCP—◀◀

Purpose of Command

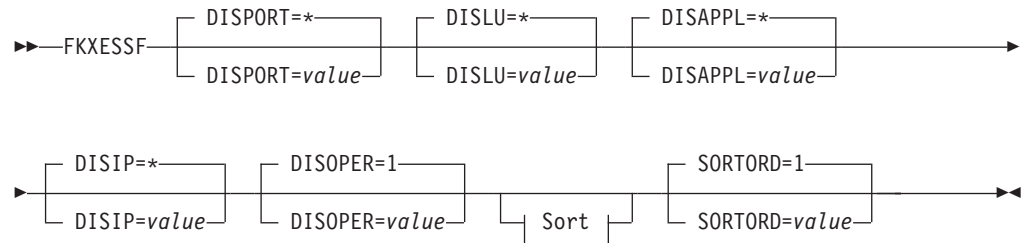
The AONTCP command displays the TCP/IP Automation: Commands Menu panel, which enables you to access the functions of AON/TCP.

Usage Notes

You can issue the AONTCP command from any command line within AON or any of the components of AON.

FKXESSF

Syntax



Sort:

(1)
|----- SORTPORT=value----- SORTLU=value----- SORTAPPL=value----- SORTIP=value-----|

Notes:

- 1 These parameters are optional. However, if one parameter is specified, then all must be specified.

Purpose of Command

The FKXESSF command invokes the session status filters panel or sets them directly when the parameters are passed.

Operand Descriptions

DISPORT

Data to be displayed in the Port field. An asterisk (*) is the default.

DISLU

Data to be displayed in the Logical Unit field. An asterisk (*) is the default.

DISAPPL

Data to be displayed in the Appl field. An asterisk (*) is the default.

DISIP

Data to be displayed in the IP Address field. An asterisk (*) is the default.

DISOPER

The logical operator to be used for the filtering criteria.

- 1 The OR operator. This is the default.
- 2 The AND operator.

SORTPORT

The sort order to be used for the Port field. Zero (0) indicates that a sort order for this field is not needed.

SORTLU

The sort order to be used for the Logical Unit field. Zero (0) indicates that a sort order for this field is not needed.

SORTAPPL

The sort order to be used for the Appl field. Zero (0) indicates that a sort order for this field is not needed.

SORTIP

The sort order to be used for the IP Address field. Zero (0) indicates that a sort order for this field is not needed.

SORTORD

The sort order to be used.

- 1 Sort is to be completed in ascending order. This is the default.
- 2 Sort is to be completed in descending order.

Note: All of the sort keywords must be specified when invoked from the command line or another command list. The values of the sort keywords must be sequential, beginning with 1.

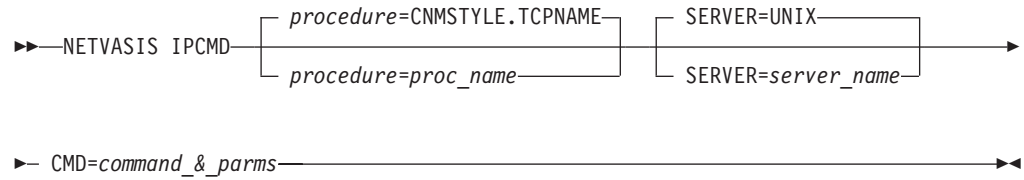
Usage Notes

You can use the panel interface by issuing FKXESSF without parameters from any NetView command line. Or, to set the filter values directly without using the panel interface, you can issue FKXESSF with parameters from any NetView command line or from within a command procedure.

IPCMD

Syntax

IPCMD



Purpose of Command

The IPCMD command provides users with a generic API to execute any IP command for the UNIX or TSO environments. The command is issued from NetView with correlated responses returned to the user.

Operand Descriptions

procedure

Specifies the MVS procedure. Valid values are PROC and STACK.

proc_name

Specifies the name of the MVS procedure. The default is defined in sample CNMSTYLE with the common global variable TCPNAME. The value for TCPNAME is the *proc_name* of the TCP/IP stack. Refer to CNMSTYLE for more information about setting this value.

SERVER

Specifies the server name, either TSO or UNIX. The default is UNIX.

CMD

Specifies the TCP/IP command and parameters.

Usage Notes

- Specify NETVASIS when entering UNIX commands or commands that are case sensitive.
- Parameters are not positional, but enter `CMD=command_&_parms` last to enable proper parameter parsing.
- The default stack and server can be changed by modifying CNMSTYLE. Refer to the comments in CNMSTYLE for more information.

Examples

To use the UNIX server to ping the host named quigley, using the default TCP/IP for MVS stack name, enter the following command:

```
NETVASIS IPCMD CMD=oping QUIGLEY
```

The response is returned to the invoker.

To use the first available TSO server to ping host name quigley, enter the following command:

```
IPCMD SERVER=TSO CMD=PING QUIGLEY
```

The response is returned to the invoker.

IPMAN

Syntax

▶▶—IPMAN—◀◀

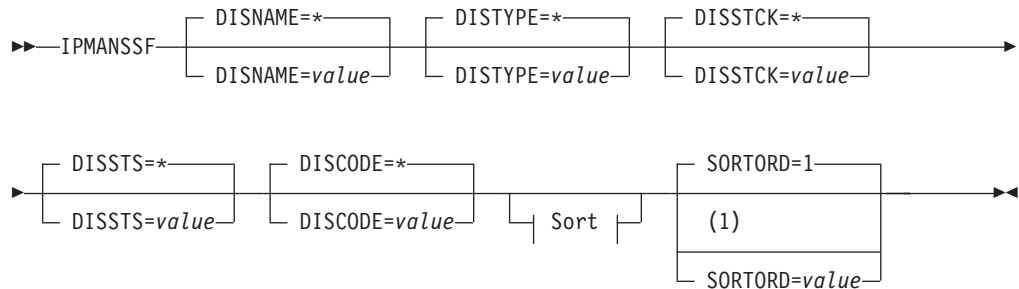
Purpose of Command

The IPMAN command displays the IP Resource Manager main panel, which is used to manage IP resources defined in control files. From this panel you can start and stop monitoring, add, change, or delete an instore control file policy of a given resource, and display resources. The ADD function starts active monitoring and the CHANGE function restarts it. The commands function enables you to link to other IP functions.

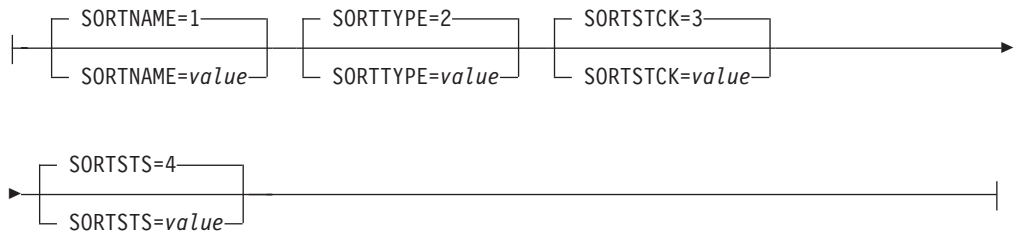
Refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide* for details about using the IP Resource Manager main panel and its associated panels.

IPMANSSF

Syntax



Sort:



Notes:

- 1 These parameters are optional. However, if one parameter is specified, then all must be specified.

Purpose of Command

The IPMANSSF command enables an operator to set filters for the SNMP Resource Management Panel for the current operator. The filter and sort values are saved in task global variables. IPMANSSF invokes the SNMP Resource Management Panel when no parameters are specified.

Operand Descriptions

DISNAME

Sets the search criteria based on the resource name. If the text is followed by an *, then all resources beginning with characters preceding the * will be included. Without the * only resources matching this field exactly will be returned. Coding just the * displays all. The * is the default.

DISTYPE

Sets the search criteria based on the resource type. If the text is followed by an *, then all resources beginning with characters preceding the * will be included. Without the * only resources matching this field exactly will be returned. Coding just the * displays all. The * is the default.

DISSTCK

Sets the search criteria based on the TCP/IP stack name. If the text is followed by an *, then all resources beginning with characters preceding the * will be

included. Without the * only resources matching this field exactly will be returned. Coding just the * displays all. The * is the default.

DISSTS

Sets the search criteria based on the resource name. If the text is followed by an *, then all resources beginning with characters preceding the * will be included. Without the * only resources matching this field exactly will be returned. Coding just the * displays all. The * is the default.

DISCODE

- 1 – indicates an "OR" relationship should be used when searching the DISNAME, DISTYPR, DISSTCK and DISSTS fields. This is the default.
- 2 – indicates an "AND" relationship should be used when searching the DISNAME, DISTYPR, DISSTCK and DISSTS fields.

SORTNAME

The sort order to be used for the SORTNAME field. One (1) is the default.

SORTTYPE

The sort order to be used for the SORTTYPE field. Two (2) is the default.

SORTSTCK

The sort order to be used for the TCP/IP Stack Name field. Three (3) is the default.

SORTSTS

The sort order to be used for the Resource Status field. Four (4) is the default.

SORTORD

The sort order to be used.

- 1 Sort is to be completed in ascending order. This is the default.
- 2 Sort is to be completed in descending order.

Note: All of the sort keywords must be specified when invoked from the command line or another command list. The values of the sort keywords must be sequential beginning with 1.

Usage Notes

You can issue the IPMANSSF command from any command line within AON or from any of the components of AON.

IPTRACE

Syntax

▶▶ IPTRACE [*service_point*] ▶▶

Purpose of Command

The IPTRACE command enables you to perform diagnostic traces to help resolve TCP/IP problems. There are two types of traces available, component trace (CTRACE) and packet trace (PKTTRACE). For more information about using the IP Trace functions, refer to *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

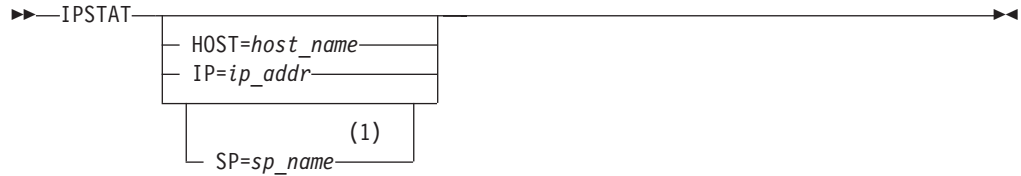
service_point

Specifies the service points for which TCP/IP services will be traced. These are defined in the DSIPARM member FKXCFG01 configuration file.

IPSTAT

Syntax

IPSTAT



Notes:

- 1 If you specify the SP operand, you must first specify the HOST or IP operand.

Purpose of Command

The IPSTAT command provides you with Session Management and debugging capabilities for Telnet and FTP sessions connecting into your TCP/IP for MVS. A series of full screen panels are provided to assist you in managing these sessions.

Operand Descriptions

host_name

Specifies the IP host name of the workstation for which you are requesting session status information.

ip_name

Specifies the IP address of the workstation for which you are requesting session status information.

sp_name

Specifies which MVS TCP/IP stack to use for the session status request.

Usage Notes

- You cannot specify both the HOST and IP operands in the same IPSTAT command.
- If you specify the SP operand, you must first specify the HOST or IP operand.

Examples

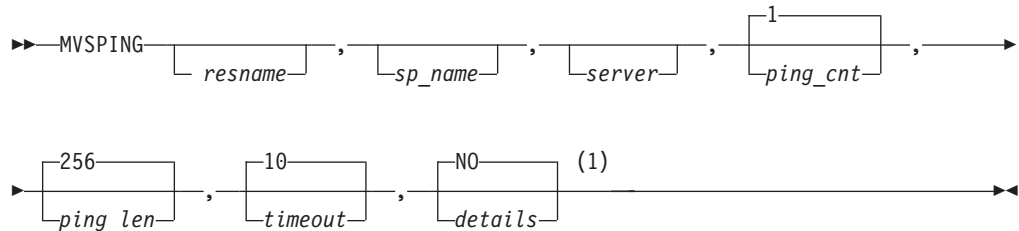
This example displays session status information for host RAL1 using the TCP/IP for MVS service point MYMVS.

```
IPSTAT HOST=RAL1 SP=MYMVS
```

MVSPING

Syntax

MVSPING



Notes:

- 1 If you do not specify a positional parameter, indicate the absence of the parameter by specifying a comma in its place.

Purpose of Command

The MVSPING command enables you to ping a TCP/IP resource using TCP/IP for 390. A full-screen panel is displayed to enable you to override the defaults used by AON. Otherwise, the ping response will be sent to your NCCF screen or command list, depending on how MVSPING was invoked.

Operand Descriptions

resname

Specifies the IP host name or address to be pinged.

sp_name

Specifies which MVS TCP/IP stack to use for the ping request.

server

Specifies the name of the TSO or UNIX server. The default is the next available server.

ping_cnt

Specifies the number of pings to issue. The default is 1.

ping_len

Specifies the size of each packet. The default is 256.

timeout

Specifies the ping timeout in seconds. The default is 10 seconds.

details

Specifies to return full ping response. The default is NO.

Usage Notes

- If you specify a stack name, you must first specify a *resname*.
- If only the *resname* and *sp_name* parameters are specified, you will be directed to panel FKXK2100.

- When entering MVSPING from the command line, if all parameters are properly specified, you will receive a response to your request. If all parameters are not provided, you will receive the MVSPING panel. When that occurs, host names greater than 64 bytes will be truncated.

Examples

To ping a host named RAL1, enter the following command:

```
MVSPING RAL1
```

A full-screen panel is displayed, listing the valid stacks. When a stack has been selected, the MVSPING command can be issued.

To ping a host named RAL1 with the NMPIPL25 stack, enter the following command:

```
MVSPING RAL1,NMPIPL25
```

Default values for all ping parameters are used, while the server is based on AON policy definitions. The response is returned to the invoker.

To send a ping request to MVS Stack TCP34, using the UNIX server and a ping length of 64, enter the following command:

```
MVSPING RAL1,TCP34,UNIX,,64,,N
```

Only a ping response will be returned to the invoker.

NV6KCMD

Syntax

▶▶ NV6KCMD [*service_point*] ▶▶

Purpose of Command

The NV6KCMD command displays the TCP/IP Automation: Issue Command to Service Point panel, which enables you to send any AIX command to the NetView for AIX service point.

Operand Descriptions

service_point

Indicates the service point to which you want to send the command.

Usage Notes

To issue the NV6KCMD command, do either of the following:

- Type **NV6KCMD** on any command line.
- Type **AON 4.1.2** on any command line.

NV6KLIST

Syntax

▶▶—NV6KLIST—◀◀

Purpose of Command

The NV6KLIST command displays the TCP/IP Automation: Resource List panel, which displays information about a particular resource.

Usage Notes

To issue the NV6KLIST command, do either of the following:

- Type **NV6KLIST** on any command line.
- Type **AON 4.1.5** on any command line.

NV6KPERF

Syntax

▶▶ NV6KPERF type ▶▶

Purpose of Command

The NV6KPERF command displays the TCP/IP Automation: Performance Thresholds panel, which enables you to manage the performance threshold values for a resource.

Operand Descriptions

type

Indicates the alias name of the resource you want to manage or the resource type of the resource defined in the control file that you want to update.

Usage Notes

To issue the NV6KPERF command, do either of the following:

- Type **NV6KPERF** on any command line.
- Type **AON 4.1.4** on any command line.

NV6KPING

Syntax

```
▶▶ NV6KPING [ node_name ] [ service_point_name ] ▶▶
```

Purpose of Command

The NV6KPING command displays the TCP/IP Automation: Ping a Service Point panel, which enables you to ping any NetView for AIX managed TCP/IP resource.

Operand Descriptions

node_name

Indicates the node name to which to send the ping.

service_point_name

Indicates the service point name to which to send the ping.

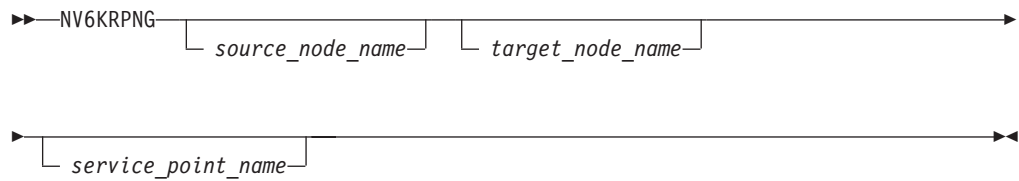
Usage Notes

To issue the NV6KPING command, do either of the following:

- Type **NV6KPING** on any command line.
- Type **AON 4.1.1** on any command line.

NV6KRPNG

Syntax



Purpose of Command

The NV6KRPNG command displays the TCP/IP Automation: Remote Ping between two Nodes panel, which enables you to send a ping command from one remote node to another and have the results sent to your terminal.

Operand Descriptions

SourceNodeName

Indicates the name of the node from which you want to send the ping.

TargetNodeName

Indicates the name of the node to which to send the ping.

ServicePointName

Indicates the name of the NetView for AIX service point to which to issue the ping.

Usage Notes

To issue the NV6KRPNG command, do either of the following:

- Type **NV6KRPNG** on any command line.
- Type **AON 4.1.3** on any command line.

NV6KVIEW

Syntax

```
▶▶ NV6KVIEW [ option ] [ resname ] ▶▶
```

Purpose of Command

The NV6KVIEW command displays the TCP/IP Automation: AutoView panel, which enables you to view summary data for a resource and control automation.

Operand Descriptions

option

Indicates the type of resource for which you want to display automation information. Valid values for option are TCPIP, IP390, or NV6K.

resname

Indicates the name of the AON/TCP resource for which you want to display automation information.

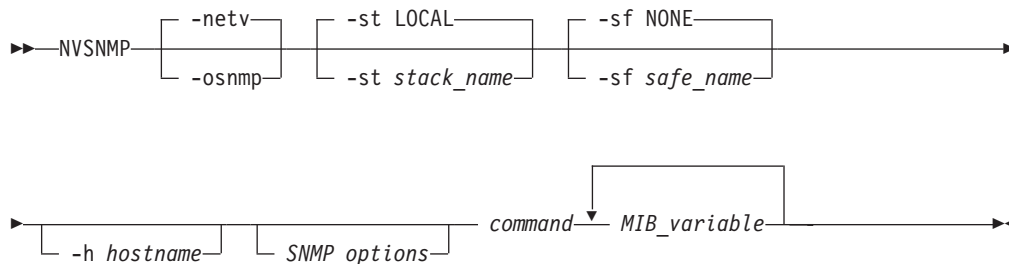
Usage Notes

To issue the NV6KVIEW command, type **NV6KVIEW** on any command line.

NVSNMP

Syntax

NVSNMP



Purpose of Command

The NVSNMP command enables you to manage devices through SNMP with either NetView SNMP or CS/390 osnmp. The NVSNMP command can be issued from the NetView command line or from a REXX procedure. Depending on how NVSNMP is issued, requests are returned to the screen or to the issuing procedure in safe FKXTCMD.

You can use NetView or CS/390 osnmp from the command line depending on the option specified. If you are running AON, entering NVSNMP without any parameters brings up the AON SNMP panels, which use NetView SNMP only.

Notes:

1. The interface is case sensitive. Use NETVASIS to ensure that lowercase fields are not converted to uppercase. Use Address NETVASIS in the issuing procedure.
2. MIB variable names, where they form part of an SNMP request, are expected to be in ASN.1 notation by the NetView SNMP command.

Operand Descriptions

-osnmp

Specifies to issue the request to CS/390 osnmp.

-netv

Specifies that the request is issued using NetView SNMP services. This is the default.

-st *stack_name*

Specifies the TCP/IP service point to which the command is sent. The default is LOCAL.

-sf *safe_name*

Specifies an input safe which contains MIB variable names to be passed to SNMP. The default is NONE.

-h *hostname*

Specifies the host name or TCP/IP address of the host from where the SNMP MIBs are to be pulled.

SNMP_options

Specifies valid SNMP options.

command

Specifies the name of the command. Valid names are as follows:

- GET | GETBULK
- GETNEXT
- WALK | BULKWALK
- SET

MIB_variable

Specifies the variable name to be acted upon by the command. If a safe name is used, the MIBs should not be listed on the command line.

Usage Notes

Consider the following when using the NVSNMP command:

- If you want to provide command security for the SNMP commands, refer to the *Tivoli NetView for z/OS Security Reference* for information about setting up SNMP command authorization definitions.
- NetView SNMP requires a community name as part of the input. If the community name (-c parameter) is not supplied when the -netv option is specified, the default is public.

Examples

NETVASIS NVSNMP GET sysDescr.0

Simple get to LOCAL of sysDescr.0. Response is returned to the user.

NETVASIS NVSNMP -st NMPIPL10 -sf TESTSAFE -d ALL -t 10 GETBULK

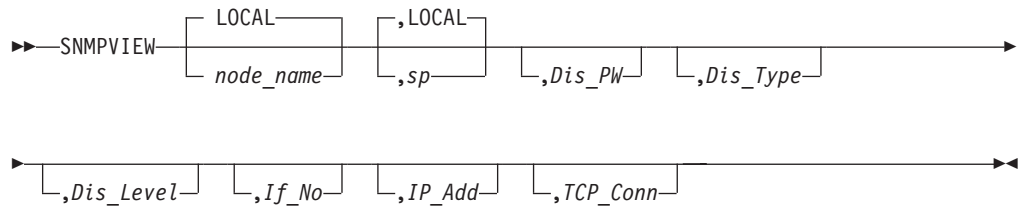
Getbulk to TCP/IP stack NMPIPL10 using safe TESTSAFE. MIBs is read from host NMPIPL10. Use -d to request all debugging information and -t for a timeout of 10 seconds.

NETVASIS NVSNMP -st LOCAL -sf NONE -h pkoch WALK system

Send a walk system command to host pkoch using the local system TCP/IP stack. No safe is specified which causes the response to be returned to the user.

SNMPVIEW

Syntax



IBM-Defined Synonyms

Command or Operand	Synonym
SNMPVIEW	SNMPV
SNMPVIEW	SV

Purpose of Command

The SNMPVIEW command can be used to collect logically grouped portions of MIB information about a resource and return this information in REXX variables to the calling program. For more information about using the SNMP views function, refer to the *Tivoli NetView for z/OS Automated Operations Network User's Guide*.

Operand Descriptions

node_name

Specifies the host name or TCP/IP address of the host from where the information is to be gathered. The default is LOCAL, which will use the local IP stack.

sp Specifies the MVS TCP/IP service point to which the command is sent. The default is LOCAL, which will use the local IP stack.

Dis_PW

The community name used for the MIB get requests.

Dis_Type

The type of resource. Valid values are:

- MVS** MVS TCP/IP stack
- IP** Generic IP resource

Dis_Level

The level of screen display. Valid values are:

- SYS** System screen
- IF** Interface list
- IFD** Interface detail. If IFD is specified, a value for *If_No* is also required.
- CONN**

Connection list. If CONN is specified, a value for *IP_Add* is also required.

CONND

Connection detail. If CONND is specified, *Dis_Type*=MVS and a value for *TCP_Conn* are also required.

If_No

The interface number for data collection. This is required if a screen display level of IFD is specified.

IP_Add

The IP address for data collection. This is required if a screen display level of CONN is specified.

TCP_Conn

The TCP/IP connection. This is required if a screen display level of CONND is specified. The connection types consist of the following:

- Local IP address
- Local Port
- Remote IP address
- Remote Port

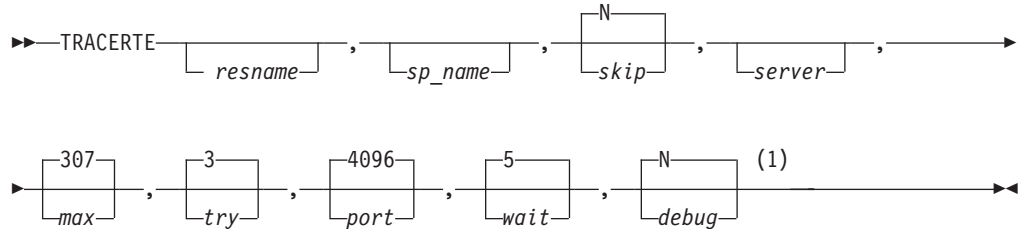
and are specified in the following format:

LocalIpAddr.LocalPort.RemIpAddr.RemPort.

TRACERTE

Syntax

TRACERTE



Notes:

- 1 If you do not specify a positional parameter, indicate the absence of the parameter by specifying a comma in its place.

Purpose of Command

The TRACERTE command can be used to issue the TCP/IP for MVS TRACERTE command against specified TCP/IP resources.

Operand Descriptions

resname

Specifies the IP host name or address of the resource.

sp_name

Specifies which MVS TCP/IP stack will issue the TRACERTE command.

skip

If Y is specified, the AON TRACERTE panel is bypassed and the TCP/IP TRACERTE command is issued with AON-supplied defaults. If L is specified, the response is returned as a MLWTO message. If N is specified, the AON TRACERTE panel is displayed. This is the default.

server

Specifies the name of the TSO or UNIX server.

max

Specifies the maximum TTL. The default is 30.

try Specifies the number of attempts. The default is 3.

port

Specifies the starting port number. The default is 4096.

wait

Specifies the time, in seconds, to wait for a response. The default is 5.

debug

Specifies whether or not to display *debug* messages. The default is N.

Usage Notes

- If you specify *sp_name*, you must first specify *resname*.

- When entering TRACERTE from the command line, if all parameters are properly specified, you will receive a response to your request. If all parameters are not provided, you will receive the TRACERTE panel. When that occurs, host names greater than 64 bytes will be truncated.

Examples

The following example will display the AON TRACERTE panel for resource quigley using the NMPIPL10 stack.

```
TRACERTE QUIGLEY,NMPIPL10
```

This example will issue a TRACERTE request to MVS stack defined in AON policy for the NMPIPL10 stack. The server is used from the AON policy definitions while overriding max=30, try=5, port=4096 and using the AON defaults for wait and debug.

```
TRACERTE QUIGLEY,NMPIPL10,L,,30,5,4096
```

Because SKIP=L was specified, the response is returned to the invoker.

Part 3. MultiSystem Manager Commands

Chapter 6. MultiSystem Manager Base

Commands.	127
DISPTOPO	128
DMCS (Distribution Manager Command Support)	130
FLCACTIP	133
INITTOPO	134
REMOV OBJs	135
RESTOPO	139
SETREMOV	140
SUSPTOPO	142

Chapter 7. MultiSystem Manager IP Commands 143

CRITRES	144
GETTOPO IPDETAIL.	145
GETTOPO IPONLY and IPRES	150

Chapter 8. MultiSystem Manager LNM

Commands.	157
GETTOPO LNMADP, LNMBRG and LNMCAU	158
GETTOPO LNMRES and LNMONLY	163
GETTOPO LNMSEG	170

Chapter 9. MultiSystem Manager NetFinity

Commands.	175
DISPMON	176
DISPPRC	177
DISPPRO.	178
DISPTHRR.	179
GETTOPO NFGRP	180
GETTOPO NFOONLY	186
GETTOPO NFRES.	192
GETTOPO NFWKST	199

Chapter 10. MultiSystem Manager Open

Commands.	205
GETTOPO HOSTONLY	206
GETTOPO OPENRES.	211

Chapter 11. MultiSystem Manager TMR

Commands.	217
GETTOPO TMEONLY	218
GETTOPO TMERES	223

Chapter 6. MultiSystem Manager Base Commands

This section contains all base MultiSystem Manager commands, which are listed in alphabetical order.

DISPTOPO

Syntax

▶—DISPTOPO—▶

Purpose of Command

The DISPTOPO command displays information about MultiSystem Manager. The information is displayed in your host NetView window.

Usage Notes

You can issue the DISPTOPO command from a NetView command line or a NetView command procedure. You cannot issue it from the MultiSystem Manager command facility.

Command Responses

Current Status

The current status of MultiSystem Manager.

ENABLED

MultiSystem Manager is able to process GETTOPO commands.

INITIALIZATION_FAILED

MultiSystem Manager is not able to process GETTOPO commands. An INITTOPO command was issued, but initialization failed because of a syntax error or incorrect information in the initialization file.

INITIALIZING

An INITTOPO command was issued and MultiSystem Manager is processing the initialization file. After the initialization file is processed, the status is changed to *INITIALIZATION_FAILED* or *ENABLED*.

NEVER_INITIALIZED

MultiSystem Manager has not been initialized and is not able to process GETTOPO commands.

SUSPENDED

A SUSPTOPO command was issued and the processing of GETTOPO commands is suspended. Use the RESTOPO command to resume processing of topology requests.

Default Autotask Name

| The name of the default autotask that MultiSystem Manager is using to issue
| RUNCMDs. This name is specified on the (MSM)function.autotask.MSMdefault
| statement in CNMSTYLE.

Default Network View Description

| The description for the default network view. This description is specified on
| the *network_view_annotation* portion of the
| (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

Default Network View Name

| The name of the MultiSystem Manager default network-level view. This name
| is specified on the *network_view_name* portion of the
| (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

Exception View Member Name

The name of the exception view file that is specified on the (MSM)COMMON.FLC_EXCEPTION_VIEW_FILE statement in CNMSTYLE.

Initialization Member Name

The name of the current MultiSystem Manager initialization file. This is the name of the initialization member specified on the INITTOPO command. If the initialization member is not specified on the INITTOPO command, the default initialization member FLCAINP is used.

NetView Version

The version and release level of the NetView program on your system.

RODM Application ID

The user application ID used by MultiSystem Manager to access RODM. This ID is specified on the COMMON.FLC_RODMAPPL statement in CNMSTYLE. The default RODM application ID is the NetView domain, concatenated with the letters MSM.

RODM Interval

The amount of time, in seconds, between retries of a RODM request that has failed because RODM is checkpointing. This value is specified on the (MSM)COMMON.FLC_RODMINT statement in CNMSTYLE.

RODM Name

The name of the RODM that MultiSystem Manager is using to store topology and status information. This name is specified on the COMMON.FLC_RODMNAME statement in CNMSTYLE.

RODM Retry Count

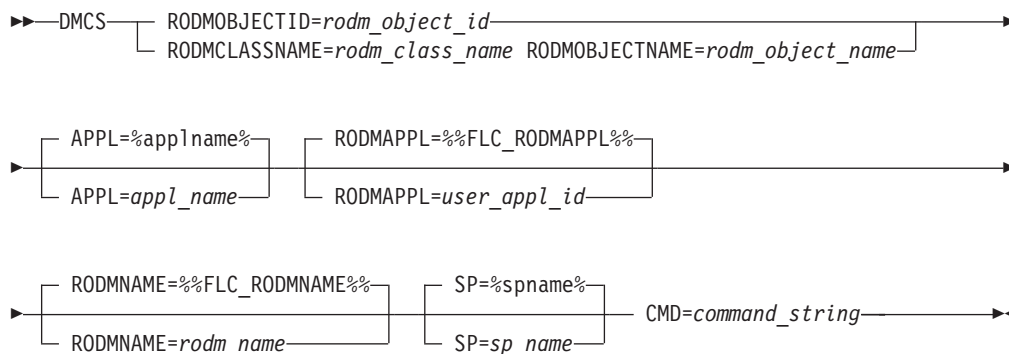
The number of times MultiSystem Manager retries a RODM request that has failed because RODM is checkpointing. This number is specified on the (MSM)COMMON.FLC_RODMRETRY statement in CNMSTYLE.

RUNCMD Retry Count

The number of times MultiSystem Manager retries a RUNCMD after an initial failure that has a sense code of 0851 (session busy). This value is specified on the (MSM)COMMON.FLC_RUNCMDRETRY statement in CNMSTYLE.

DMCS (Distribution Manager Command Support)

Syntax



Purpose of Command

The Distributed Manager Command Support (DMCS) command is used to send commands to the network topology agent. The command processor can be part of your automated network operations or an extension to the command facility. The DMCS command obtains required information about the target object, then builds and sends the correct command.

Operand Descriptions

APPL

The name of the network management application that processes the command string. This name is the same as the application name used on the NetView RUNCMD APPL keyword.

For NetView for AIX networks, this is the name of the service point application registered to the AIX[®] NetView Service Point. The service point application name is defined when the options for host connection daemons are set up in NetView for AIX. You can get the service point application name from your NetView for AIX system programmer.

CMD

The command string that is sent to the agent. The string can contain substitution variables for attribute names. A substitution variable can be any attribute on the RODM object specified in the RODMxxx keyword or a NetView global variable. Enclose RODM object substitution variables in percent signs (%); for example: %1.2.840.6203.7.0%. Enclose NetView global variables in pairs of percent signs (%%); for example: %%FLC_DEF_AUTOTASK%%.

MultiSystem Manager obtains the value of RODM object variables from RODM and substitutes them in the string. Therefore, all attributes specified in the command string must exist on the specified object.

MultiSystem Manager obtains the value of NetView global variables from NetView and substitutes them in the string. If a specified variable is null in NetView, MultiSystem Manager does not make the substitution, but still issues the command.

The total length of the command, including the values substituted for the other keywords, cannot exceed the maximum length restriction of 240 characters for the RUNCMD.

RODMAPPL

The user application ID used to access the RODM that contains the topology and status data. The ID is case-sensitive and must be entered exactly as created with RACF.

| The application ID for MultiSystem Manager is set during NetView
| initialization. Specify this keyword only if you are overriding the ID used by
| MultiSystem Manager, or if you have not coded the
| COMMON.FLC_RODMAPPL statement in CNMSTYLE.

You can use the DISPTOPO command to determine the application ID for the RODM being used by MultiSystem Manager.

RODMCLASSNAME

The RODM class name of the object that is the target of the command.

RODMNAME

| The name of the RODM where topology and status information is stored. This
| is normally the RODM used by MultiSystem Manager. For MultiSystem
| Manager this is the 1- to 8-character name that was specified on the
| COMMON.FLC_RODMNAME statement in CNMSTYLE. Specify this keyword
| only if you want to override the RODM name used by MultiSystem Manager,
| or if the statement in CNMSTYLE was not coded.

| You can use the DISPTOPO command to determine the name of the RODM
| being used by MultiSystem Manager.

RODMOBJECTID

The RODM object ID (ObjectID) of the object. The object ID is 16 characters that represent an 8-byte hexadecimal field; for example: 00010027EC211161.

You can enter either the RODM object ID or the RODM class name and object name combination. MultiSystem Manager uses the object ID in the commands that it builds during command facility processing, because it can determine the ID based on the resource you selected in the view. If you are entering this command from the command line or an automated procedure, use the RODM class name and object name combination.

RODMOBJECTNAME

The RODM object name of the object that is the target of the command.

SP

The name of the service point that processes this command.

Usage Notes

- The CMD parameter is positional and must be coded last.
- Do not code the APPL or RODMAPPL keywords for MultiSystem Manager command sets.
- If MultiSystem Manager determines that a RMTCMD must be used, based on the object that was selected, an asterisk (*) is used in the RMTCMD OPERID parameter. When it processes the RMTCMD, the local NetView looks for a task on the remote NetView that is owned by the operator and attempts to issue the command under that task. If this fails, NetView uses the operator's ID as the task name.

Examples

To issue a find route command, your DMCS command may look like this, depending upon the MultiSystem Manager feature:

```
DMCS RODMOBJECTID=40001A18B006 CMD=NFRSYSCL /GETGRP /ALL
```

To issue a command using NetView global variables, your command may look like this:

```
DMCS RODMOBJECTID=0000005A000000642 CMD=MY_CMD MY_KEYWORD=%%MY_GLOBAL_
VAR%%
```

In this example, MY_CMD is any command you might choose to issue, MY_KEYWORD is any keyword on that command, and %%MY_GLOBAL_VAR%% is the NetView global variable that you want substituted.

FLCACTIP

Syntax

▶▶—FLCACTIP— HOST=*hostname*— PORT=*port_number*— CMD=*command*—▶▶

Purpose of Command

The FLCACTIP command enables you to issue commands to the service point using the TCP/IP protocol.

Operand Descriptions

CMD

Specifies the command to be run.

This keyword must be coded last.

HOST

The host name or fully qualified host name of the target machine.

PORT

The TCP/IP Sockets port number associated with the command receiver.

Usage Notes

The following applies to the FLCACTIP command:

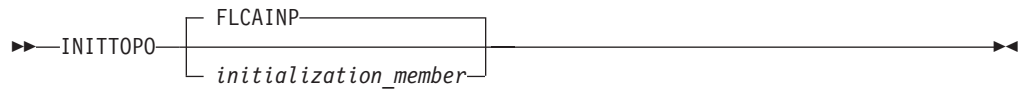
- This command is intended to test the connection of the host to the agent. Take care when running lengthy commands because your host will be unavailable until the command completes.
- The CMD keyword must be the final keyword.

Restrictions

Your service point must be able to communicate over TCP/IP in order to use the FLCACTIP command.

INITTOPO

Syntax



Purpose of Command

The INITTOPO command initializes MultiSystem Manager using the information in CNMSTYLE to set the system defaults. It also creates the topology manager objects in RODM. If the MultiSystem Manager initialization file contains GETTOPO statements, topology and status information is retrieved and stored in RODM for the resources specified by those statements. These resources can then be viewed and managed from the workstation.

If the initialization file does not contain GETTOPO statements, topology and status information is not initially retrieved and stored in RODM. MultiSystem Manager is enabled, but resource topology and status information is obtained only when GETTOPO commands are issued, or alerts are received.

Operand Descriptions

initialization_member

The name of the MultiSystem Manager initialization file.

Usage Notes

You can issue the INITTOPO command from a NetView command line or a NetView command procedure. You cannot issue it from the MultiSystem Manager command facility.

The following statements are included in CNMSTYLE:

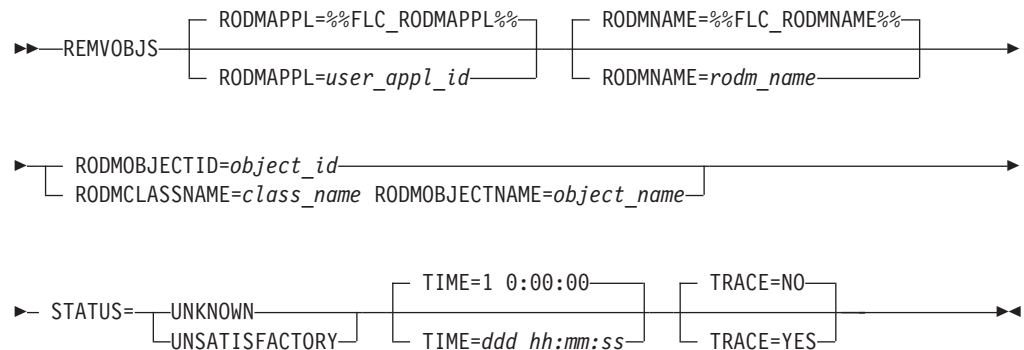
- (MSM)AUTOTASK.?MSMdefault.Console = *NONE*
- (MSM)AUTOTASK.?MSMdefault.InitCmd = INITTOPO

The first statement starts the autotask defined in the MSMdefault statement, which is also in CNMSTYLE. The second statement runs the INITTOPO command on that autotask. Comment out these statements if you do not want to run the INITTOPO command during NetView initialization.

INITTOPO overrides a MultiSystem Manager status of suspended set by the SUSPTOPO command.

REMOVBOJS

Syntax



Purpose of Command

The REMOVBOJS command removes objects from MultiSystem Manager views, provided they meet specific criteria. The criteria for removal is a combination of the PURGE attribute value and status of the object, and the length of time the object has maintained that status.

Operand Descriptions

RODMAPPL

The user application ID used to access the RODM that contains the topology and status data. The ID is case-sensitive and must be entered exactly as created with RACF.

| The application ID for MultiSystem Manager is set during NetView
| initialization. Specify this keyword only if you are overriding the ID used by
| MultiSystem Manager, or if the COMMON.FLC_RODMAPPL statement in
| CNMSTYLE has not been coded.

RODMCLASSNAME

The RODM class name of the object that is the target of the command.

RODMNAME

| The name of the RODM where topology and status information is stored. This
| is normally the RODM used by MultiSystem Manager. For MultiSystem
| Manager, this is the 1- to 8-character name that was specified on the
| COMMON.FLC_RODMNAME statement in CNMSTYLE. Specify this keyword
| only if you are overriding the ID used by MultiSystem Manager, or if the
| statement in CNMSTYLE has not been coded.

RODMOBJECTID

The RODM object ID (ObjectID) of the object. The object ID is 16 characters that represent an 8-byte hexadecimal field; for example: 00010027EC211161.

You can enter either the RODM object ID or the RODM class name and object name combination. MultiSystem Manager uses the object ID in the commands that it builds during command facility processing, because it can determine the ID based on the object you selected in the view. If you are entering this command from the command line or an automated procedure, use the RODM class name and object name combination.

RODMOBJECTNAME

The RODM object name of the object that is the target of the command.

STATUS

The current status of the real object you are removing. You can enter UNKNOWN or UNSATISFACTORY.

An aggregate object's status is a reflection of the real objects that it contains, so if you issue this command on an aggregate object, this is the status of the real objects in the aggregate that you want removed.

TIME

The minimum time interval an object must have been in the specified status. You can specify *days* and *time*, or just *time*. If you want days without time, enter 0 for *time*. Specify the interval in the format: *ddd hh:mm:ss*, where:

ddd 0–365 days

hh 0–23 hours

mm 00–59 minutes

ss 00–59 seconds

Minimum

:00 (0 seconds)

Default

1 day (24 hours)

For example:

2 full days:

2 0

0 days, 5 minutes:

5

0 days, 5 minutes, 3 seconds:

0:05:03

2 days, 1 hour, 10 minutes, 30 seconds:

2 1:10:30

0 days, 30 seconds:

:30

TRACE

Determines whether a trace is generated.

YES MultiSystem Manager generates messages FLC040I, FLC041I, and FLC042I, which contain a running account of the objects that were removed. If an aggregate object is selected for removal, the trace messages indicate the real objects in the aggregate that are removed.

NO No account of the objects removed is generated.

Usage Notes

The display of special connectivity relationships, such as in an IBM Token-Ring, might be affected by the removal of objects in the view. If the workstation will not display the view after the REMVOBJS command runs, you must rebuild the view by issuing the appropriate GETTOPO command.

You can issue the REMVOBJS command on an aggregate or a real object. When you issue this command on an aggregate object, the objects that make up the

aggregate are also affected. Removal of the real objects that comprise an aggregate depends on their PURGE attributes, statuses, and length of time at those statuses.

When you issue this command on a real object, only the real object is affected. Removal of a real object depends on its PURGE attribute, status, and the length of time in that status.

- If you are removing a real object:
 - Ensure that it has a status of *unknown* or *unsatisfactory*.
 - Determine the minimum time interval that will allow removal of the object.
- If you are removing an aggregate object:
 - Determine whether you want to remove its composite real objects that have statuses of *unknown* or *unsatisfactory*.
 - Determine the minimum time interval that allows removal of the real objects.

Because an aggregate reflects the status of its composite real objects, the status and time do not apply directly to the aggregate, but to the real objects in it.

- Ensure that the object's PURGE attribute permits it to be removed. You can use the NetView RODMVIEW command to determine the current value of the PURGE attribute. You can use the SETREMOV command to change the value. The PURGE attribute values are:

0 You can remove the object and all its links from the views and RODM if all other criteria are met.

This value is valid for real or aggregate objects. Real objects are removed if they have a status of *unsatisfactory* or *unknown* for the time specified in the REMVOBJS command. Aggregate objects are removed if all real objects in them have been removed.

All MultiSystem Manager objects have an initial PURGE attribute of 0.

1 You cannot remove the object from RODM. However, any of the links that connect the object to a object with a PURGE attribute of 0 can be removed. For example, if an aggregate object has a PURGE attribute of 0, the real object with a PURGE attribute of 1 can be unlinked and removed from the view.

This value is valid only for real objects. If you specify a value of 1 for an aggregate object, MultiSystem Manager treats it as if it has a value of 2.

2 You cannot remove the object or its links from the views or RODM.

This value is valid for real and aggregate objects. If a REMVOBJS command is issued on aggregate objects, this value shields the aggregate and the real objects in it from removal. If an aggregate has a value of 2, the real objects in the aggregate are not removed, regardless of their PURGE attribute values.

If a PURGE attribute was never defined for a object, MultiSystem Manager treats the object as if it has a value of 2.

Command Responses

The REMVOBJS command returns one or more of the following messages when TRACE=YES. Refer to *Tivoli NetView for z/OS Messages and Codes* for a complete description of MultiSystem Manager messages.

- FLC040I OBJECT *object_name* AND ALL OF ITS LINKS WERE REMOVED.
- FLC041I OBJECT *object_name* AND ITS LINKS WERE NOT REMOVED.

- FLC042I THE LINK BETWEEN *object_name1* FIELD *field_name1* AND *object_name2* FIELD *field_name2* WAS REMOVED.

Examples

This example removes the composite real objects of aggregate object number 00010027EC211161 that have been in unknown status for more than 29 hours. If all of the composite real objects are removed, the aggregate is removed. The name of the RODM that is being used is RODMNAME and the application name is MYAPPL.

```
REMOBJS RODMNAME=RODMNAME,RODMAPPL=MYAPPL,  
RODMOBJECTID=00010027EC211161,STATUS=UNKNOWN,  
TIME=1 5:00:00
```

RESTOPO

Syntax

▶▶—RESTOPO—◀◀

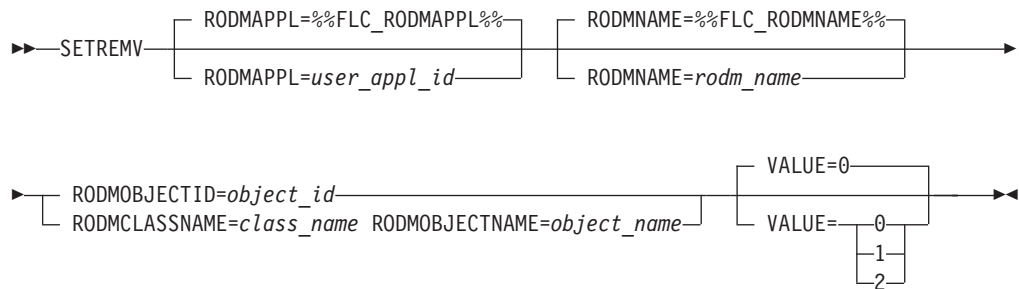
Purpose of Command

The RESTOPO command resumes processing of MultiSystem Manager GETTOPO commands for all NetView operators and sets the status of the topology manager objects displayed in the Graphic Monitor-Details window to *AVAILABLE*.

The status of the MultiSystem Manager must be *SUSPENDED* to resume processing. After successful completion of the RESTOPO command, the MultiSystem Manager's status is set to *enabled*.

SETREMOV

Syntax



Purpose of Command

The SETREMOV command sets the value of an object's PURGE attribute in RODM to indicate whether the object and its links can be removed. You can issue this command to an aggregate or a real object. If you issue it to an aggregate object, the current PURGE attributes of the objects that make up the aggregate are not changed.

Operand Descriptions

RODMAPPL

The user application ID used to access the RODM that contains the topology and status data. The ID is case-sensitive and must be entered exactly as created with RACF.

| The application ID for MultiSystem Manager is set during NetView
| initialization. Specify this keyword only if you are overriding the ID used by
| MultiSystem Manager, or if the COMMON.FLC_RODMAPPL statement in
| CNMSTYLE has not been coded.

RODMCLASSNAME

The RODM class name of the object that is the target of the command.

RODMNAME

| The name of the RODM where topology and status information is stored. This
| is normally the RODM used by MultiSystem Manager. For MultiSystem
| Manager, this is the 1- to 8-character name that was specified on the
| COMMON.FLC_RODMNAME statement in CNMSTYLE. Specify this keyword
| only if you are overriding the ID used by MultiSystem Manager, or if the
| statement in CNMSTYLE has not been coded.

RODMOBJECTID

The RODM object ID (ObjectID) of the object. The object ID is 16 characters that represent an 8-byte hexadecimal field. An example object ID is: 00010027EC211161.

You can enter either the RODM object ID or the RODM class name and object name combination. MultiSystem Manager uses the object ID in the commands that it builds during command facility processing, because it can determine the ID based on the object you selected from the view. If you are entering this command from the command line or an automated procedure, use the RODM class name and object name combination.

RODMOBJECTNAME

The RODM object name of the object that is the target of the command.

VALUE

The value for the object's PURGE attribute, indicating what you can remove. Valid values are:

- 0** You can remove the object and all its links from the views and RODM if all other criteria are met.

This value is valid for real or aggregate objects. Real objects are removed if they have a status of *unsatisfactory* or *unknown* for the time specified in the REMVOBJS command. Aggregate objects are removed if all real objects in them have been removed.

All MultiSystem Manager objects have an initial PURGE attribute of 0.

- 1** You cannot remove the object from RODM. However, any of the links that connect the object to a object with a PURGE attribute of 0 can be removed. For example, if an aggregate object has a PURGE attribute of 0, the real object with a PURGE attribute of 1 can be unlinked and removed from the view.

This value is valid only for real objects. If you specify a value of 1 for an aggregate object, MultiSystem Manager treats it as if it has a value of 2.

- 2** You cannot remove the object or its links from the views or RODM.

This value is valid for real and aggregate objects. If a REMVOBJS command is issued on aggregate objects, this value shields the aggregate and the real objects in it from removal. If an aggregate has a value of 2, the real objects in the aggregate are not removed, regardless of their PURGE attribute values.

If a PURGE attribute was never defined for a object, MultiSystem Manager treats the object as if it has a value of 2.

Examples

This example changes the PURGE attribute so that the object ID 00010027EC11161 and all of its links are subject to removal. The name of the RODM that is used by MultiSystem Manager is EKGXRODM and the application name is MYAPPL.

```
SETREMV
```

```
RODMNAME=EKGXRODM,RODMAPPL=MYAPPL,RODMOBJECTID=00010027EC11161,VALUE=0
```

SUSPTOPO

Syntax

▶▶—SUSPTOPO—◀◀

Purpose of Command

The SUSPTOPO command suspends processing of MultiSystem Manager topology requests (GETTOPO commands) for all NetView operators. If a GETTOPO command is issued while MultiSystem Manager processing is suspended, the GETTOPO command is ignored and message FLC045E is generated.

The status of the MultiSystem Manager must be *enabled* in order to suspend processing. After successful completion of this command, MultiSystem Manager status is set to *suspended*.

Chapter 7. MultiSystem Manager IP Commands

This section contains all MultiSystem Manager IP commands, which are listed in alphabetical order.

CRITRES

Syntax

▶—CRITRES—▶

Purpose of Command

The CRITRES command reads the TN3270 Management configuration file. The parameters for this command are in the configuration file.

Operand Descriptions

IPSTACK_RUN

Sets the TN3270 Management feature to run when the IPSTACK_RUN parameter is set to any of the following:

- YES
- yes
- Y
- y

Any other value will not implement the TN3270 feature after the GETTOPO IPRES command completes successfully.

ServerClient

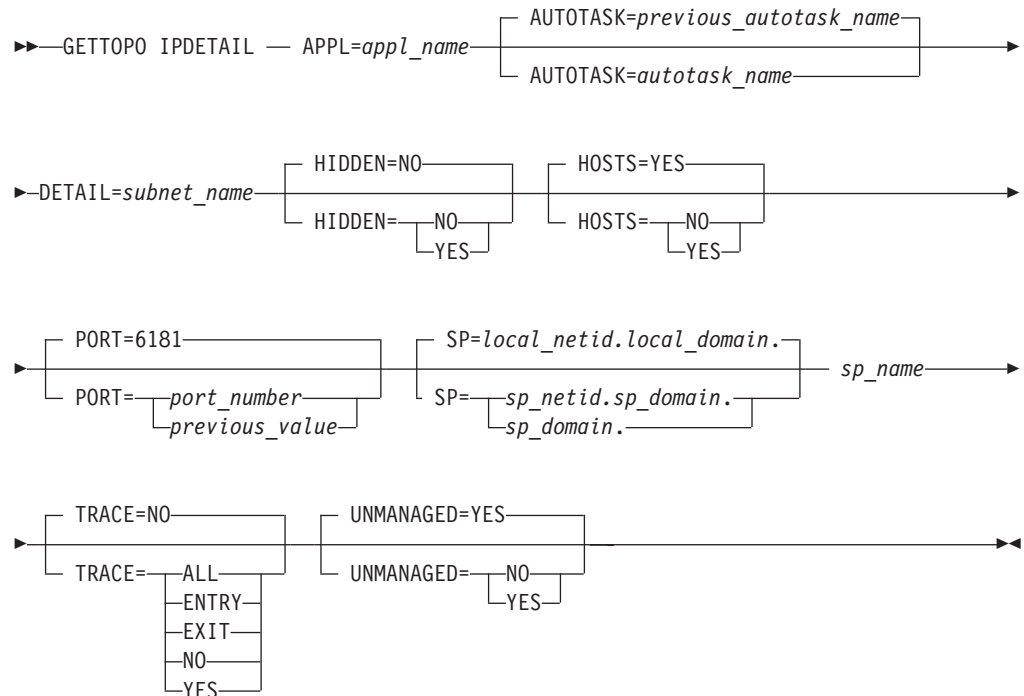
Sets the TN3270 Management feature to discover TN3270 servers or TN3270 servers and clients. If the ServerClient parameter is set to 1, then only TN3270 servers are discovered. If the ServerClient parameter is set to 2, then both TN3270 servers and clients are discovered. Any other value on this parameter causes neither the servers nor the clients to be discovered. If the parameter is set to 2, then there is one list of TN3270 client critical resources that can be used to limit the number of TN3270 clients discovered.

There is one list for TN3270 critical client resource addresses. Either a valid, fully qualified IP address or a wildcard can be used. The wildcard value must be an "*" at the end of the IP address. For example, 9.* is a valid value, but 9*.199.124 is not valid. If either the wildcard or a fully qualified IP address is determined to be invalid, then this address is discarded and the FLC151E error message is displayed.

Note: Any resources that are no longer critical are deleted from RODM.

GETTOPO IPDETAIL

Syntax



Purpose of Command

The GETTOPO IPDETAIL command retrieves and adds the topology data and status for the specified IP object to RODM.

You can use SNA or TCP/IP to communicate between Tivoli NetView for z/OS and the service point. If your service point is running NetView for AIX, the communication protocol can be SNA or TCP/IP. The communication protocol for all other service points running the MultiSystem Manager IP agent is TCP/IP.

Operand Descriptions

APPL

The name of the service point application.

If you are using SNA to communicate between Tivoli NetView for z/OS and the service point, the service point is an AIX machine. In the AIX environment, the service point application name is defined when the options for host connection daemons are set up in NetView for AIX. The service point application name must be obtained from the NetView for AIX system programmer. This name is the same as the application name used on the NetView RUNCMD APPL keyword.

If you are using TCP/IP to communicate between Tivoli NetView for z/OS and the service point, you must specify FLCIP01 as the application name.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to

communicate with the service point. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

DETAIL

The name of the subnet or other resource on which you want topology information. You use the internet dot notation to specify the name. You can enter either the decimal name, such as 9.67.223.21, the full host name, such as gandalf.raleigh.ibm.com, or a shortened version of the host name, such as gandalf.

HIDDEN

Specify whether to retrieve and add to RODM information for resources that are hidden on the NetView or Network Node Manager map.

NO Do not add to RODM.

If information on a hidden resource is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other (non-hidden) resources.

YES Add to RODM.

HOSTS

Specify whether to retrieve and add to RODM the information for host systems discovered by NetView or the Network Node Manager.

The topology status for routers, hubs, bridges, and their interfaces is updated, regardless of whether HOSTS is YES or NO.

NO Do not add to RODM.

YES Add to RODM.

If information on the host systems is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other resources in the network.

PORT

(TCP/IP only)

Specifies the TCP/IP Sockets PORT number associated with the IP agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined on the workstation where the agent is installed. Refer to the customization information described in the README file shipped with the MultiSystem Manager IP agent for information about changing the default port number. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO

command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6181 is used as the default value for PORT.

SP The fully-qualified name of the service point where the MultiSystem Manager IP agent is running.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

(SNA only)

The network identifier of the SNA network in which the service point is located. Use of the 1- to 8-alphanumeric identifier is optional. If *sp_netid* is specified, *sp_domain* must also be specified.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

(SNA only)

The name of the NetView domain in which the service point resides. Use of the 1- to 5- alphanumeric name is optional. If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* must also be specified and is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point associated with the workstation where the IP Agent is installed. The *sp_name* is required.

If the connection between Tivoli NetView for z/OS and the service point is SNA, the *sp_name* is either the LU name if the connection is LU 6.2 or the PU name if the connection is SSCP-PU. The *sp_name* must be a 1–8 alphanumeric name. The name can contain the special characters #, @, or \$, and cannot start with a number. SNA connectivity is only supported when the MultiSystem Manager IP agent is running on a NetView for AIX service point.

If the connection between Tivoli NetView for z/OS and the service point is TCP/IP, the *sp_name* is the host name of the service point preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique. If the service point identified in the *sp_name* is in a cross-domain network (not in this domain), the fully-qualified host name must be used, for example, specify &smith.raleigh.ibm.com instead of &smith).

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command tree facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

UNMANAGED

Specify whether to display resources that are in the unmanaged state in the NetView or Network Node Manager view.

NO Do not display unmanaged resources.

YES Display unmanaged resources.

Usage Notes

- The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

- If you have views below the networks view open, you should close them before issuing this command. Closing them saves processing time.

If you do not close the views, they might be closed by the workstation and the following error message is received:

```
DUI16441I:Your request to refresh viewname could not be completed.  
The view has been closed
```

If this occurs, you can ignore the message and reopen the views.

Examples

The GETTOPO IPDETAIL example provides the following:

- Retrieves topology and status from service point NTB7I445 down to all of its managed children.
- Retrieves topology and status information for all subnets, locations and segments in the subnet 9.67.32. This includes all router and router interfaces for the subnet.
- Sends the command to the MultiSystem Manager agent named FLCIP01 on the service point.
- Uses the AUTOIP1 autotask to communicate with the service point.
- Adds information for host systems to RODM by taking the default of YES for the HOSTS keyword.
- Does not display trace messages while processing the request by taking the default of NO for the TRACE keyword.
- Displays unmanaged resources by taking the default value of YES for the UNMANAGED keyword.

```
GETTOPO IPDETAIL DETAIL=9.67.32 APPL=FLCIP01 AUTOTASK=AUTOIP1  
SP=NTB7I445
```

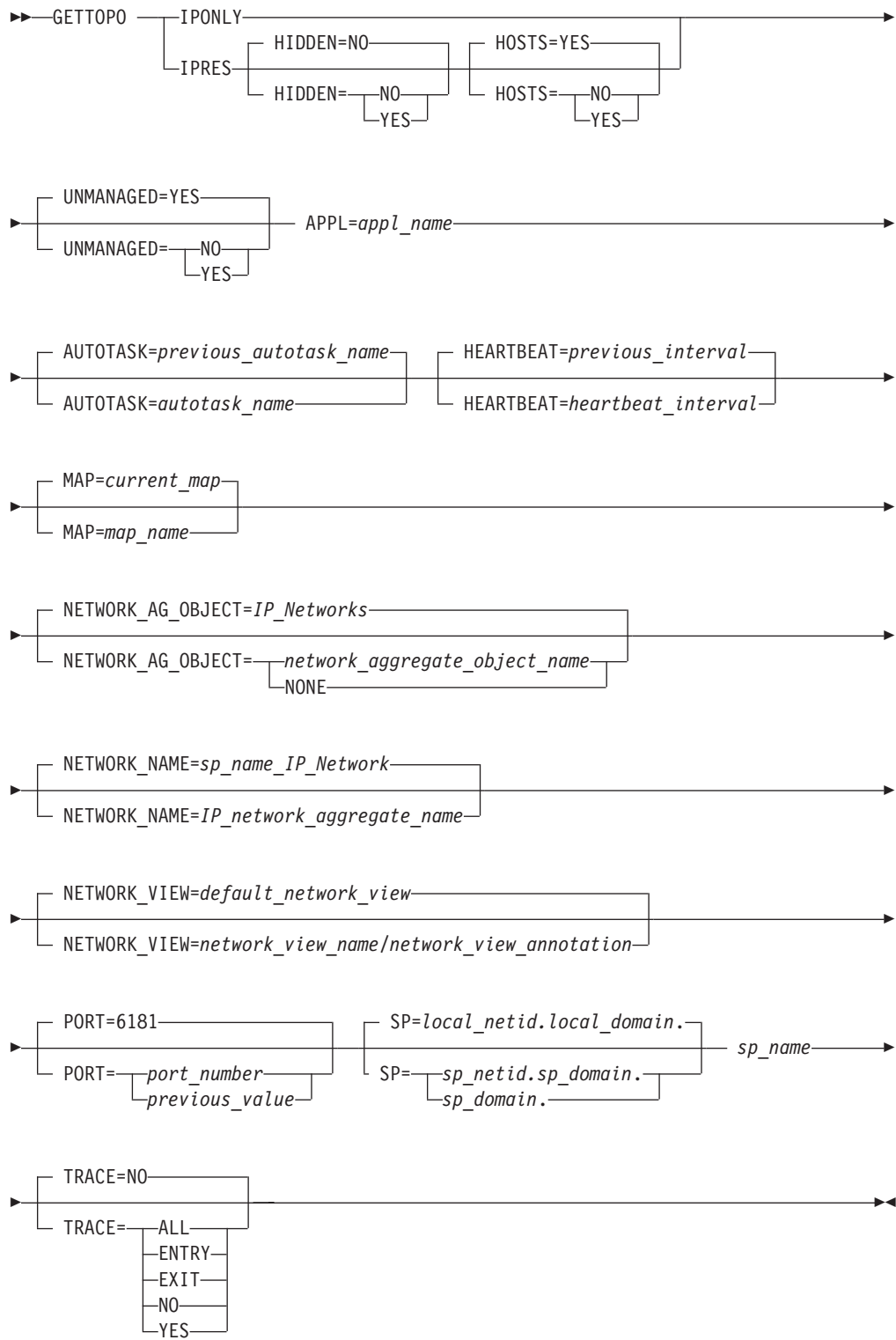
The GETTOPO IPDETAIL example provides the following:

- Retrieves topology and status for service point NTB7I445.
- Retrieves topology and status information for all subnets, locations and segments in the subnet 9.67.32. This includes all router and router interfaces for the subnet.
- Sends the command to the MultiSystem Manager agent named FLCIP01 on the service point.
- Uses the AUTOIP1 autotask to communicate with the service point.
- Does not add information for host systems and host interfaces (HOSTS=NO).
- Displays trace messages while processing the request (TRACE=YES).
- Does not display unmanaged resources.

```
GETTOPO IPDETAIL DETAIL=9.67.32 APPL=FLCIP01 AUTOTASK=AUTOIP1  
HOSTS=NO SP=NTB7I445 TRACE=YES UNMANAGED=NO
```

GETTOPO IPONLY and IPRES

Syntax



Purpose of Command

The GETTOPO IPONLY and IPRES command retrieves topology and status information for a network managed by the MultiSystem Manager IP agent and adds the information to RODM.

You can use SNA or TCP/IP to communicate between Tivoli NetView for z/OS and the service point. If your service point is running NetView for AIX, the communication protocol can be SNA or TCP/IP. The communication protocol for all other service points running the MultiSystem Manager IP agent is TCP/IP.

Operand Descriptions

APPL

The name of the service point application.

If you are using SNA to communicate between Tivoli NetView for z/OS and the service point, the service point is an AIX machine. In the AIX environment, the service point application name is defined when the options for host connection daemons are set up in NetView for AIX. The service point application name must be obtained from the NetView for AIX system programmer. This name is the same as the application name used on the NetView RUNCMD APPL keyword.

If you are using TCP/IP to communicate between Tivoli NetView for z/OS and the service point, you must specify FLCIP01 as the application name.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the service point. This is the autotask from which the topology request, in the form of a RUNCMD, is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time, in minutes, between heartbeat requests to the service point. Heartbeat requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The heartbeat request can result in notification that topology or status changes have occurred.

Specify a zero (0) to stop current heartbeat requests for the service point.

If the HEARTBEAT keyword is not coded, it defaults to the current heartbeat interval. If an interval was never set, MultiSystem Manager sets it to zero.

Value: An integer from 0–3599. A value of zero means that no heartbeat requests are issued.

Default:

The previous value.

HIDDEN

Specify whether to retrieve and add to RODM the information for resources that are hidden on the NetView or Network Node Manager map.

NO Do not add to RODM.

If information on a hidden resource is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other (non-hidden) resources.

YES Add to RODM.

HOSTS

Specify whether to retrieve and add to RODM the information for host systems discovered by NetView or Network Node Manager.

The topology status for routers, hubs, bridges, and their interfaces is updated, regardless of whether HOSTS is YES or NO.

NO Do not add to RODM.

YES Add to RODM.

If information on the host systems is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other resources in the network.

IPONLY

Retrieve and add to RODM the topology data and status for the specified service point only. Do not retrieve and store topology data and status for the resources managed by the service point.

IPRES

Retrieve and add to RODM the topology data and status for the specified service point and resources it manages.

MAP

The name of the NetView or Network Node Manager submap that you want to use for topology and status. If the specified map is not active at the service point workstation, MultiSystem Manager returns the message, FLCI011E Topology agent is managing map: *attached_map_name*, providing the name of the active map.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the syntax diagram.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

If you code this name or accept the default name, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the object on the NMC screen. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the NMC screen for the aggregate object representing this IP network. This name is stored in the DisplayResourceName field in RODM for this IP network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; : ? ' - _ % * < and >.

If the aggregate object representing this IP network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation* and they must be separated by a right slash (/). For example: NETWORK_VIEW=My_View/Joe's own view.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network_view name and annotation.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; : ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

If you do not specify NETWORK_VIEW, MultiSystem Manager uses the name specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

PORT

(TCP/IP only)

Specifies the TCP/IP Sockets PORT number associated with the IP agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined on the workstation where the agent is installed. Refer to the customization information described in the README file shipped with the MultiSystem Manager IP agent for information about changing the default port number. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6181 is used as the default value for PORT.

SP The fully-qualified name of the service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

(SNA only)

The network identifier of the SNA network in which the service point is located. Use of the 1–8 alphanumeric character *sp_netid* is optional. If *sp_netid* is specified, *sp_domain* must also be specified.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

(SNA only)

The name of the NetView domain in which the service point resides. Use of the 1–5 alphanumeric character *sp_domain* is optional. If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* must also be specified and is used to create an SNA_Domain_Class object in RODM. The *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point associated with the workstation where the IP Agent is installed. The *sp_name* is required.

If the connection between Tivoli NetView for z/OS and the service point is SNA, the *sp_name* is either the LU name if the connection is LU 6.2 or the PU name if the connection is SSCP-PU. The *sp_name* must be a 1–8 alphanumeric name. The name can contain the special characters #, @, or \$, and cannot start with a number. SNA connectivity is only supported when the MultiSystem Manager IP agent is running on the service point.

If the connection between Tivoli NetView for z/OS and the service point is TCP/IP, the *sp_name* is the host name of the service point preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique. If the service point identified in the *sp_name* is in a cross-domain network (not in this domain), the fully-qualified host name must be used, for example, specify &smith.raleigh.ibm.com instead of &smith.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command tree facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

UNMANAGED

Specify whether to display resources that are in the unmanaged state in the NetView or Network Node Manager view.

NO Do not display unmanaged resources.

YES Display unmanaged resources.

Usage Notes

- The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

- If you have views below the networks view open, you should close them before issuing this command. Closing the views saves processing time.

If you do not close the views, they might be closed by the workstation and the following error message is received:

```
DUI16441I:Your request to refresh viewname could not be completed.  
The view has been closed
```

If this occurs, you can ignore the message and reopen the views.

Examples

The GETTOPO IPRES example provides the following:

- Retrieves topology and status for the service point and all of the resources managed by service point NTBPU02.
- Does not add information for host systems and host interfaces (HOSTS=NO).
- Sends the command to the MultiSystem Manager agent named FLCIP01 on the service point.
- Uses the AUTOIP1 autotask to communicate with the service point.
- Specifies a heartbeat interval of 10 minutes.
- Places the view in the aggregate object *NV6K_Networks*.
- Names the aggregate object representing the network *My_NV6K_Network*.
- Names the view *My_View* with a description of *Joe's own view*.
- Displays trace messages for the GETTOPO command.
- Displays unmanaged resources.

```
GETTOPO IPRES HOSTS=NO APPL=FLCIP01 AUTOTASK=AUTOIP1 HEARTBEAT=10  
NETWORK_AG_OBJECT=NV6K_Networks NETWORK_NAME=My_NV6K_Network  
NETWORK_VIEW=My_View/Joe's own view SP=NTBPU02 TRACE=YES
```

The GETTOPO IPONLY example provides the following:

- Retrieves topology and status for service point NTBPU02.
- Sends the command to the MultiSystem Manager agent named FLCIP01 on the service point.
- Uses the AUTOIP1 autotask to communicate with the service point.
- Uses the previous heartbeat interval by taking the default for the HEARTBEAT keyword.
- Does not include the objects representing the network in an aggregate object (NETWORK_AG_OBJECT=NONE)
- Names the view *My_View* with a description of *Regina's own view*.
- Does not display trace messages by taking the default for the TRACE keyword.

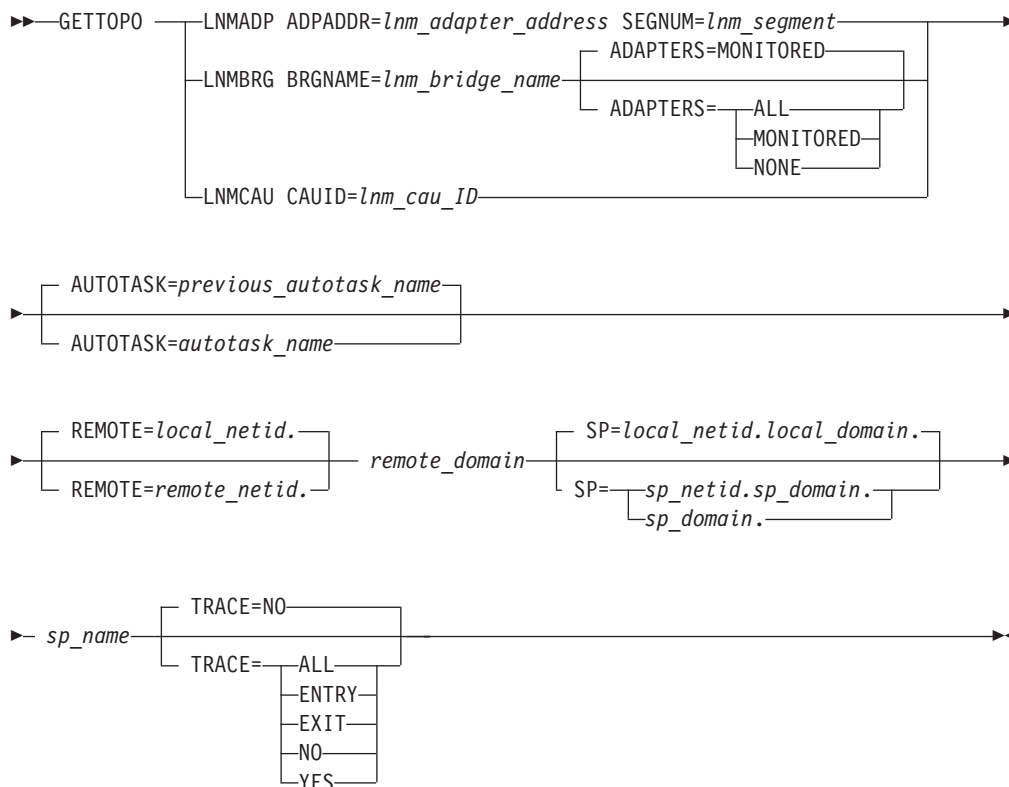
```
GETTOPO IPONLY APPL=FLCIP01 AUTOTASK=AUTOIP1  
NETWORK_AG_OBJECT=NONE NETWORK_VIEW=My_View/Regina's own view  
SP=NTBPU02
```

Chapter 8. MultiSystem Manager LNM Commands

This section contains all MultiSystem Manager LNM commands, which are listed in alphabetical order.

GETTOPO LNMADP, LNMBRG and LNMCAU

Syntax



Purpose of Command

The GETTOPO LNMADP, LNMBRG and LNMCAU command retrieves topology and status information for a LAN Network Manager (LNM) adapter (LNMADP), bridge (LNMBRG), or hub (LNMCAU).

Operand Descriptions

ADAPTERS

Specifies the level of adapter information that you want added to RODM and presented on the view as a result of this GETTOPO command.

Note: The ADAPTERS keyword can only be used with the LNMBRG parameter.

ALL Add to RODM, information for all bridge adapters.

LNM only generates alerts for adapters that it monitors. Information for an adapter must be stored in RODM and monitored by LNM in order for MultiSystem Manager to dynamically update the status of that adapter.

MONITORED

Only add to RODM information for bridge adapters monitored by LNM.

If information on the adapters is already in RODM, MultiSystem Manager deletes the information for unmonitored adapters. MultiSystem Manager also updates the topology and status information for other resources in the network, including adapters monitored by this LNM.

NONE

Do not add information for any adapters to RODM.

If information on the adapters for the LNM is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other resources in the network.

If the LNM is monitoring adapters, it generates an alert whenever the status of a monitored adapter changes. If you specify ADAPTERS=NONE these alerts will be ignored for this LNM, because the information for these adapters is not in RODM. Issue the GETTOPO command (LNMRES or LNMSEG with ADAPTERS=MONITORED or ALL) to have the adapter topology data and status added to RODM.

ADPADDR

The name or address of the LNM adapter for which you are retrieving status. You can use either the 1–16 alphanumeric character name of the adapter or the 12 hexadecimal digit address. If you use the name, the only special characters MultiSystem Manager permits in the name are: @, #, and \$. It does not permit percent signs (%).

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

BRGNAME

The name of the LNM bridge for which you are retrieving topology data and status. Use the 1–8 alphanumeric character name of the bridge. MultiSystem Manager permits this name to include the special characters @, #, \$, and %.

CAUID

The ID of the LNM hub for which you are retrieving topology data and status. This name is 8-hexadecimal characters in length.

LNMA DP

Retrieve and update in RODM, the status for the LNM adapter specified in ADPADDR.

LNMBRG

Retrieve and add to RODM, topology data and status for the LNM bridge, specified in BRGNAME, and its adapters.

LNMC AU

Retrieve and store in RODM, topology data and status for the LNM hub, specified in CAUID, and its adapters.

REMOTE

The remote NetView domain in which this LNM service point resides. REMOTE must be specified if the LNM service point is connected through SSCP-PU and resides in a NetView domain other than the one in which MultiSystem Manager resides. If REMOTE is coded, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. Use of the 1–8 alphanumeric character *remote_netid* is optional. If you code *remote_netid*, separate it from *remote_domain* with a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. MultiSystem Manager permits a 1–5 alphanumeric character NetView domain name. It is used as the LU parameter on the RMTCMD command.

SEGNUM

The segment number of the LNM segment on which this adapter resides. The segment number is 1–4 hexadecimal characters in length. If you enter a 1–3 character number, it will have additional zeros on the left to make a 4-character number.

SP

The fully-qualified SNA name of the LNM service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

The network identifier of the SNA network in which the service point is located. Use of the 1–8 alphanumeric character *sp_netid* is optional. If *sp_netid* is specified, *sp_domain* must also be specified.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

The name of the NetView domain in which the service point resides. Use of the 1–5 alphanumeric character *sp_domain* is optional. If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* must also be specified and is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the service point associated with the LNM being described. If the LNM is connected through an SSCP-PU session, this is the name by which the PU is known to VTAM. If the LNM is connected through LU 6.2, this is the LU name.

Use of the 1–8 alphanumeric character *sp_name* is required. It is used as the SP parameter on RUNCMDs sent to the service point. MultiSystem Manager permits this name to include only these special characters: #, @, or \$. MultiSystem Manager expects it to start with an alphabetic character, #, @, or \$.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The following conditions must exist before you can use this command:

- The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

- If LNMBRG is specified, the topology information for the related LNM that is managing this network must be in RODM. If it is not, you can issue a GETTOPO LNMRES or GETTOPO LNMONLY command for this LNM. You can also include a GETTOPO LNMRES or a GETTOPO LNMONLY statement for this LNM in the initialization file and re-issue the INITTOPO command.
- If LNMCAU is specified, the topology information for the related LNM that is managing this network must be in RODM. If it is not, you can issue a GETTOPO LNMRES command for this LNM. You can also include a GETTOPO LNMRES statement for this LNM in the initialization file and re-issue the INITTOPO command.
- If LNMADP is specified, the topology information for the related adapter must be in RODM. If it is not, you can issue either a GETTOPO LNMBRG or a GETTOPO LNMRES command for this LNM, with ADAPTERS=ALL or ADAPTERS=MONITORED. You can also issue a GETTOPO LNMONLY command followed by a GETTOPO LNMSEG command for the specific segment on which the adapter resides. In either case, specify ADAPTERS=ALL or ADAPTERS=MONITORED, depending on whether the adapter is monitored. Another option is to include a GETTOPO LNMRES statement for this LNM in the initialization file and re-issue the INITTOPO command.

Examples

The following adapter example:

- Retrieves and updates topology and status information for the local adapter 40001A8B006 on segment number 018B, managed by service point NTB6PU02.
- Uses the AUTOLNM autotask to communicate with the LNM.
- Does not display trace messages.

```
GETTOPO LNMADP ADPADDR=40001A18B006 SEGNUM=018B AUTOTASK=AUTOLNM
SP=NTB6PU02
```

The following bridge example:

- Retrieves and updates topology and status information for the local bridge named MYBRG and its adapters. The bridge is managed by service point NTB6PU02.
- Uses the AUTOLNM autotask to communicate with the LNM.
- Does not display trace messages.

```
GETTOPO LNMBRG BRGNAME=MYBRG ADAPTERS=ALL AUTOTASK=AUTOLNM SP=NTB6PU02
```

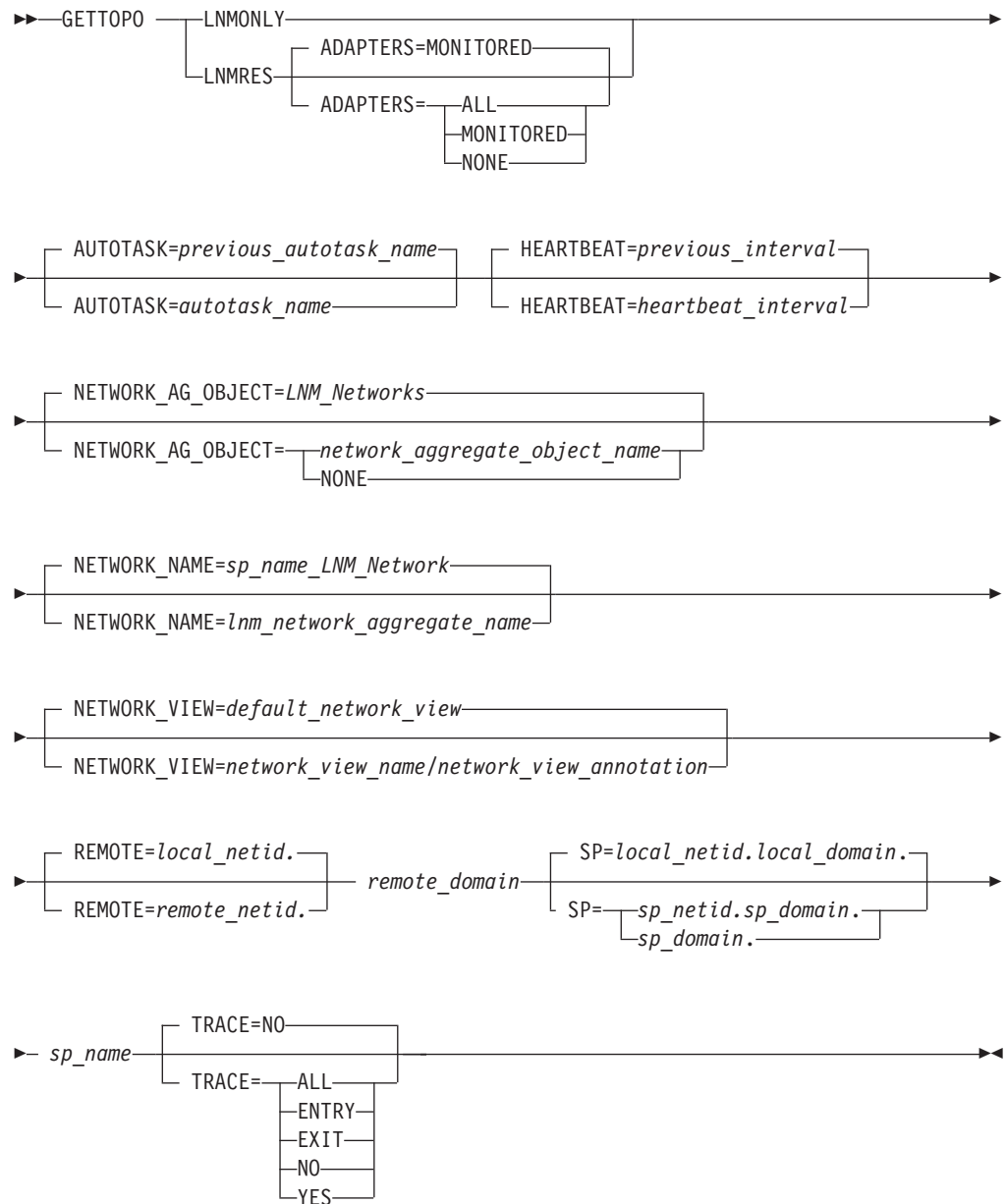
The following hub example:

- Retrieves and updates topology and status information for the local hub and its adapters. The hub is managed by service point NTB6PU02.
- Uses the AUTOLNM autotask to communicate with the LNM.
- Does not display trace messages.

```
GETTOPO LNMCAU CAUID=5A98CCA2 AUTOTASK=AUTOLNM SP=NTB6PU02
```

GETTOPO LNMRES and LNMONLY

Syntax



Purpose of Command

The GETTOPO LNMRES and LNMONLY command retrieves and updates in RODM topology and status information for a LAN Network Manager (LNM) network, and its resources.

MultiSystem Manager supports LNMs running in observing or controlling mode. If you issue a GETTOPO command for an LNM running in observing mode, you receive message FLC005I.

If you issue GETTOPO commands for observing LNMs and a controlling LNM that manage the same resources, MultiSystem Manager creates objects in RODM that represent each of the LNMs and their associated resources. This duplicate representation of resources in RODM can be avoided by issuing a GETTOPO statement for the controlling LNM only.

Operand Descriptions

ADAPTERS

Specifies the level of adapter information that you want added to RODM and presented on the view as a result of this GETTOPO command.

Note: The ADAPTERS keyword can only be used with the LNMRES parameter.

ALL Add to RODM, information for all adapters, hub adapters, and bridge adapters known to LNM.

LNM only generates alerts for adapters that it monitors. Information for an adapter must be stored in RODM and monitored by LNM in order for MultiSystem Manager to dynamically update the status of that adapter.

MONITORED

Only add to RODM information for monitored adapters, all hub adapters, and monitored bridge adapters known to LNM.

If information on the adapters is already in RODM, MultiSystem Manager deletes the information for unmonitored adapters. MultiSystem Manager also updates the topology and status information for other resources in the network, including adapters monitored by this LNM.

NONE

Do not add information for any adapters to RODM.

If information on the adapters for the LNM is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other resources in the network.

If the LNM is monitoring adapters, it generates an alert whenever the status of a monitored adapter changes. If you specify ADAPTERS=NONE these alerts will be ignored for this LNM, because the information for these adapters is not in RODM. Issue the GETTOPO command (LNMRES or LNMSEG with ADAPTERS=MONITORED or ALL) to have the adapter topology data and status added to RODM.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time, in minutes, between heartbeat requests to the service point. Heartbeat requests are queries issued by MultiSystem Manager to ensure that the service point is still active.

Specify a zero (0) to stop current heartbeat requests for the service point.

If the HEARTBEAT keyword is not coded, it defaults to the current heartbeat interval. If an interval was never set, MultiSystem Manager sets it to zero.

Value: An integer from 0–3599. A value of zero means that no heartbeat requests are issued.

Default:

The previous value.

LNMONLY

Retrieve, and add to RODM, topology data and status for the specified LNM only. Do not retrieve and store topology data and status for the resources managed by the LNM.

LNMRES

Retrieve, and update in RODM, topology data and status for the specified LNM and the resources it manages. ADAPTERS will be used to determine if topology and status information for adapters will be retrieved and stored in RODM.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the syntax diagram.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

If you code this name or accept the default name, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the object on the screen. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this LNM network. This name is stored in the DisplayResourceName field in RODM for this LNM network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ; ? ' - _ & + % * < and >.

The name specified does not have to be the same as the LNM name specified on the **Miscellaneous Parameters** pull-down on the LNM workstation. However, we recommend that the names be the same to help you correlate the NMC display with the physical workstation.

If the aggregate object representing this LNM network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation* and they must be separated by a right slash (/). For example: NETWORK_VIEW=My_View/Joe's own view.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network_view name and annotation.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

If you do not specify NETWORK_VIEW, MultiSystem Manager uses the name specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

REMOTE

The remote NetView domain in which this LNM service point resides.

REMOTE must be specified if the LNM service point is connected through SSCP-PU and resides in a NetView domain other than the one in which MultiSystem Manager resides. If REMOTE is coded, NetView uses RMTCCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. Use of the 1–8 alphanumeric character *remote_netid* is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. MultiSystem Manager permits a 1–5 alphanumeric character NetView domain name. It is used as the LU parameter on the RMTCCMD command.

SP The fully-qualified SNA name of the LNM service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

The network identifier of the SNA network in which the service point is located. Use of the 1–8 alphanumeric character *sp_netid* is optional. If *sp_netid* is specified, *sp_domain* must also be specified.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

The name of the NetView domain in which the service point resides. Use of the 1–5 alphanumeric character *sp_domain* is optional. If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* must also be specified and is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the service point associated with the LNM being described. If the LNM is connected through an SSCP-PU session, this is the name by which the PU is known to VTAM. If the LNM is connected through LU 6.2, this is the LU name.

Use of the 1–8 alphanumeric character *sp_name* is required. It is used as the SP parameter on RUNCMDs sent to the service point. MultiSystem Manager permits this name to include only these special characters: #, @, or \$. MultiSystem Manager expects it to start with an alphabetic character, #, @, or \$.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

Examples

The following GETTOPO LNMRES example:

- Retrieves topology and status for the local LNM managed by service point NTBPU02.
- Retrieves topology and status for monitored adapters, all hubs, and monitored bridge adapters known to the LNM.
- Uses the AUTOLNM autotask to communicate with the LNM.
- Specifies a heartbeat interval of 10 minutes.
- Places the view in the aggregate object *LNM_Networks*.
- Names the aggregate object representing the network *My_LNM_Network*.
- Names the view *My_View* with a description of *Joe's own view*.

- Displays trace messages for the GETTOPO command.

```
GETTOPO LNMRES AUTOTASK=AUTOLNM HEARTBEAT=10
NETWORK_AG_OBJECT=LNM_Networks NETWORK_NAME=My_LNM_Network
NETWORK_VIEW=My_View/Joe's own view SP=NTBPU02 TRACE=YES
```

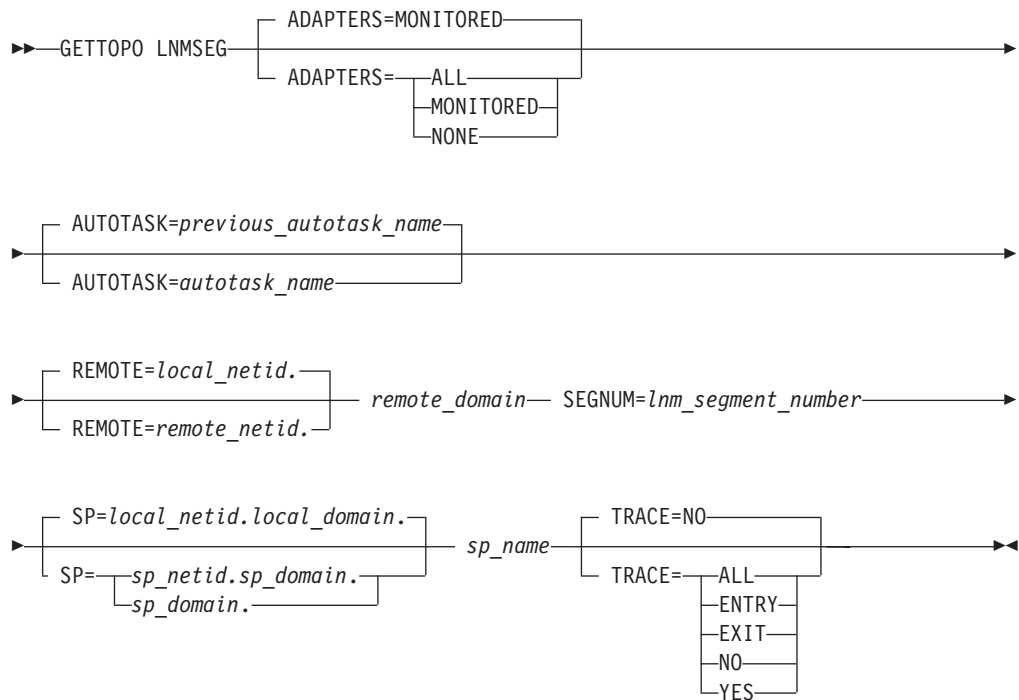
The following GETTOPO LNMONLY example:

- Retrieves topology and status for the local LNM managed by service point NTBPU02.
- Uses the AUTOLNM autotask to communicate with the LNM.
- Places the view in the aggregate object *LNM_Networks*.
- Names the aggregate object representing the network *My_LNM_Network*.
- Names the view *My_View* with a description of *Paul's own view*.

```
GETTOPO LNMONLY AUTOTASK=AUTOLNM NETWORK_AG_OBJECT=LNM_Networks
NETWORK_NAME=My_LNM_Network NETWORK_VIEW=My_View/Paul's own view
SP=NTBPU02
```

GETTOPO LNMSEG

Syntax



Purpose of Command

The GETTOPO LNMSEG command retrieves and updates in RODM, topology and status information for a LAN Network Manager (LNM) segment, and its associated adapters.

Usage Notes

The following conditions must exist before you can use this command.

- The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

- The topology information for the related LNM that is managing this network must be in RODM. If it is not, you can issue a GETTOPO LNMRES or a GETTOPO LNMONLY command for this LNM. You can also include a GETTOPO LNMRES or a GETTOPO LNMONLY statement for this LNM in the initialization file and re-issue the INITTOPO command.

Operand Descriptions

ADAPTERS

Specifies the level of adapter information that you want added to RODM for presentation on a view.

ALL Add to RODM information for all adapters on this segment.

LNМ only generates alerts for adapters that it monitors. Information for an adapter must be stored in RODM and monitored by LNM in order for MultiSystem Manager to dynamically update the status of that adapter.

MONITORED

Only add to RODM information for adapters on this segment that are monitored by LNM.

If information on the adapters is already in RODM, MultiSystem Manager deletes the information for unmonitored adapters. MultiSystem Manager also updates the topology and status information for other resources in the network, including adapters on this segment that are monitored by this LNM.

Refer to the LAN Network Manager for OS/2 Reference for more detail on monitoring adapters.

NONE

Do not add information for any adapters on this segment in RODM.

If information on the adapters for the LNM is already in RODM, MultiSystem Manager removes the information from RODM and updates the topology and status of other resources in the network.

If the LNM is monitoring adapters, it generates an alert when the status of a monitored adapter changes. If you specify ADAPTERS=NONE, these alerts will be ignored for this LNM, because the information for these adapters is not in RODM. Issue the GETTOPO command (LNMRES or LNMSEG with ADAPTERS=MONITORED or ALL) to have the adapter topology data and status added to RODM.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

REMOTE

The remote NetView domain in which this LNM service point resides. REMOTE must be specified if the LNM service point is connected through SSCP-PU and resides in a NetView domain other than the one in which MultiSystem Manager resides. If REMOTE is coded, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. Use of the 1–8 alphanumeric character

remote_netid is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. MultiSystem Manager permits a 1–5 alphanumeric character NetView domain name. It is used as the LU parameter on the RMTCMD command.

SEGNUM

The segment number of the LNM segment on which this adapter resides. The segment number is 1–4 hexadecimal characters in length. If you enter a 1–3 character number, it will have additional zeros on the left to make a 4-character number.

SP

The fully-qualified SNA name of the LNM service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

The network identifier of the SNA network in which the service point is located. Use of the 1–8 alphanumeric character *sp_netid* is optional. If *sp_netid* is specified, *sp_domain* must also be specified.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

The name of the NetView domain in which the service point resides. Use of the 1–5 alphanumeric character *sp_domain* is optional. If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* must also be specified and is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the service point associated with the LNM being described. If the LNM is connected through an SSCP-PU session, this is the name by which the PU is known to VTAM. If the LNM is connected through LU 6.2, this is the LU name.

Use of the 1–8 alphanumeric character *sp_name* is required. It is used as the SP parameter on RUNCMDs sent to the service point. MultiSystem Manager permits this name to include only these special characters: #, @, or \$. MultiSystem Manager expects it to start with an alphabetic character, #, @, or \$.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Examples

The following example:

- Retrieves topology and status information for local segment number 018B managed by service point NTB6PU02.
- Retrieves topology and status information for all adapters on the segment.
- Uses the AUTOLNM autotask to communicate with the LNM.

```
GETTOPO LNMSEG ADAPTERS=ALL AUTOTASK=AUTOLNM SEGNUM=018B SP=NTB6PU02
```

Chapter 9. MultiSystem Manager NetFinity Commands

This section contains all MultiSystem Manager NetFinity commands, which are listed in alphabetical order.

DISPMON

Syntax

▶▶—DISPMON *object_class object_name*—▶▶

Purpose of Command

The DISPMON command displays monitors and their thresholds that are in unsatisfactory status.

Operand Descriptions

object_class

The RODM class of the object that you are issuing the command against.

object_name

The RODM object name, specified in the RODM MyName field, for the object you are issuing the command against.

Usage Notes

When you issue this command from the MultiSystem Manager command facility, you can select only objects that belong to the MonitorManager class.

DISPPRC

Syntax

▶—DISPPRC *object_class object_name*—▶

Purpose of Command

The DISPPRC command displays a process that is in an alert condition.

Operand Descriptions

object_class

The RODM class of the object that you are issuing the command against.

object_name

The RODM object name, specified in the RODM MyName field, for the object you are issuing the command against.

Usage Notes

When you issue this command from the MultiSystem Manager command facility, you can select only objects that belong to the Process class.

DISPPRO

Syntax

▶—DISPPRO *object_class object_name*—▶

Purpose of Command

The DISPPRO command displays protocols and their status.

Operand Descriptions

object_class

The RODM class of the object that you are issuing the command against.

object_name

The RODM object name, specified in the RODM MyName field, for the object you are issuing the command against.

Usage Notes

When you issue this command from the MultiSystem Manager command facility, you can select only objects that belong to the OperatingSystem class.

DISPTHR

Syntax

▶▶—DISPTHR *object_class object_name*—▶▶

Purpose of Command

The DISPTHR command displays the names of thresholds that are in unsatisfactory status.

Operand Descriptions

object_class

The RODM class of the object that you are issuing the command against.

object_name

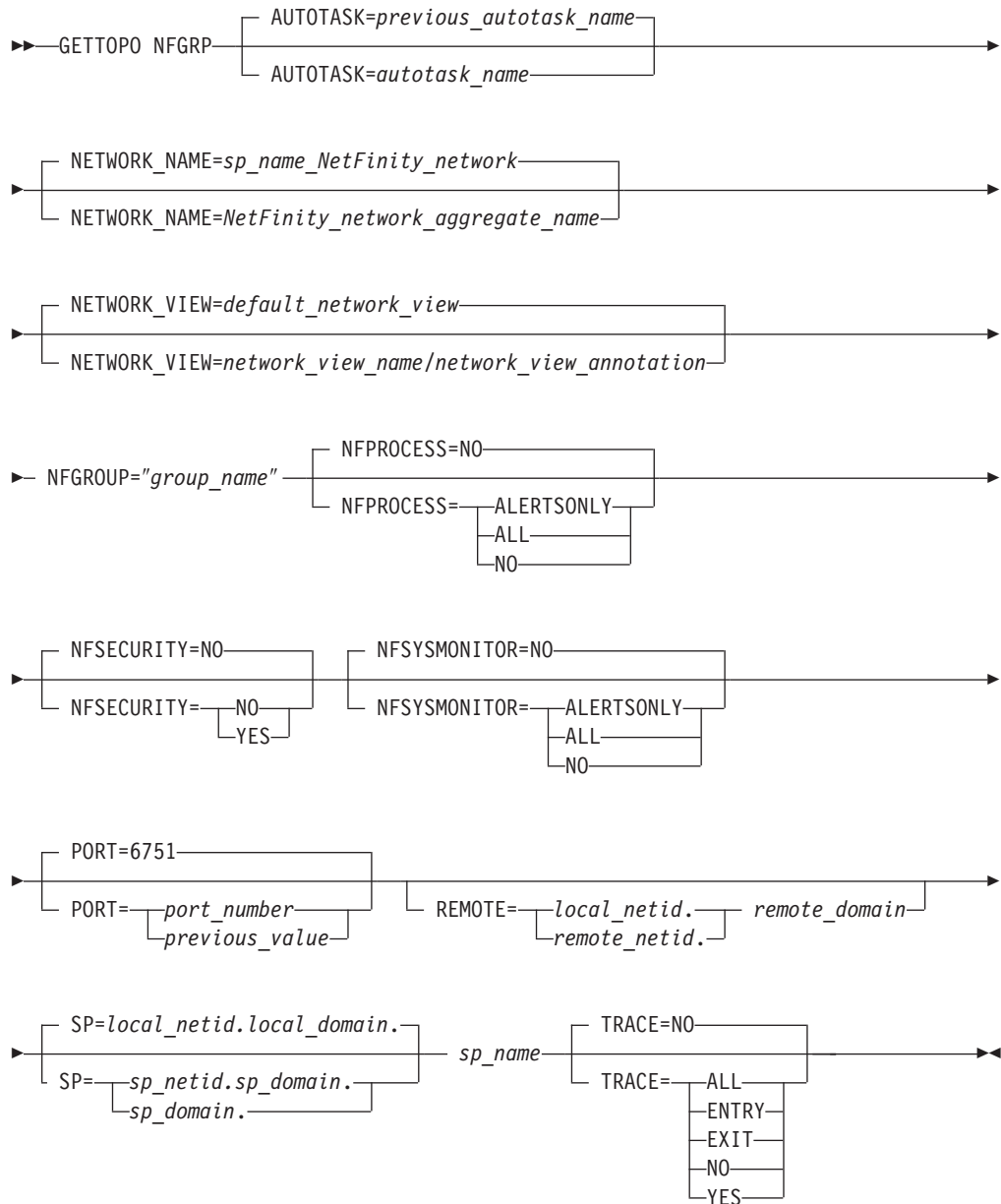
The RODM object name, specified in the RODM MyName field, for the object you are issuing the command against.

Usage Notes

When you issue this command from the MultiSystem Manager command facility, you can select only objects that belong to the Service class.

GETTOPO NFGRP

Syntax



- The PORT keyword is valid for Windows/NT only.
- The REMOTE keyword is valid for OS/2[®] only.
- To specify a TCP/IP address, code an ampersand (&) as the first character of *sp_name*. For example:
SP=&smith.raleigh.ibm.com

Code either the host name or the fully-qualified name.

- For SP, *sp_netid* and *sp_domain* are only valid for OS/2.

Purpose of Command

The GETTOPO NFGRP command retrieves and stores in RODM topology and status information for a group of NetFinity workstations.

Operand Descriptions

AUTOTASK

Specifies the name of the autotask that MultiSystem Manager should use to communicate with the service point. This is the autotask from which the topology request is issued. The autotask name must be 1–8 alphanumeric characters and can contain the special characters #, @, and \$. It cannot start with a number.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this network. This name is stored in the DisplayResourceName field in RODM for this network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ? ' - _ % * < and >.

If the aggregate object representing this network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:

```
NETWORK_VIEW=My_View/Joe's own view
```

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values.

If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network view name and annotation specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () : ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

NFGROUP

Specifies the NetFinity system group name. The group name must be in double quotes.

NFGRP

Retrieves and stores in RODM status and topology information for the specified NetFinity system group.

NFPROCESS

Specifies whether to create a ProcessManager class object, as well as objects for the processes that it manages.

NO Do not create a ProcessManager class object. Ignore all alerts that would affect this object.

ALL Create a ProcessManager class object, as well as objects for each process being managed.

ALERTSONLY

Create only a ProcessManager class object. Do not create any objects for the processes being managed, but record alerts that affect these managed processes and change the status of the ProcessManager class object to reflect these alerts.

NFSECURITY

Specifies whether to create a SecurityMonitor class object.

NO Do not create a SecurityMonitor class object. Ignore alerts that would normally affect this object.

YES Create a SecurityMonitor class object. Change the status of the object as appropriate to reflect alerts received by the object.

NFSYSMONITOR

Specifies whether to create a MonitorManager class object, as well as objects for the systems it monitors.

NO Do not create a MonitorManager class object. Ignore all alerts that would affect this object.

ALL Create a MonitorManager class object, as well as objects for the systems being monitored.

ALERTSONLY

Create only a MonitorManager class object. Do not create any objects

for the systems being monitored, but record alerts that affect these monitored systems and change the status of the MonitorManager class object to reflect these alerts.

PORT

For Windows/NT only

Specifies the TCP/IP Sockets PORT number associated with the NetFinity Agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined for the agent in the Windows/NT Services file. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6751 is used as the default value for PORT.

REMOTE

For OS/2 only

The NetView network ID and domain in which this NetFinity service point resides. Code REMOTE if the service point uses an SSCP-PU session. If you code REMOTE, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. The ID can be up to 8 alphanumeric characters and is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. The domain name can be up to 5 alphanumeric characters. The domain name is used as the LU parameter on the RMTCMD command.

SP Specifies the name of the NetFinity service point that is managing your network. If you specify an SNA cross domain or cross network service point, separate the names with periods with no intervening blanks.

sp_netid

For OS/2 only

The network identifier of the network in which the service point is located. *sp_netid* is an optional 1- to 8-character alphanumeric name.

If you specify *sp_netid*, you must also specify *sp_domain*. The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

For OS/2 only

The name of the NetView domain in which the service point resides. *sp_domain* is an optional 1- to 5-character alphanumeric name.

If you do not specify *sp_domain*, the command processor uses the local domain in which MultiSystem Manager resides. If you specify *sp_netid*, you must also specify *sp_domain*. *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point. The *sp_name* is required.

If the NetFinity connection is LU 6.2, this is the LU name. If the NetFinity connection is SSCP-PU, this is the PU name. It must be a 1- to 8-character alphanumeric name. The name can contain the special characters #, @, or \$ and cannot start with a number.

If the NetFinity connection is IP, this is the host name preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique.

TRACE

Determines whether to trace this GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced, along with trace information related to the specified trace value. The FLC003I message is displayed in your NetView host window and the command response window, and is also stored in the NetView message log.

If you issue this GETTOPO command from an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should use these parameters only to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the MultiSystem Manager command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. To display the trace messages as they are

issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

The topology information for the NetFinity managing system must be stored in RODM. If it is not, you can issue a GETTOPO NFRES or a GETTOPO NFONLY command for this service point. You can also include a GETTOPO NFRES or GETTOPO NFONLY statement for this service point in the MultiSystem Manager initialization file and re-issue the INITTOPO command.

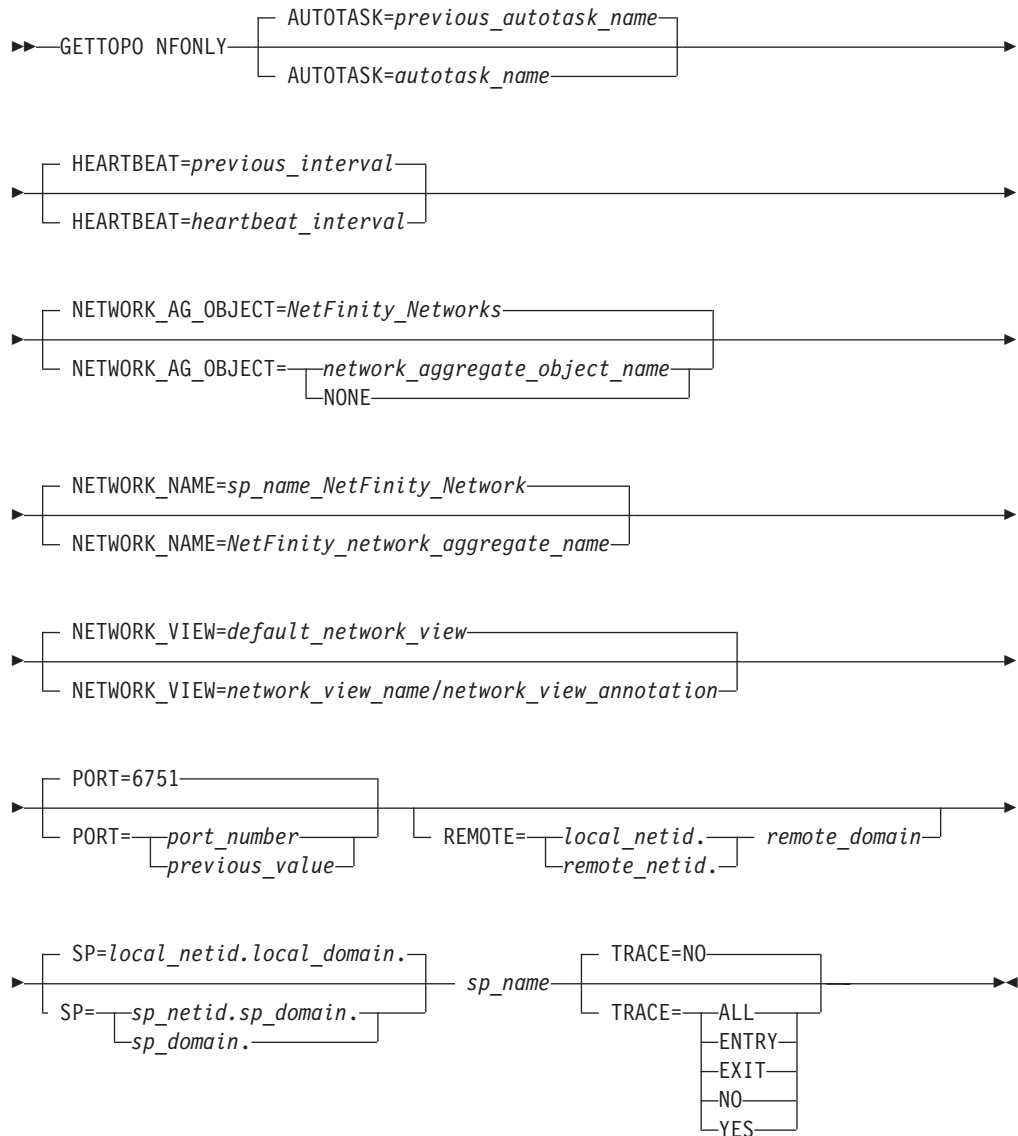
Examples

Following is an example of a GETTOPO NFGRP command:

```
GETTOPO NFGRP AUTOTASK=NETOP1 NFGROUP="Building 305" NFPROCESS=ALL NFSECURITY=NO  
NFSYSMONITOR=ALERTSONLY SP=NTB7P112
```

GETTOPO NFOONLY

Syntax



- The PORT keyword is valid for Windows/NT only.
- The REMOTE keyword is valid for OS/2 only.
- To specify a TCP/IP address, code an ampersand (&) as the first character of *sp_name*. For example:
SP=&smith.raleigh.ibm.com

Code either the host name or the fully-qualified name.

- For SP, *sp_netid* and *sp_domain* are only valid for OS/2.

Purpose of Command

The GETTOPO NFOONLY command retrieves status information for a NetFinity agent and adds it to RODM. If there are any resources underneath the network aggregate object from a previous GETTOPO command, they will be removed from RODM.

Operand Descriptions

AUTOTASK

Specifies the name of the autotask that MultiSystem Manager should use to communicate with the service point. This is the autotask from which the topology request is issued. The autotask name must be 1–8 alphanumeric characters and can contain the special characters #, @, and \$. It cannot start with a number.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time, in minutes, between HEARTBEAT requests to the service point. Valid values are 0–3599. HEARTBEAT requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The HEARTBEAT request can result in notification that topology or status changes have occurred.

If you do not code this keyword, MultiSystem Manager uses the last value you specified on a previous GETTOPO request. If you have never specified a HEARTBEAT value, MultiSystem Manager uses the default value of 0.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, the default is used.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

Unless you specify NONE, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the resource in the view. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this network. This name is stored in the DisplayResourceName field in RODM for this network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ; ? ' - _ % * < and >.

If the aggregate object representing this network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:

```
NETWORK_VIEW=My_View/Joe's own view
```

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values.

If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network view name and annotation specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

NFONLY

Retrieves and stores in RODM status and topology information only for the specified NetFinity service point.

PORT

For Windows/NT only

Specifies the TCP/IP Sockets PORT number associated with the NetFinity Agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined for the agent in the Windows/NT "Services" file. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6751 is used as the default value for PORT.

REMOTE

For OS/2 only

The NetView network ID and domain in which this NetFinity service point resides. Code REMOTE if the service point uses an SSCP-PU session. If you code REMOTE, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. The ID can be up to 8 alphanumeric characters and is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. The domain name can be up to 5 alphanumeric characters. The domain name is used as the LU parameter on the RMTCMD command.

SP Specifies the name of the NetFinity service point that is managing your network. If you specify an SNA cross domain or cross network service point, separate the names with periods with no intervening blanks.

sp_netid

For OS/2 only

The network identifier of the network in which the service point is located. *sp_netid* is an optional 1- to 8-character alphanumeric name.

If you specify *sp_netid*, you must also specify *sp_domain*. The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

For OS/2 only

The name of the NetView domain in which the service point resides. *sp_domain* is an optional 1- to 5-character alphanumeric name.

If you do not specify *sp_domain*, the command processor uses the local domain in which MultiSystem Manager resides. If you specify *sp_netid*,

you must also specify *sp_domain*. *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point. The *sp_name* is required.

If the NetFinity connection is LU 6.2, this is the LU name. If the NetFinity connection is SSCP-PU, this is the PU name. It must be a 1- to 8-character alphanumeric name. The name can contain the special characters #, @, or \$ and cannot start with a number.

If the NetFinity connection is IP, this is the host name preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique.

TRACE

Determines whether to trace this GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed in your NetView host window and the command response window, and is also stored in the NetView message log.

If you issue this GETTOPO command from an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should use these parameters only to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the MultiSystem Manager command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. To display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

Examples

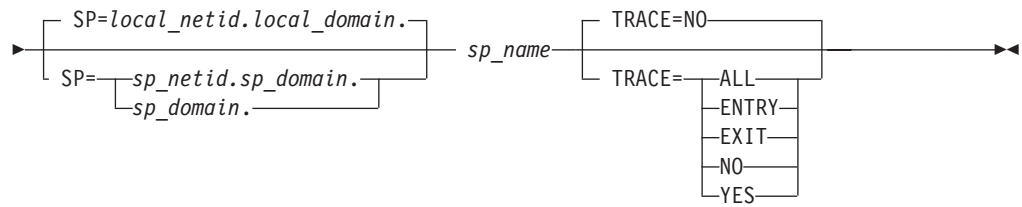
Following is an example of a GETTOPO NFOONLY command:

```
GETTOPO NFOONLY AUTOTASK=NETOP1 NETWORK_NAME=NTB7P112_Special_Projects  
SP=NTB7P112
```

GETTOPO NFRES

Syntax





- The `PORT` keyword is valid for Windows/NT only.
- The `REMOTE` keyword is valid for OS/2 only.
- To specify a TCP/IP address, code an ampersand (&) as the first character of `sp_name`. For example:
`SP=&smith.raleigh.ibm.com`

Code either the host name or the fully-qualified name.

- For `SP`, `sp_netid` and `sp_domain` are only valid for OS/2.

Purpose of Command

The `GETTOPO NFRES` command retrieves and stores in RODM topology and status information for the specified NetFinity service point and its managed resources.

Operand Descriptions

AUTOTASK

Specifies the name of the autotask that MultiSystem Manager should use to communicate with the service point. This is the autotask from which the topology request is issued. The autotask name must be 1–8 alphanumeric characters and can contain the special characters #, @, and \$. It cannot start with a number.

If `AUTOTASK` is not specified, MultiSystem Manager uses the autotask specified on a previous `GETTOPO` command to this service point. If this is the initial `GETTOPO` command being issued to this service point and `AUTOTASK` is not specified, MultiSystem Manager uses the autotask specified in the `(MSM)function.autotask.MSMdefault` statement in `CNMSTYLE`.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next `GETTOPO` command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time between `HEARTBEAT` requests to the service point. Valid values are 0–3599 (minutes) or `AON`. When `AON` is specified, the `AON` automation policy for the MultiSystem Manager agent at the service point is queried to retrieve the time interval.

If you do not code this keyword, MultiSystem manager uses the last value specified on a previous `GETTOPO` request. If `AON` was specified and there was no value set in the automation policy or if you have never specified a `HEARTBEAT` value, MultiSystem Manager uses the default value of zero.

`HEARTBEAT` requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The `HEARTBEAT` request can result in notification that topology or status changes have occurred.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, the default is used.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

Unless you specify NONE, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the resource in the view. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this network. This name is stored in the DisplayResourceName field in RODM for this network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ? ' - _ % * < and >.

If the aggregate object representing this network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:

```
NETWORK_VIEW=My_View/Joe's own view
```

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values.

If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network view name and annotation specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

NFPROCESS

Specifies whether to create a ProcessManager class object, as well as objects for the processes that it manages.

NO Do not create a ProcessManager class object. Ignore all alerts that would affect this object.

ALL Create a ProcessManager class object, as well as objects for each process being managed.

ALERTSONLY

Create only a ProcessManager class object. Do not create any objects for the processes being managed, but record alerts that affect these managed processes and change the status of the ProcessManager class object to reflect these alerts.

NFRES

Retrieves and stores in RODM topology and status information for the specified NetFinity workstation and the resources it manages.

NFRESET

Specifies if you want to force the MultiSystem Manager NetFinity agent to query NetFinity for current topology or if you want the previous topology to remain. The MultiSystem Manager NetFinity agent maintains an internal database of the topology of the NetFinity network. If the NFRESET parameter is implemented ("yes" is specified), the agent will be forced to discard the topology database and query NetFinity for current topology information. Depending on the size of your NetFinity network, this parameter can impact performance of the GETTOPO command.

NO Do not force recollection of the topology.

YES Force recollection of the topology.

NFSECURITY

Specifies whether to create a SecurityMonitor class object. Alerts are forwarded to this object.

- NO** Do not create a SecurityMonitor class object. Ignore alerts that would normally affect this object.
- YES** Create a SecurityMonitor class object. Change the status of the object as appropriate to reflect alerts received by the object.

NFSYSMONITOR

Specifies whether to create a MonitorManager class object, as well as objects for the systems it monitors.

- NO** Do not create a MonitorManager class object. Ignore all alerts that would affect this object.
- ALL** Create a MonitorManager class object, as well as objects for the systems being monitored.

ALERTSONLY

Create only a MonitorManager class object. Do not create any objects for the systems being monitored, but record alerts that affect these monitored systems and change the status of the MonitorManager class object to reflect these alerts.

PORT

For Windows/NT only

Specifies the TCP/IP Sockets PORT number associated with the NetFinity Agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined for the agent in the Windows/NT "Services" file. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6751 is used as the default value for PORT.

REMOTE

For OS/2 only

The NetView network ID and domain in which this NetFinity service point resides. Code REMOTE if the service point uses an SSCP-PU session. If you code REMOTE, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. The ID can be up to 8 alphanumeric characters and is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. The domain name can be up to 5 alphanumeric characters. The domain name is used as the LU parameter on the RMTCMD command.

SP Specifies the name of the NetFinity service point that is managing your

network. If you specify an SNA cross domain or cross network service point, separate the names with periods with no intervening blanks.

sp_netid

For OS/2 only

The network identifier of the network in which the service point is located. *sp_netid* is an optional 1- to 8-character alphanumeric name.

If you specify *sp_netid*, you must also specify *sp_domain*. The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

For OS/2 only

The name of the NetView domain in which the service point resides. *sp_domain* is an optional 1- to 5-character alphanumeric name.

If you do not specify *sp_domain*, the command processor uses the local domain in which MultiSystem Manager resides. If you specify *sp_netid*, you must also specify *sp_domain*. *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point. The *sp_name* is required.

If the NetFinity connection is LU 6.2, this is the LU name. If the NetFinity connection is SSCP-PU, this is the PU name. It must be a 1- to 8-character alphanumeric name. The name can contain the special characters #, @, or \$ and cannot start with a number.

If the NetFinity connection is IP, this is the host name preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique.

TRACE

Determines whether to trace this GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed in your NetView host window and the command response window, and is also stored in the NetView message log.

If you issue this GETTOPO command from an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should use these parameters only to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT

Display message FLC003I containing the return code value when exiting each module.

NO

Do not display trace messages while processing this topology request.

YES

Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the MultiSystem Manager command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. To display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

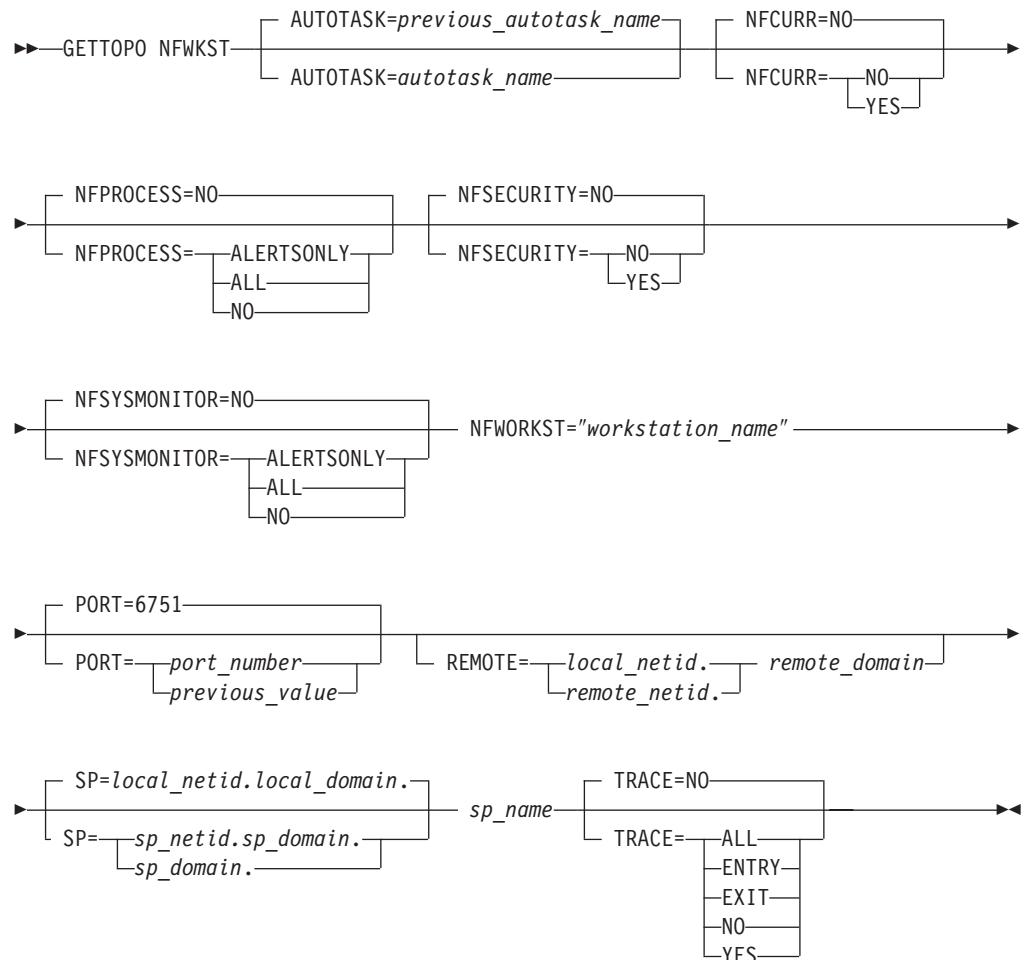
Examples

Following is an example of a GETTOPO NFRES command:

```
GETTOPO NFRES AUTOTASK=NETOP1 NETWORK_NAME=NTB7P112_Special_Project  
NFPROCESS=ALL NFSECURITY=NO NFSYSMONITOR=ALERTSONLY SP=NTB7P112
```


GETTOPO NFWKST

Syntax



- The PORT keyword is valid for Windows/NT only.
- The REMOTE keyword is valid for OS/2 only.
- To specify a TCP/IP address, code an ampersand (&) as the first character of *sp_name*. For example:
SP=&smith.raleigh.ibm.com

Code either the host name or the fully-qualified name.

- For SP, *sp_netid* and *sp_domain* are only valid for OS/2.

Purpose of Command

The GETTOPO NFWKST command retrieves and stores in RODM topology and status information for the specified NetFinity workstation.

Operand Descriptions

AUTOTASK

Specifies the name of the autotask that MultiSystem Manager should use to

communicate with the service point. This is the autotask from which the topology request is issued. The autotask name must be 1–8 alphanumeric characters and can contain the special characters #, @, and \$. It cannot start with a number.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

NFCURR

Specifies whether to use the previous or the default GETTOPO NFWKST parameter values for the workstation.

NO Use the default parameter values for any GETTOPO NFWKST parameters not specified.

YES Use the previously specified parameter values for any GETTOPO NFWKST parameters not specified.

NFPROCESS

Specifies whether to create a ProcessManager class object, as well as objects for the processes that it manages.

NO Do not create a ProcessManager class object. Ignore all alerts that would affect this object.

ALL Create a ProcessManager class object, as well as objects for each process being managed.

ALERTSONLY

Create only a ProcessManager class object. Do not create any objects for the processes being managed, but record alerts that affect these managed processes and change the status of the ProcessManager class object to reflect these alerts.

NFSECURITY

Specifies whether to create a SecurityMonitor class object. Alerts are forwarded to this object.

NO Do not create a SecurityMonitor class object. Ignore alerts that would normally affect this object.

YES Create a SecurityMonitor class object. Change the status of the object as appropriate to reflect alerts received by the object.

NFSYSMONITOR

Specifies whether to create a MonitorManager class object, as well as objects for the systems it monitors.

NO Do not create MonitorManager class objects. Ignore all alerts that would affect these objects.

ALL Create a MonitorManager class object, as well as objects for the systems it monitors.

ALERTSONLY

Create only a MonitorManager class object. Do not create any objects

for the systems that it monitors, but record alerts that affect these monitored systems and change the status of the ProcessManager class object to reflect these alerts.

NFWKST

Retrieves and stores in RODM topology and status information for the specified NetFinity workstation.

NFWORKST

Specifies the name of the NetFinity workstation for which to retrieve and store topology and status information. Enclose the workstation name in double quotes ("").

PORT

For Windows/NT only

Specifies the TCP/IP Sockets PORT number associated with the NetFinity Agent. PORT can be specified only if the transport protocol is IP.

The PORT number specified must be the same number as the PORT number defined for the agent in the Windows/NT "Services" file. If PORT is not specified, and the transport protocol is IP, MultiSystem Manager uses the PORT number specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and the value specified for the SP keyword was coded beginning with an ampersand (&), 6751 is used as the default value for PORT.

REMOTE

For OS/2 only

The NetView network ID and domain in which this NetFinity service point resides. Code REMOTE if the service point uses an SSCP-PU session. If you code REMOTE, NetView uses RMTCMD to send the RUNCMD to the remote NetView.

remote_netid

The network identifier of the remote NetView domain in which the service point is located. The ID can be up to 8 alphanumeric characters and is optional. If you code *remote_netid*, separate it from *remote_domain* by a period, with no intervening blanks.

The *remote_netid* variable is used as the NETID parameter on the RMTCMD command. If you do not specify *remote_netid*, MultiSystem Manager uses the network identifier of the network in which it resides (*local_netid*).

remote_domain

The name of the remote NetView domain where this service point resides. The domain name can be up to 5 alphanumeric characters. The domain name is used as the LU parameter on the RMTCMD command.

SP Specifies the name of the NetFinity service point that is managing your network. If you specify an SNA cross domain or cross network service point, separate the names with periods with no intervening blanks.

sp_netid

For OS/2 only

The network identifier of the network in which the service point is located. *sp_netid* is an optional 1- to 8-character alphanumeric name.

If you specify *sp_netid*, you must also specify *sp_domain*. The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

For OS/2 only

The name of the NetView domain in which the service point resides. *sp_domain* is an optional 1- to 5-character alphanumeric name.

If you do not specify *sp_domain*, the command processor uses the local domain in which MultiSystem Manager resides. If you specify *sp_netid*, you must also specify *sp_domain*. *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the SNA or IP service point. The *sp_name* is required.

If the NetFinity connection is LU 6.2, this is the LU name. If the NetFinity connection is SSCP-PU, this is the PU name. It must be a 1- to 8-character alphanumeric name. The name can contain the special characters #, @, or \$ and cannot start with a number.

If the NetFinity connection is IP, this is the host name preceded by an ampersand (&). The *sp_name* can be a simple host name (for example, &smith), or a fully-qualified host name (for example, &smith.raleigh.ibm.com). The simple host name must be unique. You cannot have two simple host names that are the same, even if they reside in different TCP/IP domains. For example, you cannot have &smith.raleigh.ibm.com and &smith.austin.ibm.com because the host name portion of the fully-qualified IP address is not unique.

TRACE

Determines whether to trace this GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed in your NetView host window and the command response window, and is also stored in the NetView message log.

If you issue this GETTOPO command from an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should use these parameters only to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the MultiSystem Manager command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. To display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list that is not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

Examples

Following is an example of a GETTOPO NFWKST command:

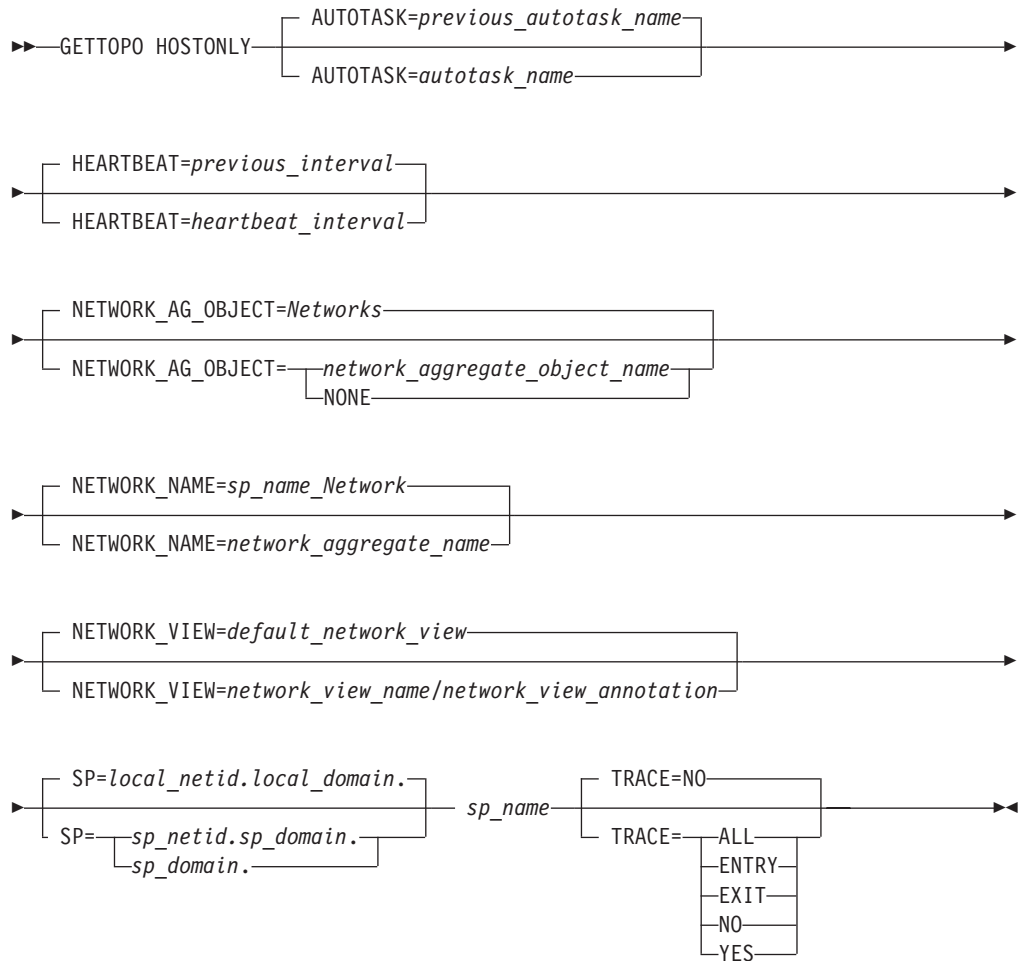
```
GETTOPO NFWKST AUTOTASK=NETOP1 NFWORKST="Group6_Dev1"  
NFPROCESS=ALL NFSYSMONITOR=ALERTSONLY SP=NTB7P112
```

Chapter 10. MultiSystem Manager Open Commands

This section contains all MultiSystem Manager Open commands, which are listed in alphabetical order.

GETTOPO HOSTONLY

Syntax



Purpose of Command

The `GETTOPO HOSTONLY` command retrieves status information for an Open topology agent and adds it to RODM. If there are any resources underneath the network aggregate object from a previous `GETTOPO` command, they will be removed from RODM.

Operand Descriptions

HOSTONLY

Retrieve and add to RODM the status for only the specified Open topology agent. Do not retrieve and store status for the resources managed by the Open topology agent.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the service point. This is the autotask from which the

topology request is issued. The autotask name must be 1–8 alphanumeric characters and can contain the special characters #, @, and \$. It cannot start with a number.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time, in minutes, between HEARTBEAT requests to the service point. Valid values are 0–3599. HEARTBEAT requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The HEARTBEAT request may result in notification that topology or status changes have occurred.

If you do not code this keyword, MultiSystem Manager uses the last value you specified on a previous GETTOPO request. If you have never specified a HEARTBEAT value, MultiSystem Manager uses the default value of zero.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, the default is used.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

Unless you specify NONE, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the resource in the view. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this network. This name is stored in the DisplayResourceName field in RODM for this network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ; ? ' - _ % * < and >.

If the aggregate object representing this network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:

```
NETWORK_VIEW=My_View/Joe's own view
```

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network view name and annotation specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHSP Programmer's Guide* for more information about network view name and annotation.

SP The fully-qualified name of the LU or PU service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

The network identifier of the SNA network in which the service point is located. *sp_netid* is an optional 1- to 8-character alphanumeric name.

If you specify *sp_netid*, you must also specify *sp_domain*. The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

The name of the NetView domain in which the service point resides. *sp_domain* is an optional 1- to 5-character alphanumeric name.

If you do not specify *sp_domain*, the RUNCMD processor uses the local domain in which MultiSystem Manager resides. If you specify *sp_netid*, you must also specify *sp_domain*. *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the service point associated with the NetView for AIX being described. The NetView for AIX connection is LU 6.2, so this is the LU name.

Use of this 1- to 8-character alphanumeric name is required. The name is used as the SP parameter on RUNCMDs sent to the service point. The name can contain the special characters #, @, or \$ and cannot start with a number.

TRACE

Determines whether to trace this GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed in your NetView host window and the command response window, and is also stored in the NetView message log.

If you issue this GETTOPO command from an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should use these parameters only to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs or RMTCMDs issued when processing the GETTOPO command.

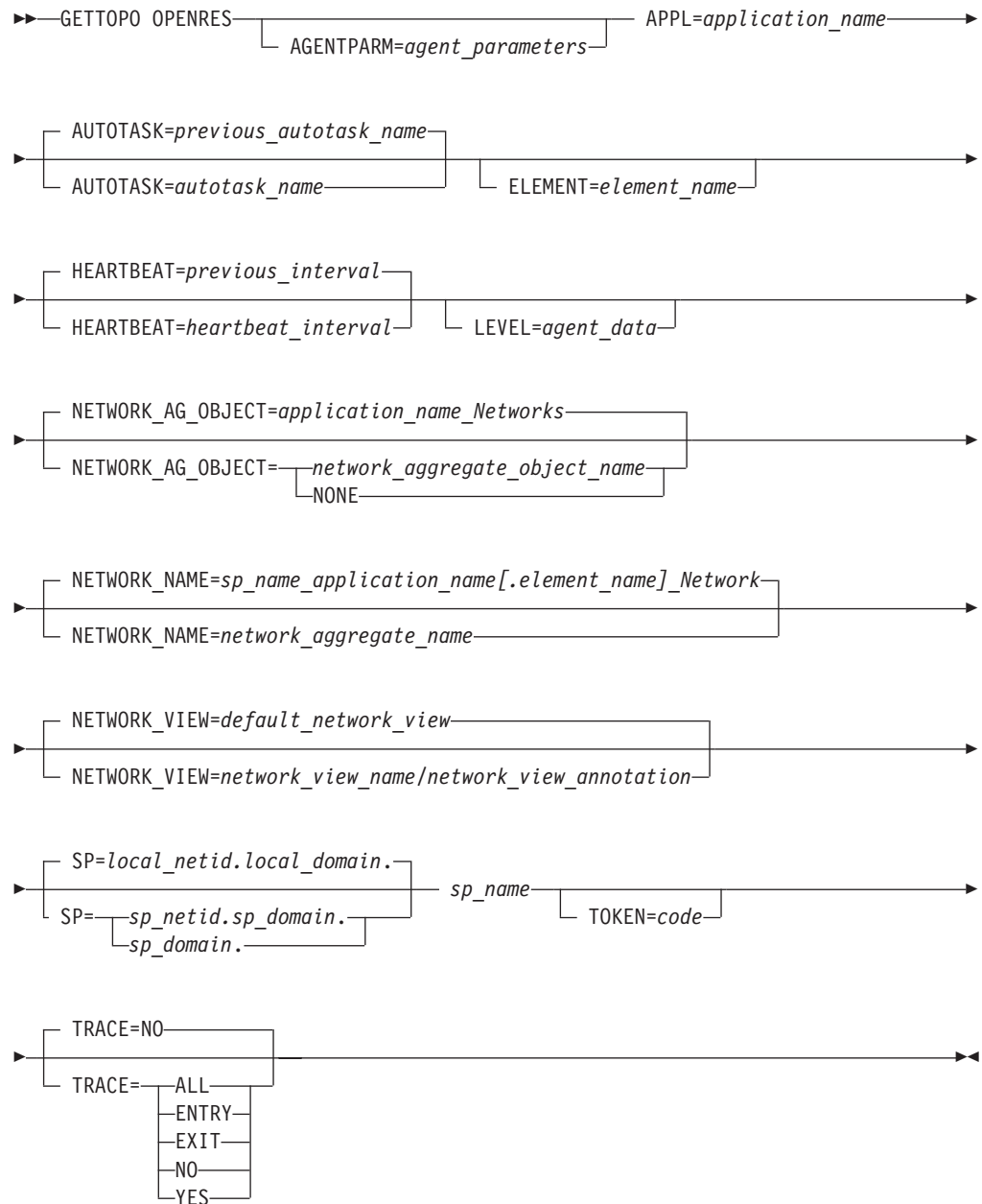
If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the MSM command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. To display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

GETTOPO OPENRES

Syntax



Purpose of Command

The GETTOPO OPENRES command retrieves and adds to RODM, the topology and status information for a managed resource, as defined in the documentation for the related Open topology agent.

Operand Descriptions

AGENTPARM

Additional parameters being sent to the agent. Do not include commas (,), equal signs (=), or blanks in the parameter string.

APPL

The name of the Open topology agent to which you are sending this GETTOPO command.

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

ELEMENT

The name of the sub-application, or element manager, to which you are sending this GETTOPO command if the Open topology agent supports sub-applications.

When you use the command facility to issue this GETTOPO command, MultiSystem Manager uses the sub-application or element manager name that is in RODM, if one exists.

HEARTBEAT

The amount of time, in minutes, between heartbeat requests to the service point. Heartbeat requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The heartbeat request may result in notification that topology or status changes have occurred.

Specify a zero (0) to stop current heartbeat requests for the service point.

If the HEARTBEAT keyword is not coded, it defaults to the current heartbeat interval. If an interval was never set, MultiSystem Manager sets it to zero.

Value: An integer from 0–3599. A value of zero means that no heartbeat requests are issued.

Default:

The previous value.

LEVEL

Specific data required by the Open topology agent.

When you use the command facility to issue this GETTOPO command, if you do not enter values for this field, MultiSystem Manager uses the data from a previous GETTOPO OPENRES command, if any was specified.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses *application_name_Networks*, replacing *application_name* with the value you specified for the APPL= keyword.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

If you code this name or accept the default name, the objects representing this network and their status are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the object on the screen. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing the network. This name is stored in the DisplayResourceName field in RODM for this network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () : ; ? ' - _ & + % * < and >.

If the aggregate object representing this network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses *sp_name_application_name[element_name]_Network* with the following substitutions:

sp_name

The value specified for *sp_name* in the SP= keyword.

application_name

The value specified for the APPL= keyword.

element_name

The value specified for the ELEMENT= keyword, if it is coded.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation* and they must be separated by a right slash (/). For example:

```
NETWORK_VIEW=My_View/Joe's own view
```

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default `network_view` name and annotation.

network_view_name

The name of your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the **Description** field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

|
| If you do not specify NETWORK_VIEW, MultiSystem Manager uses the name
| specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in
| CNMSTYLE.

SP The fully-qualified name of the LU or PU service point that is managing your workgroup.

If you specify more than one variable, separate them with periods, with no intervening blanks.

sp_netid

The network identifier of the SNA network in which the service point is located. Use of this 1–8 alphanumeric character *sp_netid* is optional. If you specify *sp_netid*, you must also specify *sp_domain*.

The *sp_netid* is used as the NETID parameter on RUNCMDs sent to the service point. If you do not specify *sp_netid*, the RUNCMD command processor looks for the target service point on the local network where MultiSystem Manager resides.

sp_domain

The name of the NetView domain in which the service point resides. Use of the 1–5 alphanumeric character *sp_domain* is optional, if *sp_netid* is not specified. If you do not specify *sp_domain*, MultiSystem Manager uses the local domain in which MultiSystem Manager resides.

When *sp_netid* is specified, *sp_domain* is used to create an SNA_Domain_Class object in RODM. *sp_domain* is not used on the RUNCMD command.

sp_name

The name of the service point associated with the network being described. If the service point is connected through an SSCP-PU

session, this is the name by which the PU is known to VTAM. If it is connected through LU 6.2, this is the LU name.

Use of the 1–8 alphanumeric character *sp_name* is required. It is used as the SP parameter on RUNCMDs sent to the service point. MultiSystem Manager permits *sp_name* to include only these special characters: #, @, or \$. MultiSystem Manager expects it to start with an alphabetic character, #, @, or \$.

TOKEN

A character string required by the Open topology agent.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Customer Support.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related RUNCMDs issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command tree facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list not invoked from NMC.

Usage Notes

The topology manager must be enabled for MultiSystem Manager to process this command.

See the INITTOPO command for initialization information, and the RESTOPO command for information about resuming the processing of GETTOPO commands if they have been suspended by the SUSPTOPO command.

Examples

The following GETTOPO OPENRES example:

- Retrieves topology and status for segment 018B connected to the service point with a PU name of NTB6PU03.
- Sends this GETTOPO command to the Open topology agent MYAGENT's sub-application named AGENTAPP.
- Passes the parameter PARM1 to the Open topology agent.
- Uses the MYAUTO autotask to communicate with the workstation.
- Sets the heartbeat interval to 10 minutes.
- Groups NTB6PU03's objects under the aggregate object *OPEN_Networks*.
- Names the aggregate object representing the network *My_OPEN_Network*.
- Names the view *My_View* with a description of *NTB6PU03's view*.
- Sends the character string ~CS to the Open topology agent.
- Returns trace messages for the command.

```
GETTOPO OPENRES APPL=MYAGENT ELEMENT=AGENTAPP
AGENTPARM=PARM1 AUTOTASK=MYAUTO HEARTBEAT=10
LEVEL=SEGMENT;018B SP=NTB6PU03 NETWORK_AG_OBJECT=OPEN_Networks
NETWORK_NAME=My_OPEN_Network NETWORK_VIEW=My_View/NTB6PU03's view
TOKEN=~CS TRACE=YES
```

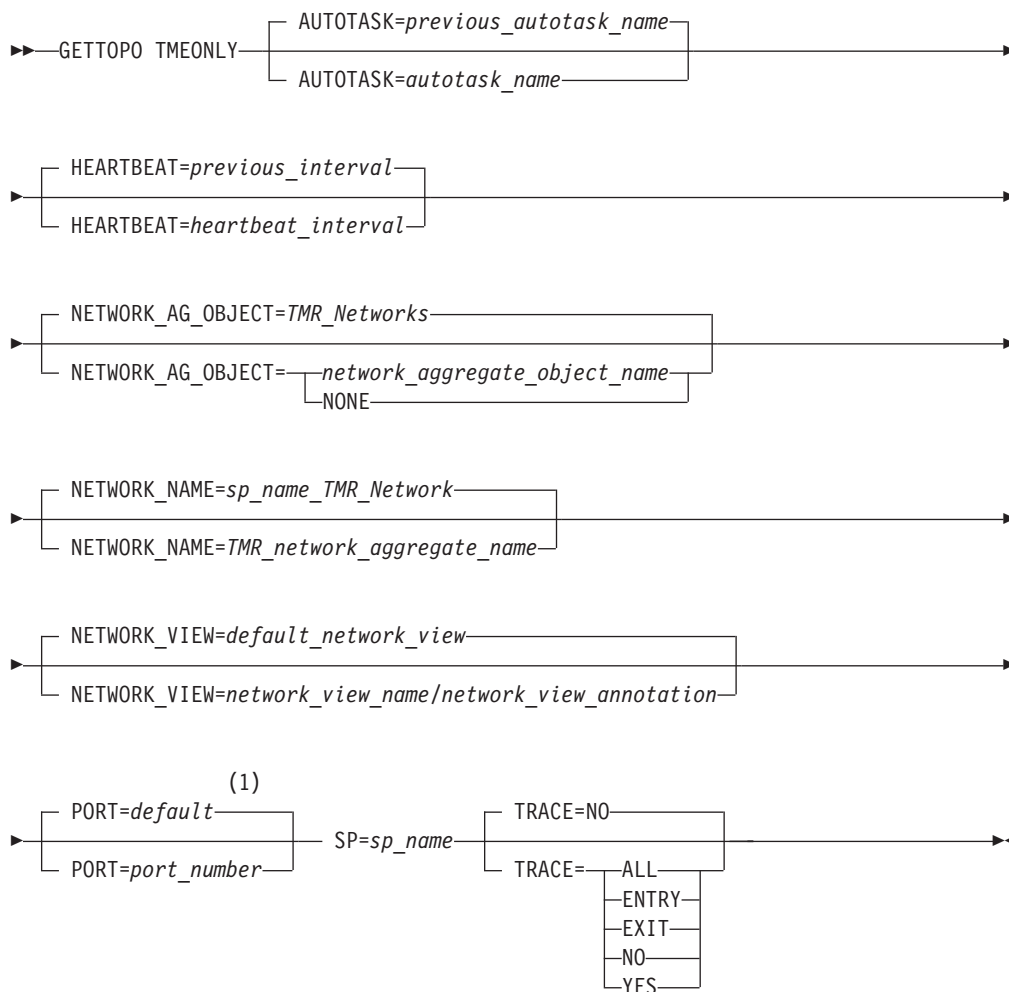
Refer to the sample initialization file FLCSIOPN that you received with MultiSystem Manager for an example GETTOPO statement coded in an initialization file.

Chapter 11. MultiSystem Manager TMR Commands

This section contains all MultiSystem Manager TMR commands, which are listed in alphabetical order.

GETTOPO TMEONLY

Syntax



Notes:

- 1 The default is either 3333 or the previous value.

Purpose of Command

This statement describes specific information about a TMR system that is managing a portion of your enterprise. Code one GETTOPO TMEONLY statement for each TMR agent.

If there are any resources below the network aggregate object from a previous GETTOPO command, they will be removed in RODM.

Operand Descriptions

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8

alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

HEARTBEAT

The amount of time, in minutes, between heartbeat requests to the service point. Heartbeat requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The heartbeat request may result in notification that topology or status changes have occurred.

Specify a zero (0) to stop current heartbeat requests for the collection point.

If the HEARTBEAT keyword is not coded, it defaults to the current heartbeat interval. If an interval was never set, MultiSystem Manager sets it to zero.

Value: An integer from 0–3599. A value of zero means that no heartbeat requests are issued.

Default:

Previous value.

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the syntax diagram.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

If you code this name or accept the default name, the objects representing this network and their statuses are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the object on the screen. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this IP network. This name is stored in the DisplayResourceName field in RODM for this IP network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; ? ' - _ % * < and >.

If the aggregate object representing this IP network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:
NETWORK_VIEW=My_View/Joe's own view.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network_view name and annotation.

network_view_name

The name for your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the Description field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

If you do not specify NETWORK_VIEW, MultiSystem Manager uses the name specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

PORT

Specifies the TCP/IP sockets PORT number associated with the TMR Topology Service agent. This must match the value specified by the agent parameter in the **MSMAgent.cfg** file in the MSMAgent install directory.

If PORT is not specified, MultiSystem Manager uses the PORT specified on the previous GETTOPO command to this service point. If this is the initial

GETTOPO command being issued to this service point, and PORT is not specified, 3333 is used as the default PORT number.

SP The TCP/IP host name where the TMR Topology Service agent resides. If a host name is used, only the portion needed by the TCP/IP name service has to be included.

TMEONLY

Retrieve and add to RODM the topology data and status for the specified TMR Topology Service agent only. Do not retrieve and store topology data and status for the resources managed by TMR.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Service.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related commands issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list not invoked from NMC.

Examples

The following GETTOPO TMEONLY example:

- Retrieves topology and status for the TMR network managed by the agent COLUMBO.
- Uses the AUTOTME1 autotask to communicate with the TMR Topology Service agent.
- Uses the previous heartbeat interval by taking the default for the HEARTBEAT keyword.

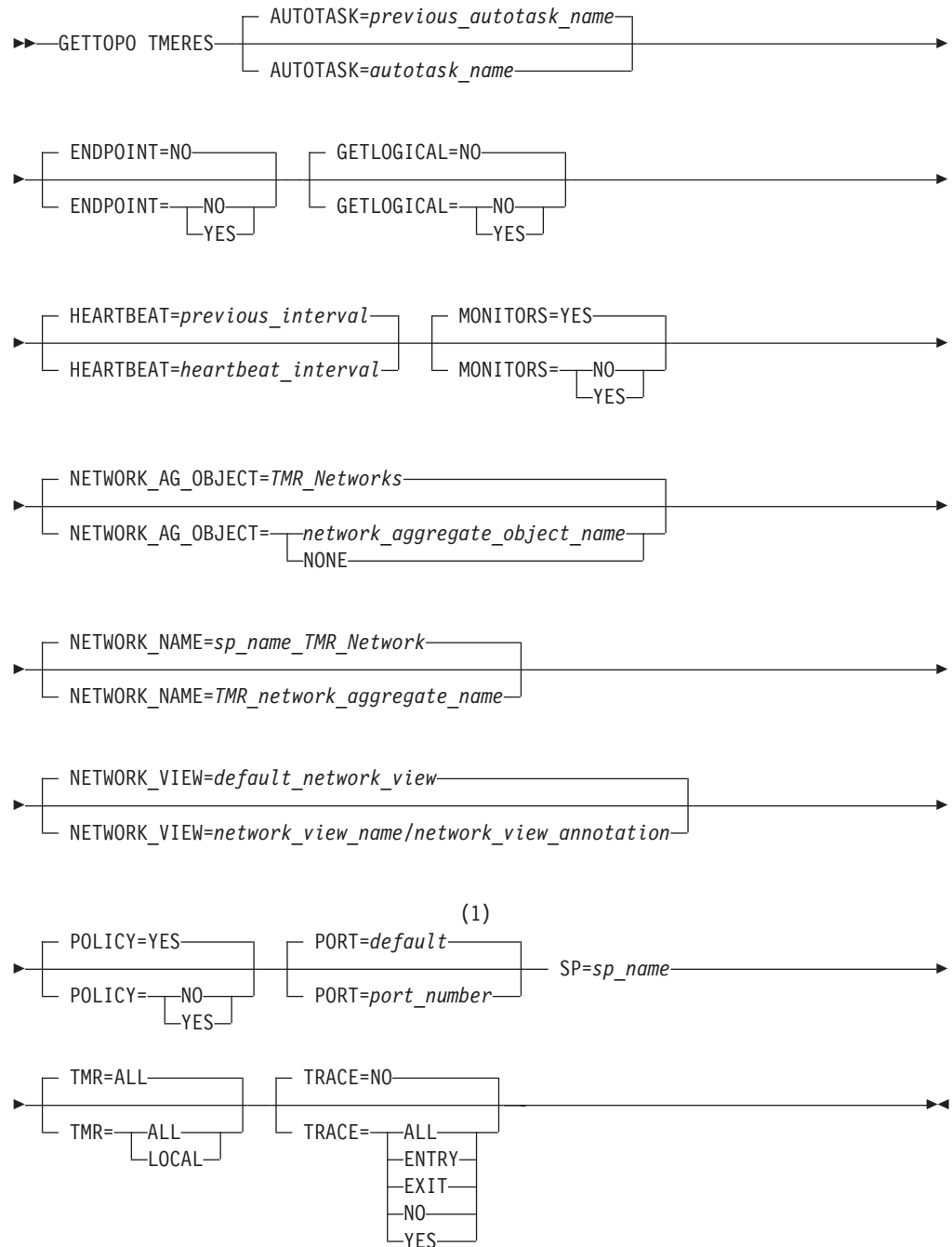
- Does not include the objects representing the network in an aggregate object (NETWORK_AG_OBJECT=NONE)
- Names the view *My_View* with a description of *Regina's own view*.
- Does not display trace messages by taking the default for the TRACE keyword.
- Uses the default port number.

```
GETTOPO TMEONLY AUTOTASK=AUTOTME1,  
NETWORK_AG_OBJECT=NONE NETWORK_VIEW=My_View/Regina's own view,  
SP=columbo
```

Refer to the sample initialization file named *FLCSITME*, which you received with MultiSystem Manager for an example GETTOPO statement coded in an initialization file.

GETTOPO TMERES

Syntax



Notes:

- 1 The default is either 3333 or the previous value.

Purpose of Command

This statement describes specific information about a TMR system that is managing a portion of your enterprise. Code one GETTOPO TMERES statement for each TMR Topology Service agent.

Operand Descriptions

AUTOTASK

The name of the autotask that MultiSystem Manager should use to communicate with the resource. This is the autotask from which the topology request is issued. MultiSystem Manager expects the autotask name to be 1–8 alphanumeric characters in length. It accepts only these special characters: #, @, and \$. It expects the name to start with an alphabetic character or #, @, or \$.

If AUTOTASK is not specified, MultiSystem Manager uses the autotask specified on a previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point and AUTOTASK is not specified, MultiSystem Manager uses the autotask specified in the (MSM)function.autotask.MSMdefault statement in CNMSTYLE.

Note: The autotask name is stored in RODM. If you rename an autotask in NetView, you must specify the new autotask name on the next GETTOPO command to establish it as the *previous_autotask_name*.

ENDPOINT

Specifies whether or not to display endpoint information.

Value: Yes or No

Default: No

GETLOGICAL

Specifies whether to retrieve TMR logical topology.

NO Retrieves only TMR physical topology. The data for logical views will not be obtained and will not be available for display. No is the default.

YES Retrieves TMR logical topology.

HEARTBEAT

The amount of time, in minutes, between heartbeat requests to the service point. Heartbeat requests are queries issued by MultiSystem Manager to ensure that the service point is still active. The heartbeat request may result in notification that topology or status changes have occurred.

Specify a zero (0) to stop current heartbeat requests for the collection point.

If the HEARTBEAT keyword is not coded, it defaults to the current heartbeat interval. If an interval was never set, MultiSystem Manager sets it to zero.

Value: An integer from 0–3599. A value of 0 means that no heartbeat requests are issued.

Default: Previous value.

MONITORS

Specifies whether or not to display TMR Distributed Monitoring monitors information gathered for each managed node with your topology data.

Value: Yes or No

Default: Yes

NETWORK_AG_OBJECT

Specifies whether you want the objects representing this network to be contained in an aggregate object.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the name of the network aggregate object already in RODM instead of the default value. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the syntax diagram.

network_aggregate_object_name

The name of the aggregate object that you want to contain the network manager and aggregate objects representing this network. The name can be 1–16 alphanumeric characters in length. You cannot use commas (,), blanks, percent signs (%), double quotes ("), or equal signs (=).

If you code this name or accept the default name, the objects representing this network and their statuses are collected in the aggregate.

The NETWORK_AG_OBJECT name is used as the DisplayResourceName of the object in RODM; therefore, it is the name of the object on the screen. Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about the network aggregate object name.

NONE

Do not include the objects representing this network in an aggregate object. If you specify NONE, the network manager object and the network aggregate object are both displayed in the network view.

NETWORK_NAME

The unique name that you want displayed on the screen for the aggregate object representing this IP network. This name is stored in the DisplayResourceName field in RODM for this IP network object.

The NETWORK_NAME can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ \$. () ; : ? ' - _ % * < and >.

If the aggregate object representing this IP network already exists in RODM, you do not need to code this keyword. If it already exists and you do not code the keyword, MultiSystem Manager uses the network name already defined in RODM instead of the default value. If the aggregate object does not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default value shown in the diagram.

NETWORK_VIEW

The name and description of the network level view in which you want the object representing the network resource to be displayed.

If you specify NETWORK_VIEW, you must also specify *network_view_name* and *network_view_annotation*, separated by a right slash (/). For example:
NETWORK_VIEW=My_View/Joe's own view.

If the objects representing this service point already exist in RODM, you do not need to code this keyword. If they already exist and you do not code the keyword, MultiSystem Manager uses the network view name and annotation in RODM instead of the default values. If the objects do not already exist in RODM and you do not code this keyword, MultiSystem Manager uses the default network_view name and annotation.

network_view_name

The name for your network view. This is the name that appears in the network view list in the window.

This name can be 1–32 alphanumeric characters in length. Only these special characters can be used: # @ % \$. () : ; ? ' " - _ & + < and >. The first character must be alphabetic or numeric.

network_view_annotation

The description of your network view. It is displayed in the Description field.

The description can be 1–32 alphanumeric characters in length. You can use all special characters, except the comma (,) and equal sign (=). Embedded blanks are also permitted.

Refer to the *Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for more information about network view name and annotation.

If you do not specify NETWORK_VIEW, MultiSystem Manager uses the name specified in the (MSM)COMMON.FLC_DEF_NETW_VIEW statement in CNMSTYLE.

POLICY

Specifies whether or not to display managed nodes under the policy regions in the view.

Value: Yes or No

Default: Yes

PORT

Specifies the TCP/IP sockets PORT number associated with the TMR Topology Service agent. This must match the value specified by the agent parameter in the **MSMAgent.cfg** file in the MSMAgent install directory.

If PORT is not specified, MultiSystem Manager uses the PORT specified on the previous GETTOPO command to this service point. If this is the initial GETTOPO command being issued to this service point, and PORT is not specified, 3333 is used as the default PORT number.

SP The TCP/IP host name where the TMR Topology Service agent resides. If a host name is used, only the portion needed by the TCP/IP name service has to be included.

TMERES

Retrieve and add to RODM the topology data and status for the specified TMR Topology Service agent and the resources that it manages.

TMR

Discover topology for either only the local TMR, or the local and all connected TMRs.

ALL Discover topology for the local TMR and all connected TMRs. ALL is the default.

LOCAL

Discover topology for only the local TMR.

TRACE

Determines whether to trace the MultiSystem Manager GETTOPO command.

If TRACE is enabled, message FLC003I is displayed. This message contains the name of the module being traced along with trace information related to the specified trace value. The FLC003I message is displayed at your host operator station task and command response window, and is stored in the NetView message log.

If you issue this GETTOPO command under an autotask, assigning message FLC003I to an OST enables you to view the trace messages as they are generated.

Note: Because the ENTRY, EXIT, and ALL parameters can generate many trace messages, you should only use these parameters to resolve network problems in coordination with Tivoli Service.

ALL Display message FLC003I for every trace point in each module. This includes trace information displayed for ENTRY, EXIT, and YES.

ENTRY

Display message FLC003I containing the parameter list passed to each module.

EXIT Display message FLC003I containing the return code value when exiting each module.

NO Do not display trace messages while processing this topology request.

YES Display message FLC003I and the related commands issued when processing the GETTOPO command.

Note: If you issue a GETTOPO command with TRACE=ALL, ENTRY, EXIT, or YES from the command facility, the system collects the trace messages and returns them to the command response window in one batch when the command completes. If you want to display the trace messages as they are issued, issue the GETTOPO command from the NetView command line or from a command list not invoked from NMC.

Examples

The following GETTOPO TMERES example:

- Retrieves topology and status for the TMR network managed by the agent COLUMBO.
- Uses the AUTOTME1 autotask to communicate with the TMR Topology Service agent.
- Specifies a heartbeat interval of 10 minutes.
- Places the view in the aggregate object *Tivoli_Networks*.
- Names the aggregate object representing the network *My_Tivoli_Network*.
- Names the view *My_View* with a description of *Joe's own view*.
- Displays trace messages for the GETTOPO command.
- Displays TMR monitors.
- Displays TMR policy regions.
- Uses the default port number.

```
GETTOPO TMERES AUTOTASK=AUTOTME1 HEARTBEAT=10,  
NETWORK_AG_OBJECT=Tivoli_Networks NETWORK_NAME=My_Tivoli_Network,  
NETWORK_VIEW=My_View/Joe's own view SP=columbo TRACE=YES MONITORS=YES  
POLICY=YES
```

Part 4. Command List NetView Commands

Chapter 12. Command List Commands	231
DSITSTAT (REXX)	232
DSIVSAM (PIPE)	236
DSIVSAM DEL (PIPE)	239
DSIVSAM GET (PIPE)	241
DSIVSAM GETREV (PIPE)	243
DSIVSAM INQUIRE (PIPE)	244
DSIVSAM PUT (PIPE)	247
DSIVSMX (PIPE)	248
DSIVSMX CLOSE (PIPE)	250
DSIVSMX DEL (PIPE)	251
DSIVSMX GET (PIPE)	253
DSIVSMX GETREV (PIPE)	255
DSIVSMX IDCAMS (PIPE)	256
DSIVSMX INQUIRE (PIPE)	257
DSIVSMX OPEN (PIPE)	260
DSIVSMX PUT (PIPE)	261
DUIFECMV (RODM)	262
FLUSHQ (REXX)	264
GETM Commands.	265
GETMLINE (REXX)	267
GETMPRES (REXX)	269
GETMSIZE (REXX)	272
GETMTFLG (REXX)	274
GETMTYPE (REXX)	276
GLOBALV (REXX)	281
GLOBALV DEF (REXX)	289
GLOBALV GET (REXX)	291
GLOBALV PURGE (REXX)	293
GLOBALV PUT (REXX)	295
GLOBALV RESTORE (REXX)	297
GLOBALV SAVE (REXX)	299
MSGREAD (REXX)	301
MSGROUTE (REXX)	304
PARSEL2R (REXX)	308
TRAP (REXX)	314
WAIT (REXX)	321
Sending Messages to the Operator Console	331
DOM (REXX)	333
WTO (REXX)	339
WTOR (REXX)	349

Chapter 12. Command List Commands

This section describes NetView commands that can be entered only from command lists or command processors written in a high-level language.

DSITSTAT (REXX)

Syntax

DSITSTAT

→ DSITSTAT taskname →

Purpose of Command

The DSITSTAT command is coded in a REXX procedure to retrieve resource statistics for every task in NetView. The REXX procedure can take action automatically or display data using WINDOW, VIEW, or messages. The statistics are returned in message BNH159I. If DSITSTAT is not issued within a command procedure, message DSI290I is returned.

Operand Descriptions

taskname

The name of the NetView task for which statistics are retrieved. If *taskname* is not specified, statistics are retrieved for all tasks.

Examples

Example: A DSITSTAT Command Response

This example shows a successful response to a DSITSTAT command:

Response:

```
BNH159I opername
taskname curcpu sesscpu maxcpu limcpu curget maxget limget slowget curmqi
sessmqi maxmqi limmqi curmqo sessmqo maxmqo limmqo curi/o sessi/o maxi/o limi/o
curpen sesspen pendpen totpen getkbm getsess g24kbm g24sess frekbm fresess
f24kbm f24sess pentime pentask
```

The BNH159I message is formatted without header lines and is in specific columns. This data is meant to be processed by a user-written REXX procedure. The output is one (non-MLWTO) line per task. All numbers are expressed in decimals with no punctuation. All data is aligned in columns to assist in parsing the data using a REXX procedure. Other than the *opername* and *taskname*, no columns are blank. The SUBSTR() function is recommended when this data is parsed. Table 5 explains the output of a DSITSTAT command:

Table 5. Output of a DSITSTAT Command

Token	Position	Description
BNH159I	1-7	The message ID for this successful response.
	8	Blank
<i>opername</i>	9-16	The operator name or task ID (TVBOPID) of an active task.
	17	Blank
<i>taskname</i>	18-25	The LU name or task ID (TVBLUNAM) of an active task.

Table 5. Output of a DSITSTAT Command (continued)

Token	Position	Description
	26	Blank
<i>curcpu</i>	27-30	The current task utilization in 0.01% units. The range is 0–9999.
	31	Blank
<i>sesscpu</i>	32-35	The task utilization for the entire session in 0.01% units. The range is 0–9999.
	36	Blank
<i>maxcpu</i>	37-40	The highest task utilization for the entire session in 0.01% in units. The range is 0–9999.
	41	Blank
<i>limcpu</i>	42-45	The applicable CPU limit from the OVERRIDE or DEFAULTS command in 0.01% units. The range is 0–9999.
	46	Blank
<i>curget</i>	47-52	The current DSIGET storage utilization in KB. The range is 0–999999.
	53	Blank
<i>maxget</i>	54-59	The maximum storage limit for this task in KB. The range is 0–999999.
	60	Blank
<i>limget</i>	61-66	The applicable maximum storage limit for this task in KB. The range is 0–999999.
	67	Blank
<i>slowget</i>	68-73	The applicable storage slowdown limit for this task in KB. The range is 0–999999.
	74	Blank
<i>curmqi</i>	65-83	The current rate of message traffic from other tasks to this task in KB per minute. The range is 0–999999999.
	84	Blank
<i>sessmqi</i>	85-93	The overall session rate of message traffic from other tasks to this task in KB per minute. The range is 0–999999999.
	94	Blank
<i>maxmqi</i>	95-103	The maximum measured limit for the rate of message traffic from other tasks to this task in KB per minute. The range is 0–999999999.
	104	Blank
<i>limmqi</i>	105-113	The applicable set limit for the rate of message traffic from other tasks to this task in KB per minute. The range is 0–999999999.
	114	Blank
<i>curmqo</i>	115-123	The current rate of message traffic from this task to other tasks in KB per minute. The range is 0–999999999.
	124	Blank

Table 5. Output of a DSITSTAT Command (continued)

Token	Position	Description
<i>sessmqo</i>	125-133	The overall session rate of message traffic from this task to other tasks in KB per minute. The range is 0-999999999.
	134	Blank
<i>maxmqo</i>	135-143	The maximum measured limit for the rate of message traffic from this task to other tasks in KB per minute. The range is 0-999999999.
	144	Blank
<i>limmqo</i>	145-153	The applicable set limit for the rate of message traffic from this task to other tasks in KB per minute. The range is 0-999999999.
	154	Blank
<i>curi/o</i>	155-163	The current number of I/Os per minute for this task. The range is 0-999999999.
	164	Blank
<i>sessi/o</i>	165-173	The number of I/Os per minute recorded for this task. The range is 0-999999999.
	174	Blank
<i>maxi/o</i>	175-183	The maximum number of I/Os per minute recorded for this task. The range is 0-999999999.
	184	Blank
<i>limi/o</i>	185-193	The applicable limit for the number of I/Os per minute for this task. The range is 0-999999999.
	194	Blank
<i>curpen</i>	195-198	The current penalty time ratio in 0.01%. The range is 0-9999.
	199	Blank
<i>sesspen</i>	200-203	The session penalty time ration in 0.01%. The range is 0-9999.
	204	Blank
<i>pendpen</i>	205-213	The current task delay time in 0.01 seconds. The range is 0-999999999.
	214	Blank
<i>totpen</i>	215-223	The total time spent in penalty wait in 0.01 seconds. The range is 0-999999999.
	224	Blank
<i>getkbn</i>	225-233	The current rate of DSIGET activity in KB per minute. The range is 0-999999999.
	234	Blank
<i>getsess</i>	235-243	The session rate of DSIGET activity in KB per minute. The range is 0-999999999.
	244	Blank
<i>g24kbn</i>	245-253	The current rate of DSIGET activity in KB per minute for only the storage in the 24-bit address area. The range is 0-999999999.

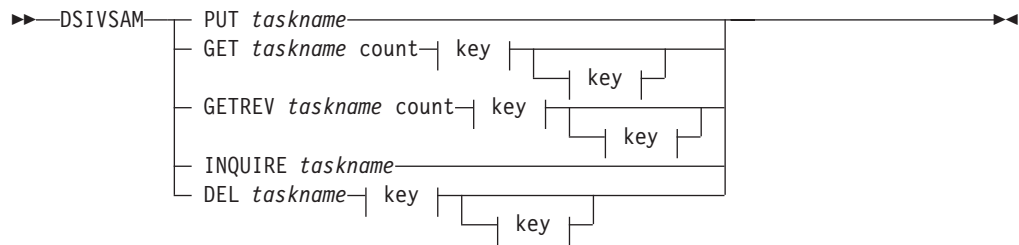
Table 5. Output of a DSITSTAT Command (continued)

Token	Position	Description
	254	Blank
<i>g24sess</i>	255-263	The session rate of DSIGET activity in KB per minute for only the storage in the 24-bit address area. The range is 0-999999999.
	264	Blank
<i>frekbm</i>	265-273	The current rate of DSIFRE activity in KB per minute. The range is 0-999999999.
	274	Blank
<i>fresess</i>	275-283	The session rate of DSIFRE activity in KB per minute. The range is 0-999999999.
	284	Blank
<i>f24kbm</i>	285-293	The current rate of DSIFRE activity in KB per minute for only the storage in the 24-bit address area. The range is 0-999999999.
	294	Blank
<i>f24sess</i>	295-303	The session rate of DSIFRE activity in KB per minute for only the storage in the 24-bit address area. The range is 0-999999999.
	304	Blank
<i>pentime</i>	305-313	The number of penalty seconds that this task has caused other tasks due to its MAXMQIN limit. The range is 0-999999999.
	314	Blank
<i>pentask</i>	315-323	The first 8 bytes contain the name of the task that caused this task to slow down due to a MAXMQIN limit. The rightmost byte contains an indicator as follows: <ul style="list-style-type: none"> + Indicates that the task is currently causing a penalty. - Indicates that the task is not currently causing a penalty, but was the last task to do so.

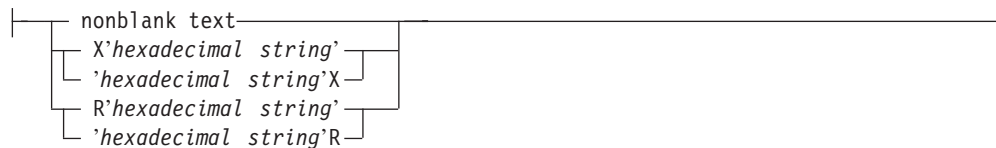
Note: If *taskname* is not valid or inactive, no messages are produced and a return code of 8 is returned.

DSIVSAM (PIPE)

Syntax



key:



Purpose of Command

The DSIVSAM command processor can access keyed VSAM files that are defined by NetView data services tasks, such as DSILog. This allows for implementation of all kinds of VSAM applications, including end-use application development in REXX (in conjunction with the pipeline facility) and intensive VSAM diagnostics.

The DSIVSAM command provides REXX access to any keyed VSAM file on any data services task.

The DSIVSAM command has the following functions:

- A variety of VSAM functions that are intended to be used with NetView pipelines.
- Ability to write multiple records or one at a time.
- Ability to read a single record or a range of records.
- Ability to read using generic (short) keys.
- Ability to delete a single record or a range of records.
- Ability to display all VSAM characteristics of the active file (similar to the LISTCAT command).
- Ability to accept any length key in any position in the record. You must format the records properly when writing them to VSAM and put the key in the correct location within the data record.
- Ability to adapt to any logical record length file.

Samples CNMS8016 and CNMS8021 illustrate the use of the DSIVSAM command.

Usage Notes

Consider the following when using the DSIVSAM command:

- In general, all operands are delimited by at least one blank, and commas, periods, and equal signs are delimiters. Using blanks improves readability.

- You can control who is allowed to initiate actions on VSAM datasets with the command security provided for the DSIVSAM command. You can protect datasets for NetView Data Services Tasks (DST) by using the DATASET class of the SAF product.

For more information, refer to the *Tivoli NetView for z/OS Administration Reference*.

- DSIVSAM is an asynchronous command processor, operating on both the task where the PIPE runs, and on the DST, which you specify as a command parameter. Use the CORRWAIT stage command to make the PIPE wait for the completion of the DST requests. Consider the following:
 - Code a high value for the CORRWAIT timeout value when using DSIVSAM. Regardless of the timeout value specified, the PIPE will finish as soon as the DST requests are done.
 - DSIVSAM requests from the same task are run sequentially at the DST, regardless of how many DSRBs are specified there.
 - DSIVSAM PUT or DEL requests specifying a large amount of data will not receive a completion message until the last I/O is completed. This is one reason the high time value on CORRWAIT is recommended.
 - DSIVSAM GET requests return data one record at a time to help maximize the parallel processing of the PIPE and DST. A high value for the CORRWAIT time might be necessary if a DSIVSAM GET request is issued right after a DSIVSAM PUT or DSIVSAM DEL. The DSIVSAM GET will not read the first buffer until all the previous I/O has completed.
 - The PIPE will be notified if the DST is inactive or terminates while a request is being processed. The CORRWAIT will end when the task ends, and you might not receive as many buffers as requested.
 - A low CORRWAIT time value can result in erratic results.
 - A high CORRWAIT time value will time out if a DST is unresponsive. Otherwise, the CORRWAIT ends immediately if the task ends and the request is completed successfully.

Restrictions

The NetView log uses a single DSRB for all I/O operations to the NetView log. The following should be considered before using DSIVSAM to access the NetView Log:

- Use the DSIWLS macro to write to the NetView log, not DSIVSAM. The DSILOG task requires that no other means of writing to the log be used.
- Using DSIVSAM to read records from the active NetView log will delay the logging of new data. If necessary, read only a small number of records and determine whether reading the records interferes with the DSILOG throughput or causes storage queuing backlogs.
- DSIVSAM provides easy access to VSAM files currently managed by Data Services Tasks. Do not use it to access IBM data sets for which the format of the data is not published. IBM reserves the right to change the format of VSAM files on tasks, for example BNJDSESV (NPDA) and AAUTSKLP (NLDM). In no case should software, except as shipped with NetView, be used to reference this data.

Examples

Example: Defining a Data Services Task and Its VSAM Files

For an example of the use of DSIVSAM, refer to sample CNMS8019. This sample is a CLIST that:

1. Creates a VSAM file by running the DEFVSAMS REXX procedure (sample CNMS8020).

The DEFVSAMS REXX procedure is an example of how to use the DSIVSMX IDCAMS command to create VSAM files without restarting NetView. Change the VOL(CPDLB2) to the volume name of your VSAM files. The other data set parameters are set to work with these examples; however, when you write applications, you can choose the parameters appropriate to your installation.

2. Allocates the files to DDs using the ALLOC command.
3. Starts Data Services Tasks that manage the VSAM file.

The MEM=*member_name* parameters contain Data Services Task definitions that refer to the DD names allocated in Step 2. When the task starts, the VSAM files will be available to the DSIVSAM command.

DSIVSAM DEL (PIPE)

Syntax

►► DSIVSAM DEL *ddname* | key | key |

key:

| nonblank text |
| X'hexadecimal string' |
| 'hexadecimal string'X |
| R'hexadecimal string' |
| 'hexadecimal string'R |

Purpose of Command

The DSIVSAM DEL command deletes the record with the matching key or all records within the boundaries of the stated key range. Message DSI633I is issued when the request has completed successfully.

This command is similar to the DSIVSMX DEL command, which is illustrated in sample CNMS8017.

Operand Descriptions

ddname The DDNAME of a VSAM file. If the file is not open, an attempt is made to open the file before the deletion is complete.

key The key range to delete, specified as one or two keys that determine the range of records to be deleted.

- For the first key specified, the options and padding rules are:
 - A single character string with no embedded blanks. The key is padded with hexadecimal nulls to the full key length and the characters are displayed left-justified.
 - A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed left-justified.
 - A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.

When a single key is used, only the record exactly matching the resulting full key value will be deleted.

- For the second key specified, the options and padding rules are:
 - A single character string with no embedded blanks. The key is padded with hexadecimal ones to the full key length and the characters are displayed left-justified.

- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal ones to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.
- When two keys are used, all records in the resulting key range are deleted.

DSIVSAM GET (PIPE)

Syntax

►►—DSIVSAM GET *taskname count* | key | key |

key:

| nonblank text |
| X'hexadecimal string' |
| 'hexadecimal string'X |
| R'hexadecimal string' |
| 'hexadecimal string'R |

Purpose of Command

The DSIVSAM GET command reads a single VSAM record or a range of records within the bounds of the specified key range. The records are returned as messages; data is returned as characters. The count value *n* limits the read to the first *n* matching records.

The first key specifies the starting key for the request. The value can be shorter than the full key length. The value will be padded on the right to the full key length with binary zeros. The first record read that has a key greater than, or equal to, the starting key will be returned with the key embedded in its physical location in the record text.

The second key is the optional ending key for the read. If the second key is omitted, the value defaults to the first key value, but is padded on the right with binary ones instead of zeros. If the value of the second key is specified, it is also padded with binary ones to the length of the VSAM key for the data set. For example, if a key of A is specified, the GET request will find the first record with an A in the first position of the key.

Note: The second key must resolve to an unsigned binary value that is equal to, or greater than, the first key. Otherwise, no data will be read.

This command is similar to the DSIVSMX GET command, which is illustrated in samples CNMS8015 and CNMS8017.

Operand Descriptions

<i>taskname</i>	The Data Services Task to be accessed.
<i>count</i>	The maximum number of records to be read.
<i>key</i>	The key range to be read, which is specified as one or two keys that determine the range of records to be read.

For the first key specified, the options and padding rules are:

- A single character string with no embedded blanks. The key is padded with hexadecimal nulls to the full key length and the characters are displayed left-justified.

- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.

For the second key specified, the options and padding rules are:

- A single character string with no embedded blanks. The key is padded with hexadecimal ones to the full key length and the characters are displayed left-justified.
- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal ones to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.
- If the second key is omitted, the value for the second key is the original value of the first key, using the padding rules for the second key value.

DSIVSAM GETREV (PIPE)

Syntax

►► DSIVSAM GETREV *taskname count* | key | key |

key:

| nonblank text |
| X'hexadecimal string' |
| 'hexadecimal string'X |
| R'hexadecimal string' |
| 'hexadecimal string'R |

Purpose of Command

The DSIVSAM GETREV command is similar to DSIVSAM GET, except that VSAM records are read in reverse sequence. As with the GET command, the records are returned as a message. The first key is expected to resolve to an unsigned value that is greater than, or equal to, the second key. The first key is padded with binary ones and the second key is padded with binary zeros, if necessary. The second key defaults to a zero-padded equivalent of the first key.

Each byte of data in the record results to one character in the output.

Note: The first key must resolve to an unsigned binary value that is equal to, or greater than, the second key. Otherwise, no data will be read.

This command is also similar to the DSIVSMX GET command, which is illustrated in samples CNMS8015 and CNMS8017.

DSIVSAM INQUIRE (PIPE)

Syntax

▶—DSIVSAM INQUIRE *taskname*—▶

Purpose of Command

The DSIVSAM INQUIRE command displays the data set characteristics for the active VSAM file on the task *taskname*.

This command is similar to the DSIVSMX INQUIRE command, which is illustrated in sample CNMS8016.

Operand Descriptions

TASKID	The operator ID or task ID where the INQUIRE command ran.
DDNAME	The name of the DD statement that identifies the data set.
DSN	The name of the VSAM file allocated to the DDNAME.
VOLUME(S)	The list of the volume names where the VSAM data set resides.
TYPE	The type of VSAM object.
BASE	The base cluster of a VSAM file.
PATH	The path used to access the VSAM file using an alternate key.
AIX	The alternate index accessed directly (not usually done).
AUTOTOKE	A 16-character time value when the data set was opened. The value is the CPU store clock value in hexadecimal.
PRISEC	Indicates whether the Data Services Task is to use the primary or secondary VSAM file.
NSR	Indicates whether the Data Services Task is to use NSR. The value is either Y or N.
LSR	Indicates whether the Data Services Task is to use LSR. The value is either Y or N.
DFR	Indicates whether the Data Services Task is to use DFR. The value is either Y or N.
ADR	Specify records by their address. The value is always N.
KEY	Specifies keyed access. The value is either Y or N.
SEQ	Specifies sequential access. The value is either Y or N.
DIR	Specifies direct access. The value is either Y or N.
IN	Specifies read-only data sets. The value is Y.
OUT	Specifies read-write data sets. The value is Y.
KSDS	Specifies keyed-sequential data sets. The value is Y.
OFLAG	Specifies if the data set is open. The value is either Y or N.

SHOWR15	If zero is returned, the operation was successful. Any other number that is returned is a SHOWCB error code from register 15.
SHOWR00	If zero is returned, the operation was successful. Any other number that is returned is a SHOWCB error code from register zero.
KEYLEN	The length of the key field for data records in the data component.
RKP	The displacement of the key field from the beginning of a data record.
STRNO	The number of requests for which VSAM is to remember its position in the data set.
BUFSP	The amount of space specified in the ACB or GENCB for I/O buffers.
DLRECL	The length of data records in the data component (maximum length for variable-length data records).
DCINV	The control interval size for the data component.
DBUFND	The number of I/O buffers to be used for data, as specified in the ACB or GENCB.
DBUFNO	The number of data component I/O buffers in use.
DNEXT	The number of extents now allocated to the data component. The maximum that can be allocated is 123.
DFS	The number of free control intervals per control area in the data component.
DNCIS	The number of control intervals that have been split in the data component.
DNSSS	The number of control areas that have been split in the data component.
DNEXCP	The number of EXCP macros that VSAM has issued for access to the data component.
DNLOGR	The number of records in the data component.
DNRETR	The number of records that have been retrieved from the data component.
DNINSR	The number of records that have been inserted into, or added to, the data component.
DNUPDR	The number of records in the data component that have been updated.
DNDELR	The number of records that have been deleted from the data component.
DAVSPAC	The amount of available space in the data component, in bytes.
STRMAX	The maximum number of strings concurrently active.
BSTRNO	The number of strings initially allocated for access to the base cluster by a path.
DENDRBA	The ending relative byte address (RBA) of the space used by the data component of the last used byte in the data set (high-used RBA).
DHALCRBA	The RBA of the end of the data component (high-allocated RBA).

ILRECL	The length of index records in the index component (control interval length minus 7).
ICINV	The control interval size for the index component.
IBUFNI	The number of I/O buffers to be used for index entries, as specified in the ACB or GENCB.
IBUFND	The number of I/O buffers in use for the index component.
INEXT	The number of extents now allocated to the index component. The maximum that can be allocated is 123.
INIXL	The number of levels in the index component.
INEXCP	The number of EXCP macros that VSAM has issued to access the index component.
INLOGR	The number of records in the index component.
IAVSPAC	The amount of available space in the index component, in bytes.
IENDRBA	The ending relative byte address (RBA) of the space used by the index component of the last used byte in the data set (high-used RBA).
IHALCRBA	The RBA of the end of the index component (high-allocated RBA).

DSIVSAM PUT (PIPE)

Syntax

▶▶—DSIVSAM PUT *taskname*—▶▶

Purpose of Command

The DSIVSAM PUT command creates or replaces the record with the specified exact key with the specified data. This command produces variable length VSAM records and sends message DSI633I to indicate the request has completed successfully.

Samples CNMS8016 and CNMS8021 illustrates the use of the DSIVSAM PUT command.

Operand Descriptions

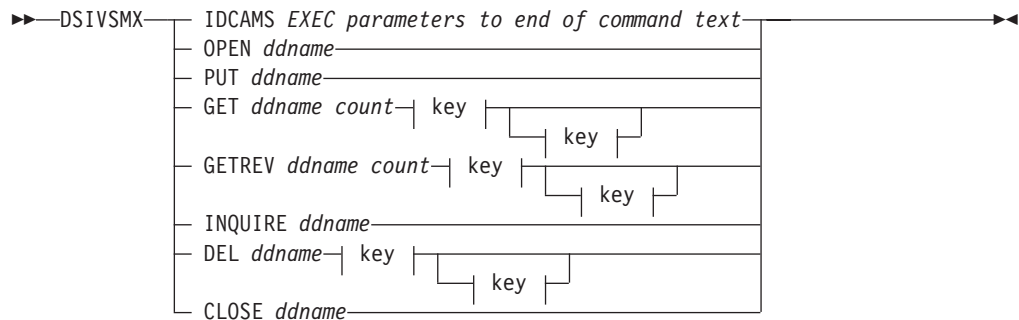
taskname

The Data Services Task to be accessed.

Separate the command parameters by one or more blanks. Other standard delimiters are treated as blanks by DSIVSAM.

DSIVSMX (PIPE)

Syntax



key:



Purpose of Command

The DSIVSMX command processor can define, read, and write keyed VSAM files directly from REXX without using Data Services Tasks. This enables the implementation of VSAM applications, including end-use application development in REXX (in conjunction with the pipeline facility), and intensive VSAM diagnostics.

The DSIVSMX command provides REXX access to keyed VSAM files and to IDCAMS, the VSAM Access Method Services utility.

The DSIVSMX command has the following functions:

- A variety of VSAM functions that are intended to be used with NetView pipelines
- IDCAMS access for defining, deleting, and maintaining VSAM files
- Open and close any number of VSAM files
- Write one record or multiple records with one request
- Read a single or range of records
- Read records in forward or reverse
- Read using generic (short) keys
- Delete a single record or range of records
- Display VSAM characteristics of the active file
- Adapt to any length key in any position in the record
- Adapt to any logical record length file
- Access alternate indexes

Operand Descriptions

- ddname* Specifies a 1- to 8-character DDNAME. This label is on a DD statement in the NetView JCL or is a DDNAME defined by a NetView ALLOC command.
- count* Specifies a decimal number. The maximum number of records to be read.
- key* Specifies the keys of the VSAM records for the request.
- If one key is specified, it is used for both a low range value and a high range value. If the length of the key used is smaller than the length of the key of the VSAM file, the key is padded (extended). Padding rules differ for the low and high values, based on which function is being performed.
- If two keys are specified, the values are treated as follows:
- For GET and DEL requests, the first key is the starting (low) value and the second key, if any, is the ending (high) value.
 - For GETREV requests, the first key is the starting (high) value and the second key, if any, is the ending (low) value.
- In general, low key values are padded with binary zeros (X'00') and high key values are padded with binary ones (X'FF').
- Keys are padded (extended) if the number of characters in the key you used is less than the physical key size. See the individual functions for specific rules, which differ by functions.
- Note:** For PUT requests, include the key in the data records as they are written. Ensure your record is long enough to include the entire key. Generic keys are not valid for PUT.

EXEC parameters to end of command text

Data is passed to the IDCAMS VSAM utility as if it were coded on the EXEC card in MVS JCL. Refer to the MVS/DFP™ library or the DFSMS/MVS® library for the description of IDCAMS.

Usage Notes

Consider the following when using the DSIVSMX command:

- In general, all operands are delimited by at least one blank. Commas, periods, and equal signs are also delimiters. Commands are easier to read when you use blanks.
- You can use DSIVSMX with alternate index VSAM files. Refer to *Tivoli NetView for z/OS Customization: Using Pipes* for more information.
- You can control who accesses data sets by the following means:
 - The command security provided for the DSIVSMX command
 - The READSEC and WRITESEC commands
 - The DATASET class of the SAF product.

For more information, refer to the *Tivoli NetView for z/OS Security Reference*.

DSIVSMX CLOSE (PIPE)

Syntax

▶▶—DSIVSMX CLOSE *ddname*—▶▶

Purpose of Command

The DSIVSMX CLOSE command closes the file associated with *ddname*, which ensures all buffered records are written to the disk. This command is issued when the operator logs off.

DSIVSMX CLOSE sends message DSI633I to indicate the request has completed successfully.

Samples CNMS8014, CNMS8015, and CNMS8020 illustrate the use of the DSIVSMX CLOSE command.

DSIVSMX DEL (PIPE)

Syntax

►► DSIVSMX DEL *ddname* | key | key |

key:

| nonblank text |
| X'hexadecimal string' |
| 'hexadecimal string'X |
| R'hexadecimal string' |
| 'hexadecimal string'R |

Purpose of Command

The DSIVSMX DEL command deletes the record with the exact matching key or all records within the boundaries of the stated key range. Message DSI633I is sent to indicate the request has completed successfully.

Sample CNMS8017 illustrates the use of the DSIVSMX DEL command.

Operand Descriptions

ddname Specifies the DDNAME of a VSAM file. If the file to be deleted is not open, an attempt is made to open the file.

key Specifies the key range to be deleted, which is specified as one or two keys.

1. For the first key specified, the options and padding rules are:
 - A single character string with no embedded blanks. The key is padded with hexadecimal nulls to the full key length and the characters are displayed left-justified.
 - A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed left-justified.
 - A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.

When a single key is used, only the record exactly matching the resulting full key value will be deleted.

2. For the second key specified, the options and padding rules are:
 - A single character string with no embedded blanks. The key is padded with hexadecimal ones to the full key length and the characters are displayed left-justified.

- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal ones to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.
- When two keys are used, all records in the resulting key range are deleted.

DSIVSMX GET (PIPE)

Syntax

►►—DSIVSMX GET *ddname count* | key | key |

key:

| nonblank text |
| X'hexadecimal string' |
| 'hexadecimal string'X |
| R'hexadecimal string' |
| 'hexadecimal string'R |

Purpose of Command

The DSIVSMX GET command reads a record or sequence of records from a VSAM file. The output is returned in a piped output stream. If the file is not open, an attempt is made to open the file before the GET is started.

The first key specifies the starting key for the request. The value can be shorter than the full key length. The value will be padded on the right to the full key length with binary zeros. The first record read that has a key greater than, or equal to, the starting key will be returned with the key embedded in its physical location in the record text.

The second key is the optional ending key for the read. If the second key is omitted, the value defaults to the first key value, but is padded on the right with binary ones instead of zeros. If the value of the second key is specified, it is also padded with binary ones to the length of the VSAM key for the data set. For example, if a key of A is specified, the GET request will find the first record with an A in the first position of the key.

The DSIVSMX GET will then read a maximum of *count* records in sequence.

Note: The second key must resolve to an unsigned binary value that is equal to, or greater than, the first key. Otherwise, no data will be read.

Samples CNMS8015 and CNMS8017 illustrate the use of the DSIVSMX GET command.

Operand Descriptions

ddname Specifies the DDNAME of a VSAM file. If the file is not open, the system attempts to open the file before GET is issued.

count Specifies the maximum number of records you want to retrieve.

keys Specifies the key range to read, specified as one or two keys that determine the range of records to read.

For the first key specified, the options and padding rules are:

- A single character string with no embedded blanks. The key is padded with hexadecimal nulls to the full key length and the characters are displayed left-justified.
- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.

For the second key specified, the options and padding rules are:

- A single character string with no embedded blanks. The key is padded with hexadecimal ones to the full key length and the characters are displayed left-justified.
- A hexadecimal character string in the form X'hexdigits' or 'hexdigits'X. The key is padded with hexadecimal ones to the full key length and the 'hexdigits' are displayed left-justified.
- A hexadecimal character string in the form R'hexdigits' or 'hexdigits'R. The key is padded with hexadecimal nulls to the full key length and the 'hexdigits' are displayed right-justified. This notation is useful if the VSAM file is intended to have numeric (binary) keys.
- If the second key is omitted, the value for the second key is the original value of the first key, using the padding rules for the second key value.

Usage Notes

Hexadecimal notation must be used with DSIVSMX when using special characters, including comma(,), space(), period(.), and equal sign(=). For example, the following will not work:

```
'PIPE NETV MOE DSIVSMX GET taskid 1 19991207-13:23:59.012345 | stem X.'
```

To resolve the problem caused by using special characters, issue GET as follows, paying special attention to single quotes and double quotes:

```
"PIPE NETV MOE DSIVSMX GET taskid 1 X'"C2X(19991207-13:23:59.012345)'" | STEM x."
```

DSIVSMX IDCAMS (PIPE)

Syntax

▶▶—DSIVSMX IDCAMS EXEC parameters to end of command text—▶▶

Purpose of Command

The DSIVSMX IDCAMS command calls the IDCAMS utility to perform data set maintenance. The IDCAMS control statements are read from the message stream that invoked the DSIVSMX command. Typically, this is used within a PIPE with a STEM variable as the input stream.

Samples CNMS8013, CNMS8014, CNMS8015, and CNMS8020 illustrate the use of the DSIVSMX IDCAMS command.

DSIVSMX INQUIRE (PIPE)

Syntax

▶—DSIVSMX INQUIRE *ddname*—▶

Purpose of Command

The INQUIRE command retrieves VSAM data set characteristics for the file named by the DDNAME *ddname*.

Following are valid output values:

TASKID	The operator ID or task ID where the INQUIRE command ran.
DDNAME	The name of the DD statement that identifies the data set.
DSN	The name of the VSAM file allocated to the DDNAME.
VOLUME(S)	The list of the volume names where the VSAM data set resides.
TYPE	The type of VSAM object.
BASE	The base cluster of a VSAM file.
PATH	The path used to access the VSAM file using an alternate key.
AIX	The alternate index accessed directly. BASE and PATH are used most often. You should not see AIX unless you are performing a task such as low-level diagnostics.
AUTOTOKE	A 16-character time value when the data set was opened. The value is the CPU store clock value in hexadecimal.
PRISEC	This value is always blank. DSIVSMX does not have a primary or secondary concept. DSIVSMX can service many DDNAME parameters.
NSR	This value is always Y. DSIVSMX does not support LSR and DFR specification.
LSR	This value is N. DSIVSMX does not support LSR specification.
DFR	This value is N. DSIVSMX does not support DFR specification.
ADR	Specify records by their address. The value is always N.
KEY	Specifies keyed access. The value is either Y or N.
SEQ	Specifies sequential access. The value is either Y or N.
DIR	Specifies direct access. The value is either Y or N.
IN	Specifies read-only data sets. The value is Y.
OUT	Specifies read-write data sets. The value is Y.
KSDS	Specifies keyed-sequential data sets. The value is Y.
OFLAG	Specifies if the data set is open. The value is either Y or N.

SHOWR15	If zero is returned, the operation was successful. Any other number that is returned is a SHOWCB error code from register 15.
SHOWR00	If zero is returned, the operation was successful. Any other number that is returned is a SHOWCB error code from register zero.
KEYLEN	The length of the key field for data records in the data component.
RKP	The displacement of the key field from the beginning of a data record.
STRNO	The number of requests for which VSAM is to remember its position in the data set.
BUFSP	The amount of space specified in the ACB or GENCB for I/O buffers.
DLRECL	The length of data records in the data component (maximum length for variable-length data records).
DCINV	The control interval size for the data component.
DBUFND	The number of I/O buffers to be used for data, as specified in the ACB or GENCB.
DBUFNO	The number of data component I/O buffers in use.
DNEXT	The number of extents now allocated to the data component. The maximum that can be allocated is 123.
DFS	The number of free control intervals per control area in the data component.
DNCIS	The number of control intervals that have been split in the data component.
DNSSS	The number of control areas that have been split in the data component.
DNEXCP	The number of EXCP macros that VSAM has issued for access to the data component.
DNLOGR	The number of records in the data component.
DNRETR	The number of records that have been retrieved from the data component.
DNINSR	The number of records that have been inserted into, or added to, the data component.
DNUPDR	The number of records in the data component that have been updated.
DNDELR	The number of records that have been deleted from the data component.
DAVSPAC	The amount of available space in the data component, in bytes.
STRMAX	The maximum number of strings concurrently active.
BSTRNO	The number of strings initially allocated for access to the base cluster by a path.
DENDRBA	The ending relative byte address (RBA) of the space used by the data component of the last used byte in the data set (high-used RBA).
DHALCRBA	The RBA of the end of the data component (high-allocated RBA).

ILRECL	The length of index records in the index component (control interval length minus 7).
ICINV	The control interval size for the index component.
IBUFNI	The number of I/O buffers to be used for index entries, as specified in the ACB or GENCB.
IBUFND	The number of I/O buffers in use for the index component.
INEXT	The number of extents now allocated to the index component. The maximum that can be allocated is 123.
INIXL	The number of levels in the index component.
INEXCP	The number of EXCP macros that VSAM has issued to access the index component.
INLOGR	The number of records in the index component.
IAVSPAC	The amount of available space in the index component, in bytes.
IENDRBA	The ending relative byte address (RBA) of the space used by the index component of the last used byte in the data set (high-used RBA).
IHALCRBA	The RBA of the end of the index component (high-allocated RBA).

Sample CNMS8016 illustrates the use of the DSIVSMX INQUIRE command.

DSIVSMX OPEN (PIPE)

Syntax

▶▶—DSIVSMX OPEN *ddname*—▶▶

Purpose of Command

The DSIVSMX OPEN command opens a VSAM file locally to the task that issues the command. Access remains open until a DSIVSMX CLOSE command is issued or the task is logged off.

DSIVSMX OPEN sends message DSI633I to indicate the request has completed successfully.

DSIVSMX automatically opens the VSAM file for any request if a DSIVSMX OPEN command is not issued first.

DSIVSMX OPEN opens the VSAM file in read mode. If the data set is empty, the open request will fail. In this case, use the DSIVSMX PUT command to put an initial record in the file.

Avoid using DSIVSMX on VSAM files of Data Services Tasks (DST), especially if they have the REUSE attribute. If you OPEN one of these files and the DST attempts a RESET (such as DSILOG switching between primary and secondary logs), the SWITCH will fail. To access VSAM files of DST tasks, use the DSIVSAM command processor.

Samples CNMS8014 and CNMS8015 illustrate the use of the DSIVSMX OPEN command.

Operand Descriptions

ddname

Specifies the DDNAME of a VSAM file.

DSIVSMX PUT (PIPE)

Syntax

▶▶—DSIVSMX PUT *ddname*—◀◀

Purpose of Command

The DSIVSMX PUT command expects a piped input stream of record images. The image format is identical to the output format of DSIVSMX GET. The key is specified within the text of the data records. This is useful in preloading a data set from data built into stem variables. Each byte of input data represents one byte of output data in the identical buffer position in the VSAM record.

DSIVSMX PUT produces variable length VSAM records.

DSIVSMX PUT sends message DSI633I to indicate the request has completed successfully.

Samples CNMS8014, CNMS8015, CNMS8017, CNMS8018, and CNMS8020 illustrate the use of the DSIVSMX PUT command.

Operand Descriptions

ddname

Specifies the DDNAME of a VSAM file. If the file is not open, the system attempts to open the file before PUT is issued.

Restrictions

The DUIFECMV command processor must run under the autotask DUIFEAUT.

Examples

Example: DUIFECMV Automation Table Entry

The following is an example of an automation table entry:

```
IF (MSUSEG(0000) ^= '' | MSUSEG(0002) ^= '') & HIER ^= ''
  THEN EXEC(CMD('DUIFECMV
                DOMAIN=SPNAME
                GMFHSDOM=CNM01
                CLASS=1.3.18.0.0.3315.0.3.1
                OBJNAME=2.9.3.2.7.4=SPNAME,1.3.18.0.0.3519=APPLNAME
                INDICAT=2
                STATUS=131')
            ROUTE(ONE DUIFEAUT)) CONTINUE(Y) ;
```

FLUSHQ (REXX)

Syntax

FLUSHQ

►► 'FLUSHQ' MESSAGES ' ◄◄

Purpose of Command

The FLUSHQ instruction is used to remove all trapped messages from the message queue that have not been removed using MSGREAD.

Return Codes

FLUSHQ sets the value of the return code RC to indicate the results of processing as follows:

Return Code	Meaning
-1	Syntax error
0	Successful completion

Usage Notes

Consider the following when using the FLUSHQ instruction:

- Because the TRAP instruction does not clear the queue of messages trapped by a previous TRAP, you should issue a FLUSHQ instruction between multiple TRAP instructions coded in the same command list.
- You can code the FLUSHQ instruction in both a nested REXX command list and the initial REXX command list. If you use the REXX CALL instruction to invoke a nested command list, all trapped messages are removed from the initial REXX command list trap message queue. If you invoke the nested list without the CALL instruction, only the trapped messages for the nested REXX command list are removed.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

GETM Commands

Introduction

Some commands return a reply that appears to be a sequence of separate messages, when in fact the reply is a single multiline write-to-operator (MLWTO) message. NetView treats an MLWTO message as a single message. Only the first nonblank message identifier that appears as part of an MLWTO message is made available to satisfy a REXX TRAP instruction, or a NetView command list language &WAIT command. The first nonblank line of a multiline message is also the only line used for comparisons in NetView automation tables.

When the NetView automation table invokes a command list as a result of automating an MSU, the MSU data and hardware monitor resource hierarchy data are available to the command list. The MSU data is contained in one buffer, and the hardware monitor resource hierarchy data is contained in a second buffer.

Figure 9 shows an example MLWTO message. The message is in response to a LIST KEY=PF1 command. The MLWTO message appears to be a sequence of several separate messages, but the single quote that appears as the message type identifier identifies the message as a single MLWTO message from NetView. A double quote identifies a multiline message from an IBM or Tivoli product other than NetView. An equal sign identifies a user-written multiline message.

```
NCCF                                TIVOLI NETVIEW  NCF01 OPER1   10/13/98 11:44:23
* NTVAA   LIST KEY=PF1
' NTVAA
DSI606I DISPLAY OF PF/PA KEY SETTINGS FOR NCCF
DSI607I KEY  ----TYPE----  -----COMMAND-----  SET-APPL
DSI608I PF1  IMMED,APPEND  HELP                                NETVIEW
```

Figure 9. Example Multiline Message

TRAP, &WAIT, and NetView automation table processing use only the first nonblank line of a multiline message.

NetView provides commands that enable you to work with multiline messages and management services unit (MSU) buffers in a command list. These commands enable you to work with information in each individual line of a multiline message or data buffer of an MSU. The commands are as follows:

GETMLINE

Assigns the text of a specific line of a multiline message or a specific MSU data buffer to a specified variable.

GETMPRES

Retrieves the message presentation attributes of the text buffer for each line of a message.

GETMSIZE

Determines the number of lines of a multiline message. GETMSIZE also determines the number of MSU data buffers.

GETMTFLG

Retrieves the message type flags from the text buffer of any message or MSU.

GETMTYPE

Determines the line type of a specific line in a multiline message, or identifies the type of MSU data.

Note: The GETM commands pre-date NetView pipelines. You can use the PIPE command to solve these types of problems. For more information about the PIPE command and its stages, refer to *Tivoli NetView for z/OS Customization: Using Pipes*.

GETMLINE (REXX)

Syntax

GETMLINE

►►—GETMLINE *variable_name line_number*—◄◄

Purpose of Command

Use the GETMLINE command within a command list to assign the text of an individual line of a multiline message to a specified variable. GETMLINE can also be executed for a buffer containing MSU type data in a NetView command list environment. Use this command in a command list that is driven by NetView automation or that has processed a message using MSGREAD (REXX) or &WAIT (NetView command list language).

Operand Descriptions

variable_name

Identifies a command list variable coded in this command. GETMLINE assigns the value of the line you specify to this variable name. If you write the command list in the NetView command list language, do not use the ampersand (&) in the variable name.

The maximum character length of a variable name depends on the following:

- 11 When invoked from a command list written in the NetView command list language.
- 31 When invoked from a REXX-format command list and is a NetView global variable name.
- 250 When invoked from a REXX-format command list and is a REXX variable name.

line_number

Identifies the number of the line in the multiline message or the specific data buffer in the MSU from which you want to obtain the value. Blank lines in a multiline message are counted as lines.

When you code a number for *line_number* in REXX command lists, put the number within single quotes that enclose GETMLINE. For example:

```
'GETMLINE LINE1 2'
```

When you code a variable for *line_number* in REXX command lists, leave a blank after *variable_name* and close the quotes. Code the name of the variable that contains the value for *line_number* outside the quotes. For example, if the value of *line_number* is contained in a variable named NUM1, code:

```
'GETMLINE LINE1 'NUM1
```

Return Codes

GETMLINE sets the return code (RC or &RETCODE) to indicate the processing results, as follows:

Return Code	Meaning
0	Processing was successful.

8	Storage is not available to continue processing.
100	There are not enough parameters, or they are not issued from a command list.
104	The <i>line_number</i> length is not valid.
108	The <i>line_number</i> value is not valid.
116	The requested line number does not exist.
120	There is no multiline message or MSU.
124	There is no message or MSU to process.
128	The command list dictionary update failed.
132	Returned data has been truncated.
136	The command parameter is too long.

Usage Notes

In a NetView command list environment, if MSU data is greater than 255 characters, GETMLINE completes with a return code of 132 indicating that the 255 character limit for command list variables was exceeded. If this happens, the data is truncated at 255 characters before it is placed in the command list variable.

GETMLINE executes successfully for a buffer containing HIER type data because the data is not greater than 255.

For example, an automation task command list written in the NetView command list language contains the following two lines:

```
&NUM = 2
GETMLINE SECOND &NUM
```

These two lines in a command list place the text of the second line of a multiline message into the command list variable &SECOND.

See “Example: Command Lists Processing MLWTO Messages” on page 277 for example command lists that show how GETMLINE can be used with MLWTO messages.

GETMPRES (REXX)

Syntax

GETMPRES

►—GETMPRES *variable_name line_number*—◄

Purpose of Command

You can use the GETMPRES command in command lists to retrieve the 4-byte message presentation value from the text buffer corresponding to each line of text for a message or MSU. If the message has no presentation attribute characters specified, the value of GETMPRES is null. If the value is *not* null and GETMTFLG bit 16 is set on, GETMPRES values are used for message and MSU presentation. These GETMPRES values are taken from one of the following sources:

- The presentation overrides specified in the message data buffer (MDB)
- The presentation overrides as specified by a previous automation table action

When one or more presentation attributes are set by the automation table (with the COLOR, HIGHINT, or XHILITE actions) all four of the presentation attributes for the message or MSU are copied to the GETMPRES fields and used to display that message or MSU. Attributes that are not set by the automation table are taken from either MDB override fields, the fields in MSGGFGPA, or the values specified with the OVERRIDE or DEFAULTS SCRNFMT commands.

If GETMPRES is null, the presentation attributes of the message or MSU are taken from *one* of the following:

- The fields in MSGGFGPA
- The values specified with the OVERRIDE or DEFAULTS SCRNFMT command
- For MSUs, the hardware monitor defaults

When GETMPRES is null, other presentation attributes will apply to this message when it is displayed. When GETMPRES is null, you can check the fields in MSGGFGPA for presentation attributes.

Before using GETMPRES, you can determine whether the presentation vectors exist and are valid by issuing a GETMTFLG and checking bit 16.

Operand Descriptions

variable_name

Indicates the name of a command list variable, without the ampersand (&), that is used to store the returned data. The maximum character length of a variable name depends on the following:

- | | |
|-----|---|
| 11 | When invoked from a command list written in the NetView command list language. |
| 31 | When invoked from a REXX-format command list and is a NetView global variable name. |
| 250 | When invoked from a REXX-format command list and is a REXX variable name. |

The data represented by the variable name consists of a string of four characters of presentation attribute data. The four characters have the following meanings and possible values:

Byte Description

- 1 Control field
 - 00 MVS alarm-off indicator
 - 80 MVS alarm-on indicator

Note: This is an MVS indicator which NetView does not use. The NetView indicators that actually control this alarm are in IFRAUTA1 bits 13, 14, and 30. The IFRAUTA1 indicators are also set by the automation table.
- 2 Color field
 - 00 For messages, the value is determined from the MSGGFGPA setting or NetView OVERRIDE or DEFAULTS SCRNFMT settings, respectively. For MSUs, the hardware monitor-established defaults are used.
 - F0 Presentation background. Black on display, white on printer.
 - F1 Blue
 - F2 Red
 - F3 Pink (magenta)
 - F4 Green
 - F5 Turquoise (cyan)
 - F6 Yellow
 - F7 Presentation neutral. White on display, black on printer.

The color field is also set by the COLOR action in the automation table.

- 3 Highlighting field
 - 00 No highlighting
 - F1 Blinking
 - F2 Reverse video
 - F4 Underscore

The highlighting field is also set by the XHILITE action in the automation table.

- 4 Intensity field
 - 00 For messages, the value is determined from the MSGGFGPA setting or NetView OVERRIDE or DEFAULTS SCRNFMT settings, respectively. For MSUs, the hardware monitor-established defaults are used.
 - E4 Normal intensity
 - E8 High (bright) intensity

The intensity field is also set by the HIGHINT action in the automation table.

line_number

Indicates the number of the line for which data is requested.

Return Codes

GETMPRES sets the return code to indicate the processing results, as follows:

Return Code	Meaning
-------------	---------

0	Processing was successful.
100	There are not enough parameters, or they were not issued from a command list.
104	The <i>line_number</i> length is not valid.
108	The <i>line_number</i> value is not valid.
116	The requested line number does not exist.
120	There is no multiline message or MSU.
124	There is no message or MSU to process.
128	The command list dictionary update failed.
132	The command parameter is too long.

Examples

Assume the following statement is coded in a REXX or NetView command list language command list, respectively:

```
'GETMPRES ATTRS 1'
&X = 1
GETMPRES ATTR2 &X
```

This coding causes NetView to update the specified variable with the presentation attributes of the first line of the message being processed.

The presentation data is considered null and a zero length is returned under the following conditions:

- The IFRAUTA1 indicator 48 is zero.
- You are using &IFRAUTA1 (NetView command list language) or IFRAUTA1() (REXX).
- The buffer header vectors are incorrect. For example, they have the incorrect length or key fields.

GETMSIZE (REXX)

Syntax

GETMSIZE

▶▶—GETMSIZE *variable_name*—————▶▶

Purpose of Command

You can use the GETMSIZE command in command lists to determine the number of lines in a multiline message. GETMSIZE can also be executed for a buffer string containing management services unit (MSU) data. The size returned is 1 or 2, depending on whether HIER data exists. Use this command in a command list invoked by NetView automation or that processes a message using MSGREAD (REXX) or &WAIT (NetView command list language).

Operand Descriptions

variable_name

Identifies a command list variable coded in this command. If you write the command list in the NetView command list language, do not use the ampersand (&) in the variable name. GETMSIZE sets the value of the variable to the number of lines in the multiline message or the number of associated MSU buffers. If a message is a single-line instead of a multiline message, the variable value is set to 1. This number includes any blank lines in the multiline message.

The maximum character length of a variable name depends on the following:

- 11 When invoked from a command list written in the NetView command list language.
- 31 When invoked from a REXX-format command list and is a NetView global variable name.
- 250 When invoked from a REXX-format command list and is a REXX variable name.

Return Codes

GETMSIZE sets the return code (RC or &RETCODE) to indicate the processing results, as follows:

Return Code	Meaning
0	Processing was successful.
8	Storage is not available to continue processing.
100	There are not enough parameters, or they were not issued from a command list.
104	The internal numeric conversion failed.
108	The command list dictionary update failed.
112	The command parameter is too long.

For example, assume the following statement is coded in an automation command list:

```
GETMSIZE NUMLINES
```

If the command list containing this command is triggered by the message in the following example, the variable &NUMLINES is set to the value of 7:

```
| IEE104I 13.02.15 90.89 ACTIVITY 607 C
| JOBS      M/S      TS USERS      SYSAS      INITS      ACTIVE/MAX VTAM
| 00000     00007     00000     00008     00001     00000/00300
| LLA      LLA      LLA      NSW S    JES2JES2   IEFPROC  NSW S
| SYSLOG   621      IEFPROC  OWT S    BASENET   BASENET  VTAM   NSW S
| TSO      TSO      TCAS     OWT S    NETVREL2  NETVREL2 NETVIEW NSW S
| NETV2    NETV2    NETVIEW  NSW S
```

Figure 10. IEE104I Message to Trigger an Automation Command List

See “Example: Command Lists Processing MLWTO Messages” on page 277 for examples of command lists that show how GETMSIZE is used with MLWTO messages.

GETMTFLG (REXX)

Syntax

GETMTFLG

▶▶—GETMTFLG *variable_name line_number*—————▶▶

Purpose of Command

You can use the GETMTFLG command in command lists to retrieve the presentation override flag for each line of text for a message or MSU. The value returned by this command processor is the 16-bit HDRTLNTY field from the buffer header. Bit 16 is the presentation override flag.

The GETMTFLG command will return a null value if the buffer being evaluated does not have an extended buffer header. Messages received from MVS through extended multiple console support (EMCS) consoles will have extended buffer headers with all of the GETMTFLG information. Other messages might have an extended buffer header, or might not have all of the information. For line-type information, it is recommended that the GETMTYPE command be used.

Operand Descriptions

variable_name

Indicates the name of a command list variable, without the ampersand (&), that is used to store the returned data. The maximum character length of a variable name depends on the following:

- 11 When invoked from a command list written in the NetView command list language.
- 31 When invoked from a REXX-format command list and is a NetView global variable name.
- 250 When invoked from a REXX-format command list and is a REXX variable name.

The data represented by the variable name consists of a string of 16 bits represented as EBCDIC ones and zeros. The bits have the following meaning corresponding to the field HDRTLNTY in NetView macro DSITIB:

- 16 Text object presentation attribute field overrides general object presentation attribute field. This bit indicates that presentation information is available for messages or MSUs.

Notes:

1. Other bits can be tested, but have no recommended use.
2. Bits 1-5 contain line-type information for MVS messages. This line-type information can be of historical or diagnostic use, but is not constant from message to message. Therefore, NetView processes the buffers based on the value returned by GETMTYPE, rather than the value in the GETMTFLG bits 1-5.
3. You can determine whether the extended buffer header exists by checking bit position 48 of IFRAUTA1.

line_number

Indicates the number of the line for which data is requested.

Return Codes

GETMTFLG sets the return code to indicate the processing results, as follows:

Return Code	Meaning
0	Processing was successful.
100	There are not enough parameters, or they were not issued from a command list.
104	The <i>line_number</i> length is not valid.
108	The <i>line_number</i> value is not valid.
116	The requested line number does not exist.
120	There is no multiline message or MSU.
124	There is no message or MSU to process.
128	The command list dictionary update failed.
132	The command parameter is too long.

Examples

Assume the following statement is coded in a NetView command list language command list:

```
&X = 1  
GETMTFLG FLAG2 &X
```

NetView updates the specified variable with the presentation override flag data for the first line of the message being processed.

GETMTYPE (REXX)

Syntax

GETMTYPE

▶▶—GETMTYPE *variable_name line_number*—————▶▶

Purpose of Command

You can use the GETMTYPE command in command lists to determine the line type of an individual line in a multiline message. GETMTYPE can also be executed for a buffer string containing MSU data. Use this command in a command list that is driven by NetView automation or that processes a message using MSGREAD (REXX) or &WAIT (NetView command list language).

Operand Descriptions

variable_name

Identifies a command list variable coded in this command. If you write the command list in the NetView command list language, do not use the ampersand (&) in the variable name.

GETMTYPE sets the value of this variable as one of the following line types:

blank	Single-line message
C	Control line
L	Label line
D	Data line
DE	Combined data and end line
E	End-line without data
H	HIER data
M	Management services unit (MSU) data

The maximum character length of a variable name depends on the following:

- 11** When invoked from a command list written in the NetView command list language.
- 31** When invoked from a REXX-format command list and is a NetView global variable name.
- 250** When invoked from a REXX-format command list and is a REXX variable name.

line_number

Specifies the number of the line in the multiline message for which you want line type information, or the number of the buffer in the MSU that you want. Blank lines in the multiline message are counted as lines. For single-line messages, the value of *line_number* must be 1.

When you code a number for *line_number* in REXX command lists, put the number within single quotes that enclose GETMTYPE. For example:

```
'GETMTYPE TYPE1 3'
```

When you code a variable for *line_number* in REXX command lists, leave a blank after *variable_name* and close the quotes. Code the name of the variable

that contains the value for *line_number* outside the quotes. For example, if the value of *line_number* is contained in a variable named NUM1, code:

```
'GETMTYPE TYPE1 'NUM1
```

Return Codes

GETMTYPE also sets the return code (RC or &RETCODE) to indicate the processing results, as follows:

Return Code	Meaning
0	Processing was successful.
8	Storage is not available to continue processing.
100	There are not enough parameters, or they were not issued from a command list.
104	The <i>line_number</i> length is not valid.
108	The <i>line_number</i> value is not valid.
116	The requested line number does not exist.
120	There is no multiline message or MSU.
124	There is no message or MSU to process.
128	The command list dictionary update failed.
132	The command parameter is too long

Usage Notes

You can use GETMTYPE along with GETMSIZE in a command list to test each line of a multiline message for the type. If you use a variable name for the *line_number* field in the GETMTYPE command, you can test each line. Using GETMSIZE, you can have your command list test the correct number of lines for each message it receives.

If the command list receives the message illustrated in Figure 10 on page 273, the line types would be as follows:

- First line would equal C
- Second and third lines would equal L
- Fourth through the sixth lines would equal D
- Seventh line would equal DE

Notes:

1. MSU data typically consists of one or two buffers. The first buffer is the entire MSU buffer and the second buffer, if present, is the hardware monitor resource hierarchy buffer.
2. The GETMTYPE command pre-dates NetView pipelines. You can use the PIPE command to solve this type of problem. For more information about the PIPE command and its stages, refer to *Tivoli NetView for z/OS Customization: Using Pipes*.

Examples

Example: Command Lists Processing MLWTO Messages

The command lists in Figure 11 on page 278 and Figure 12 on page 279 are examples of how you can code your command list to process based on the information in the individual lines of a multiline message. Figure 11 is written in REXX. Figure 12 is written in the NetView command list language.

```

/*****
/* SESSCNT  COMMAND LIST                               */
/* -----                                           */
/* FUNCTION:  This command list counts the number of OPCTL */
/*            sessions and FLSCN sessions that are active. */
/*            */
/* INPUT PARMS: None                                   */
/*            */
/* OUTPUT:    Two informational messages that display the number of */
/*            OPCTL and FLSCN sessions to the operator.      */
/*****
OPCTLCNT = 0                                           /* init OPCTL counter */
FLSCNCNT = 0                                           /* init FLSCN counter */
'TRAP AND SUPPRESS MESSAGES *'                        /* TRAP the MLWTO msg */
'LISTSESS'                                             /* issue the command */
'WAIT 5 SECONDS FOR MESSAGES'                         /* WAIT for the msg */
SELECT                                                /* SELECT an EVENT */
  WHEN EVENT() = 'M' THEN                             /* message received */
    DO                                                /* process the message */
      'MSGREAD'                                       /* read the message in */
      'GETMSIZE  NUMLINES'                             /* get number of lines */
      DO CNTR = 4 TO NUMLINES                         /* loop thru MLWTO buf */
        'GETMLINE LINE' CNTR                         /* get a line in buf */
                                                /* parse the line out */
        'PARSEL2R LINE APPLNM SRCLUNM SESSNM TYPE RESTLINE'
        IF TYPE = 'OPCTL' THEN                       /* OPCTL session ? */
          OPCLCNT = OPCLCNT + 1                       /* yes,increment count */
        ELSE                                          /* must be FLSCN sess */
          FLSCNCNT = FLSCNCNT + 1                    /* increment count */
        'TRAP NO MESSAGES'                           /* end message trapping*/
        'FLUSHQ MESSAGES'                            /* flush message queue */
      END                                            /* loop thru MLWTO buf */
    END                                            /* process the message */
  OTHERWISE                                         /* event not a message */
    SAY 'ERROR PROCESSING LISTSESS COMMAND'          /* issue an error msg */
END                                                /* SELECT an EVENT */
/* issue messages to operator with count of FLSCN & OPCTL sessions */
SAY 'YOU HAVE ' OPCLCNT ' OPCTL SESSIONS ACTIVE AND'
SAY ' ' FLSCNCNT ' FLSCN SESSIONS ACTIVE'

```

Figure 11. Command List Using Multiline Messages—REXX Example


```

&CONTROL ERR
* * * * *
*
* COMMAND LIST AUTHTOTE
*
* THIS COMMAND LIST DISPLAYS A COUNT OF HOW MANY MESSAGES (OR GROUPS
*
* OF MESSAGES, USING THE * SUFFIX) HAVE BEEN ASSIGNED TO BE
*
* ROUTED TO SPECIFIC AUTHORIZED RECEIVERS.
*
* IT IS CALLED AS FOLLOWS:  AUTHTOTE
*
*                               AUTHTOTE DISPLAY
*
*                               AUTHTOTE SUPPRESS
*
* THE FIRST TWO FORMS WILL DISPLAY THE MESSAGES RECEIVED ("DISPLAY"
* IS THE DEFAULT) AND A COUNT OF THE TOTAL NUMBER OF ASSIGNMENTS
* MADE, AND THE LAST FORM WILL SIMPLY DISPLAY THE TOTAL NUMBER
* OF ASSIGNMENTS.
*
* * * * *
* *
* SAVE THE INPUT PARAMETER
&OPT = &1
* INITIALIZE THE COUNTER FOR THE NUMBER OF MESSAGES ASSIGNED TO
* AUTHORIZED RECEIVERS.
&COUNT = 0
* SET THE COUNTER FOR THE NUMBER OF THE MESSAGE LINE TO BE
* PROCESSED TO 1.  THE WAIT STATEMENT PROCESSES THE FIRST MESSAGE LINE.
&MSZI = 0
* ISSUE THE COMMAND AND ENTER WAIT STATE
&WAIT ENDWAIT &OPT
&WAIT 'LIST MSG=AUTH',DSI636I=-COUNTER,DSI643I=-NOEXIT, +
      *60=-TIMEOUT,*ENDWAIT=-GOIN
-NXTLINE
* ADD 1 TO THE COUNTER FOR NUMBER OF LINES PROCESSED
&MSZI = &MSZI + 1
* IF THE NUMBER OF THE LINE TO BE PROCESSED IS GREATER THAN THE
* NUMBER IF LINES IN THE MESSAGE, THEN DISPLAY THE RESULTS
&IF &MSZI = &NUMLINES &THEN &GOTO -ENDIT
* ASSIGN THE TEXT OF THE NEXT LINE OF THE MESSAGE TO MSGTXT
GETMLINE MSGTXT &MSZI
* PARSE MSGTXT SO THAT THE MESSAGE ID OF THE LINE IS PLACED IN
* THE MSGID VARIABLE.

```

Figure 12. Command List Using Multiline Messages—NetView Command List Language Example (Part 1 of 2)

```

PARSEL2R MSGTXT MSGID MSGSTR
&IF &MSGID = DSI642I &THEN &GOTO -ENDIT
&IF &MSGID = DSI643I &THEN &GOTO -NOEXIT
&IF &MSGID = DSI636I &THEN &GOTO -COUNTER
&GOTO -NXTLINE
-NOEXIT
* THIS ENSURES THAT THE OPERATOR IS INFORMED OF THE RESULTS OF
* THE COMMAND WHEN NO MESSAGES HAVE BEEN ASSIGNED, BUT THE
* MESSAGE INFORMING THE OPERATOR HAS BEEN SUPPRESSED
&IF .&OPT = .SUPPRESS &THEN &GOTO -ENDIT
&EXIT
-COUNTER
* IF THE FIRST MESSAGE LINE IS BEING PROCESSED, DETERMINE THE NUMBER
* OF LINES FOR THE MESSAGE
IF &MSZI = 1 &THEN &GOTO -LINES
-COUNTER1
* TOTAL THE NUMBER OF MESSAGES ASSIGNED TO PRIMARY RECEIVERS
&COUNT = &COUNT + 1
&GOTO -NXTLINE
-LINES
* ASSIGN THE NUMBER OF LINES IN THE MESSAGE TO THE VARIABLE NUMLINES
GETMSIZE NUMLINES
&GOTO -COUNTER1
-GOIN
&WRITE COMMAND LIST AUTHTOTE ENDED BY "GO" COMMAND -- COUNT INCOMPLETE
&EXIT
-TIMEOUT
&WRITE NO RESPONSE WITHIN 60 SECONDS. COMMAND LIST TERMINATING
&EXIT
-ENDIT
* DISPLAY THE COUNT OF MESSAGES ASSIGNED TO AUTH RECEIVERS
&WRITE &COUNT MESSAGES ASSIGNED TO SPECIFIC AUTH RECEIVERS
&EXIT

```

Figure 12. Command List Using Multiline Messages—NetView Command List Language Example (Part 2 of 2)

GLOBALV (REXX)

Introduction

The GLOBALV command enables you to define, put, and get global variables in REXX and NetView command lists. The GLOBALV command also enables you to save global variables in a VSAM database. You can restore saved global variables if NetView or a task is stopped and restarted, or erase (purge) saved global variables from external storage. Global variables enable multiple command procedures, regardless of their language, to share a common set of values.

When used with SAVE, RESTORE, and PURGE, the GLOBALV command is a long-running command processor that you enter from any command list. When GLOBALV is a long-running command processor, you cannot use it in an installation exit or user-written exit.

There are two types of global variables:

- Task
- Common

Task global variables are accessible only to the NetView task under which they were defined or set. *Common* global variables are accessible to any NetView task.

When you *define* a variable as global with the GLOBALV command, NetView makes the locally assigned variable globally available to command processors executing from the command list. You must use GLOBALV to define a global variable before some command processors (such as VIEW) can use the global variable value. For example, once you use GLOBALV to define a global variable, the VIEW command processor can put or get values for the variable.

Note: For more information about the VIEW command, refer to the *Tivoli NetView for z/OS Customization Guide*.

When you *put* a global variable using the GLOBALV command, NetView places that variable in a global variable dictionary. If the variable is a task global variable, it is stored in the global variable dictionary for the particular task under which the command list is running. Only command procedures running under the same task can access that task's global variable dictionary.

When you *get* a global variable using the GLOBALV command, NetView gets the global variable from the appropriate dictionary and places it in a REXX or NetView command list language variable of the same name. You can then perform arithmetic operations or other manipulations on the variable to suit the needs of your command list without affecting the value of the global variable in the dictionary.

When you *save* a global variable using the GLOBALV command, NetView retrieves the global variable from the appropriate dictionary and places it in external storage. NetView or the task can then be terminated without affecting the global variable in external storage.

When you *restore* a global variable using the GLOBALV command, NetView retrieves the variable value from external storage. When the variable value is returned, it is placed in the appropriate dictionary.

When you *purge* a global variable using the GLOBALV command, NetView purges the variable from external storage. The GLOBALV purge does not affect the variable in the global dictionary.

Return Codes

Table 6. GLOBALV Return Code Summary

Return Code	Meaning												
-5	Operator canceled or reset the procedure. This applies only if the command processor was invoked by a REXX procedure.												
0	Command completed successfully.												
4	Not called from a command list, REXX, or high-level language procedure.												
24	Nonzero return code from macro DSIGET. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIGET macro.												
52	Function is not valid. Check the spelling of the function and synonym.												
88	Incorrect length for variable name.												
144	No function or variable defined.												
156	Local dictionary not found or no user variables are defined.												
268	None of the variables requested to be restored matched what is in the VSAM directory.												
292	Not called by operator station task (OST), primary POI task (PPT), or NetView-to-NetView task (NNT).												
304	Reset request from user.												
308	The DST ended with an error.												
540	Nonvalid character in variable name.												
544	Odd count in double-byte character set (DBCS) shift-out and shift-in character.												
1000 + x	Nonzero return code of x from DSIMQS. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIMQS macro.												
4000 + x	Nonzero return code of x from DSIPUSH. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIPUSH macro.												
11000 + x	Nonzero return code of x from DSIKVS. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIKVS macro.												
13000 + x	Nonzero return code of x from DSILCS. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSILCS macro.												
14000 + x	Nonzero return code of x. Return code values are: <table border="1" data-bbox="609 1654 1242 1831"> <thead> <tr> <th>Code</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>4</td> <td>Nonvalid variable name</td> </tr> <tr> <td>12</td> <td>Insufficient storage</td> </tr> <tr> <td>20</td> <td>Value length limit exceeded</td> </tr> <tr> <td>28</td> <td>No command procedure related to current action</td> </tr> <tr> <td>32</td> <td>Data was truncated</td> </tr> </tbody> </table>	Code	Meaning	4	Nonvalid variable name	12	Insufficient storage	20	Value length limit exceeded	28	No command procedure related to current action	32	Data was truncated
Code	Meaning												
4	Nonvalid variable name												
12	Insufficient storage												
20	Value length limit exceeded												
28	No command procedure related to current action												
32	Data was truncated												
16000 + x	Nonzero return code of x from DSIPAS. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIPAS macro.												

Table 6. GLOBALV Return Code Summary (continued)

Return Code	Meaning
25000 + x	Nonzero return code of x from DSIPRS. Refer to <i>Tivoli NetView for z/OS Customization: Using Assembler</i> for more information about the DSIPRS macro.
25600 + a	$((100 + \text{Major RC}) * 256) + \text{Minor RC}$

Note: Major return codes (RCs) and minor return codes are returned when there is a VSAM failure. Refer to the authorized operator for messages AAU020I, AAU022I, and CNM475I.

Usage Notes

Considering the following when using the GLOBALV command:

- In REXX and high-level languages, the global variable name can be 1–31 alphanumeric characters. Valid alphanumeric characters are:

A–Z 0–9 . # @ ¢ \$ _ ! ?

The first character cannot be a number or a period.

Notes:

1. GLOBALV does not support lower case characters in variable names. The only exception is when lower case characters are specified after the first period in a stem variable. Lower case characters after the first period are supported.

When passing stem variables to GLOBALV, make sure the stem itself is in upper case within the quotes and that the GLOBALV command is executed in the NETVASIS address command environment.

By default, NetView translates all commands, keywords, and values to upper case. For more information, refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language*.

2. If you want global variables you create in a REXX command list to be accessible to command lists written in the NetView command list language, do not use the following characters in the global variable name because the NetView command list language does not allow these characters:

. _ ¢ ! ?

- In NetView command list language, the global variable name can be 1–11 alphanumeric characters. Valid alphanumeric characters are:

A–Z 0–9 . # @ \$

The first character cannot be a number or a period.

- When using SAVE, RESTORE, or PURGE, you can use an asterisk (*) at the end of the character string to include all variables beginning with the same characters. For example, the following string includes all variables beginning with ABC, such as ABC1, ABC2, and ABCXYZ:

ABC*

To use SAVE, RESTORE, and PURGE to access the VSAM database, the DSISVRT DST task must be properly defined and started. Otherwise, you will receive return code 1008.

Notes:

1. Refer to “Example: Using the Asterisk” on page 284 for information about preventing the use of a wildcard.
 2. For information about defining DSISVRT DST, refer to the *Tivoli NetView for z/OS Administration Reference*.
- If you specify more than one global variable on the GLOBALV command, the variable names must be delimited by commas or blanks.
 - The value of the global variable can be 255 characters long. For double-byte character sets (DBCS), the maximum number of double-byte characters between the shift-out (X'0E') and shift-in (X'0F') control characters is 126.
 - You can give global variables a numerical value between -2147483647 and 2147483647.
If you go outside these limits:
 - For REXX, they are converted to an exponential notation, making them unusable by NetView command list language command lists.
 - For NetView command list language, they are treated as character strings.

Examples

Example: Using the Asterisk

GLOBALV keywords SAVEC, SAVET, RESTOREC, RESTORET, PURGEC and PURGET all support the use of the asterisk (*) wildcard with a variable. However, there is a potential for severe system performance degradation or unintentional loss of data when the asterisk is specified. To minimize that potential, NetView provides a means of restricting usage of the asterisk to pre-authorized users. A command authorization check of the user's authority level is performed whenever a single asterisk is encountered in conjunction with one of the GLOBALV keywords listed above, as shown in the following example:

```
GLOBALV SAVEC *
```

The check involves the asterisk symbol (*) mapping to the keyword ASTERISK, then determining whether the user is authorized to use the ASTERISK keyword.

To prevent operators from using the wildcard, protect the word ASTERISK as a keyword on the GLOBALV command with your command security.

Note: For information about command security, refer to the *Tivoli NetView for z/OS Security Reference*.

Command List Examples that Put, Get, Save, Restore, Purge, and Update

Figure 13 and Figure 14 are examples of REXX command lists that show how to put, get, save, restore, purge, and update a task global variable. The first command list is named GLOBV1, and it executes the nested command list UPDATE1.

```

/* GLOBV1 Command List */
/* OP1TSK1 - OPER1's Task Variable 1 */
/* OP1COM1 - OPER1's Common Variable 1 */
/* The following assignment statements give the REXX */
/* variables, OP1TSK1 and OP1COM1 initial values. */
OP1TSK1 = '5'
OP1COM1 = '1'
/* The following statement PUTS a task global variable */
/* named OP1TSK1. The value of the task global variable */
/* is 5 because the value of the REXX variable OP1TSK1 */
/* is 5. */
'GLOBALV PUTT OP1TSK1'
/* The following statement PUTS a common global variable */
/* named OP1COM1. The value of the common global variable */
/* is 1 because the value of the REXX variable OP1COM1 */
/* is 1. */
'GLOBALV PUTC OP1COM1'
/* The following statement executes a nested Command List */
/* named UPDATE1. */
UPDATE1
/* The following statement GETS the value of task global */
/* variable OP1TSK1 and places it in a REXX variable */
/* named OP1TSK1. */
'GLOBALV GETT OP1TSK1'
/* The following statement GETS the value of common global */
/* variable OP1COM1 and places it in a REXX variable */
/* named OP1COM1. */
'GLOBALV GETC OP1COM1'
/* The following statements write out the values of the */
/* REXX variables OP1TSK1 and OP1COM1. OP1TSK1 will be 10 */
/* and OP1COM1 will be 6. */
SAY 'OP1TSK1 = 'OP1TSK1
SAY 'OP1COM1 = 'OP1COM1
/* The following statement will change the value of the */
/* REXX variables OP1TSK1 and OP1COM1 to 0. */
OP1TSK1 = '0'
OP1COM1 = '0'
/* The following statement will save the task global */
/* variable OP1TSK1 to the VSAM Save/Restore data base */
/* with DST, DSISVRT. OP1TSK1 will be saved to VSAM with */
/* a value of 10, since the task global variable value of */
/* OP1TSK1 is 10. */

```

Figure 13. GLOBV1 Command List (Part 1 of 2)

```

'GLOBALV SAVET OP1TSK1'
/* The following statement will save the common global */
/* variable OP1COM1 to the VSAM Save/Restore data base. */
/* OP1COM1 will be saved to VSAM with a value of 6, since */
/* the common global variable value of OP1COM1 is 6. */
'GLOBALV SAVEC OP1COM1'
/* The following statement will again put the task global */
/* variable name OP1TSK1. The value of OP1TSK1 is 0 */
/* because the REXX variable OP1TSK1 is 0. */
'GLOBALV PUTT OP1TSK1'
/* The following statement will again put the common */
/* global variable name OP1COM1. The value of OP1COM1 is */
/* 0 because the REXX variable OP1COM1 is 0. */
'GLOBALV PUTC OP1COM1'
/* The following statements will restore the task and */
/* common global variables OP1TSK1 and OP1COM1. The */
/* restore will change the global variables, but not the */
/* REXX variable values. */
'GLOBALV RESTORET OP1TSK1'
'GLOBALV RESTOREC OP1COM1'
/* The following statements will GET the values of the */
/* task and common global variables and place the values */
/* in the REXX variables OP1TSK1 and OP1COM1. */
'GLOBALV GETT OP1TSK1'
'GLOBALV GETC OP1COM1'
/* The following statements write out the values of the */
/* REXX variables OP1TSK1 and OP1COM1. OP1TSK1 will be 10 */
/* and OP1COM1 will be 6. */
SAY 'OP1TSK1 = 'OP1TSK1
SAY 'OP1COM1 = 'OP1COM1
/* The following statements will purge the entry in the */
/* VSAM data base for task global variable OP1TSK1 and */
/* command global variable OP1COM1. REXX variable OP1TSK1 */
/* and OP1COM1 will not be affected. Global dictionary */
/* variables OP1TSK1 and OP1COM1 will not be affected */
/* either. */
'GLOBALV PURGET OP1TSK1'
'GLOBALV PURGEC OP1COM1'
EXIT

```

Figure 13. GLOBV1 Command List (Part 2 of 2)


```

/* UPDATE1 Command List */
/* The following statement gets the value of task global */
/* variable OP1TSK1 and puts it into a REXX variable named */
/* OP1TSK1. */
'GLOBALV GETT OP1TSK1'
/* The following statement gets the value of common global */
/* variable OP1COM1 and puts it into a REXX variable named */
/* OP1COM1. */
'GLOBALV GETC OP1COM1'
/* The following statement checks the value of the REXX */
/* variables OP1TSK1 and OP1COM1 to determine if a variable */
/* exists. If no variable exists a NULL value is returned. */
IF OP1TSK1 = '' THEN OP1TSK1 = 0
/* Increment the REXX variable by 5. */
ELSE
  OP1TSK1 = OP1TSK1 + 5

IF OP1COM1 = '' THEN OP1COM1 = 0

/* Increment the REXX variable by 5. */
ELSE
  OP1COM1 = OP1COM1 + 5
/* The following statements PUT the values of the task and */
/* common global variables OP1TSK1 and OP1COM1. */
'GLOBALV PUTT OP1TSK1'
'GLOBALV PUTC OP1COM1'
EXIT

```

Figure 14. UPDATE1 Command List

Command list GLOBV1 creates two REXX variables. OP1TSK1 has a value of 5 and OP1COM1 has a value of 1. A GLOBALV PUTT instruction is issued to set a task global variable named OP1TSK1, and a GLOBALV PUTC is issued to set a common global variable named OP1COM1. Since a REXX variable already exists with the name of OP1TSK1 and has a value of 5, task global variable OP1TSK1 is also set to a value of 5. The same is true of the REXX variable OP1COM1, and the common global variable OP1COM1 is set to the value of 1. GLOBV1 then activates a nested command list named UPDATE1.

Command list UPDATE1 issues a GLOBALV GETT instruction to get the current value of task global variable OP1TSK1 into a REXX variable named OP1TSK1. UPDATE1 issues a GLOBALV GETC instruction to get the current value of common global variable OP1COM1 into a REXX variable named OP1COM1. The values of OP1TSK1 and OP1COM1 are checked to determine if they exist by testing for null values. If the value returned is null, the variable value is set to 0. The value of REXX variable OP1TSK1 is 5, and it is incremented by 5, making its current value 10. The value of REXX variable OP1COM1 is 1, and it is incremented by 5, making its current value 6. The GLOBALV PUTT instruction updates the value of task global variable OP1TSK1 in the task global variable dictionary. The GLOBALV PUTC instruction updates the value of common global variable OP1COM1 in the common global variable dictionary.

When UPDATE1 completes execution, control is returned to the GLOBV1 command list. GLOBV1 contains a GLOBALV GETT and a GLOBALV GETC instruction to get the current values of task global variable OP1TSK1 and common global variable OP1COM1, and places the values in the REXX variables OP1TSK1 and OP1COM1. The value of REXX variable OP1TSK1 is 10 and the value of OP1COM1 is 6. The SAY instruction displays the current values of the REXX variables OP1TSK1 and OP1COM1.

The REXX variables OP1TSK1 and OP1COM1 are then set to 0. This does not change the values of the task global variable OP1TSK1 or the common global variable OP1COM1. The GLOBALV SAVET instruction writes the value of the task global variable OP1TSK1 to the VSAM Save/Restore database. The current value of task global variable OP1TSK1 is 10, which is stored to VSAM. The GLOBALV SAVEC instruction writes the value of the common global variable OP1COM1 to the VSAM Save/Restore database. The current value of common global variable OP1COM1 is 6, which is stored to VSAM.

The task and command global variables OP1TSK1 and OP1COM1 are set by issuing the GLOBALV PUTT OP1TSK1 and the GLOBALV PUTC OP1COM1 instructions. Currently both the task global variable OP1TSK1 and the common global variable OP1COM1 have a value of 0. The GLOBV1 command list now restores the values for OP1TSK1 and OP1COM1 that were stored to the VSAM database. GLOBALV RESTORET OP1TSK1 takes the value of task global variable OP1TSK1 that was stored to VSAM previously and replaces the current value of task global variable OP1TSK1, leaving the REXX variable OP1TSK1 unchanged. GLOBALV RESTOREC OP1COM1 takes the value of common global variable OP1COM1 that was stored to VSAM previously and replaces the current value of common global variable OP1COM1, leaving the REXX variable OP1COM1 unchanged.

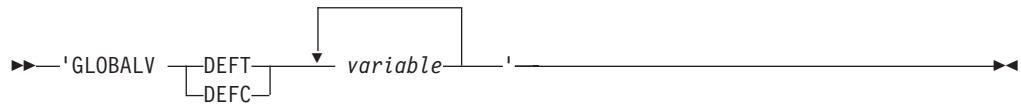
GLOBV1 issues a GLOBALV GETT instruction to get the current value of task global variable OP1TSK1 into a REXX variable named OP1TSK1. If the value of OP1TSK1 is equal to null, there was no entry in the VSAM database for task global variable OP1TSK1. The same is true for common global variable OP1COM1.

The variables OP1TSK1 and OP1COM1 are deleted from the VSAM database by issuing the GLOBALV PURGET and GLOBALV PURGEC instruction. The GLOBALV PURGET and GLOBALV PURGEC instructions have no effect on the REXX variables OP1TSK1 and OP1COM1, the task global variable OP1TSK1, or the command global variable OP1COM1.

GLOBALV DEF (REXX)

Syntax

GLOBALV DEFTIDEFC



Purpose of Command

Use the GLOBALV DEFT or DEFC command to define a task or common global variable from a REXX or NetView command list.

The GLOBALV DEFT or DEFC command makes the specified task or common global variable available to command processors executing from the command list. For example, once you use GLOBALV DEFT or DEFC to define a task or common global variable, the VIEW command processor can put or get values for the variable.

Note: For more information about the VIEW command, refer to the *Tivoli NetView for z/OS Customization Guide*.

For NetView command list language, the GLOBALV DEFT or DEFC command is equivalent to &TGLOBAL and &CGLOBAL when you use it in a command list. After you issue a GLOBALV DEFT or DEFC, the command list references and sets the global variable. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more information about &TGLOBAL and &CGLOBAL.

In REXX command lists, the DEFT or DEFC operations creates an entry for the variable, but the value for that global variable is not changed by the actions of the REXX exec unless you issue a GLOBALV GET or PUT.

Operand Descriptions

DEFT or DEFC

Indicates that the specified variable name or names are to be defined as task (DEFT) or common (DEFC) global variables. Any prior definitions of the specified variable names are overridden, but variable values are not affected.

variable

For REXX, specifies the 1- to 31-character name or names of the task or common global variables to be defined. For NetView command list language, specifies the 1-11 character name or names of the task or command global variables to be defined. See “Usage Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Coding Variables for GLOBALV DEF

Use any of the following examples to code variables for the GLOBALV DEFT or DEFC command:

```
'GLOBALV DEFT VAR1 VAR2 VAR3'
```

```
'GLOBALV DEFT VAR1,VAR2,VAR3'
```

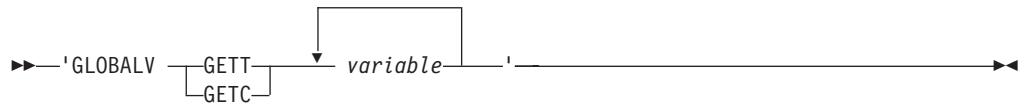
```
'GLOBALV DEFC VAR1 VAR2 VAR3'
```

```
'GLOBALV DEFC VAR1,VAR2,VAR3'
```

GLOBALV GET (REXX)

Syntax

GLOBALV GETTIGETC



Purpose of Command

Use the GLOBALV GETT or GETC command to access a task or common global variable in a REXX or NetView command list.

When a GLOBALV GETT or GETC command is processed, NetView gets the current value of the specified global variable from the task or common global variable dictionary and places the value in a REXX or NetView command list language variable of the same name. If a variable with the same name does not already exist, the variable is created and its value is set to the task value currently assigned to the global variable.

If no task or common global variable with the specified variable name exists, the variable is not created. Any attempt to retrieve the variable causes a null value to be returned.

Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for examples about using common global variables. You can also use the `&TGLOBAL(name)` function to get task global variables and the `&CGLOBAL(name)` function to get common global variables.

The function of the GLOBALV GET command is influenced by any previous `&TGLOBAL`, `&CGLOBAL`, or GLOBALV defined command issued within the same NetView command list for the same named variable. For example, if a variable has been defined as a task global variable (`&TGLOBAL variable`), a subsequent GLOBALV GETC *variable* copies the common global value to the task global variable instead of to a NetView command list language variable. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more information about `&TGLOBAL` and `&CGLOBAL`.

The command lists in “Command List Examples that Put, Get, Save, Restore, Purge, and Update” on page 284 show how to put, get, and update a task global variable.

Operand Descriptions

GETT or GETC

Indicates that the task or common global variables with the specified variable names should be retrieved.

variable

For REXX, specifies the 1- to 31-character name or names of the task or common global variables to be retrieved. For NetView command list language, specifies the 1-11 character name of the task or common global variables to be retrieved. See “Usage

Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Coding Variables for GLOBALV GET

Use any of the following examples to code variables for the GLOBALV GETT or GETC command:

```
'GLOBALV GETT VAR1 VAR2 VAR3'
```

```
'GLOBALV GETT VAR1,VAR2,VAR3'
```

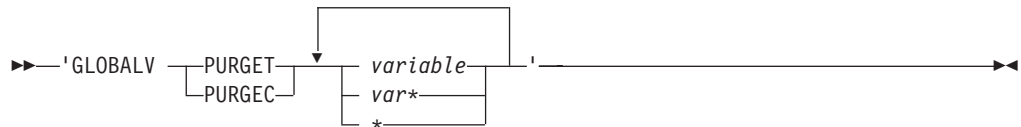
```
'GLOBALV GETC VAR1 VAR2 VAR3'
```

```
'GLOBALV GETC VAR1,VAR2,VAR3'
```

GLOBALV PURGE (REXX)

Syntax

GLOBALV PURGET|PURGEC



Purpose of Command

Use the GLOBALV PURGET or PURGEC command to purge task or common global variables from external storage.

When you use an asterisk symbol, the entire global dictionary is purged from external storage. If you are purging a large number of variables, consider how much CPU time is required for the I/O. See “Example: Using the Asterisk” on page 284 for information about preventing operators from using the wildcard.

Operand Descriptions

PURGET or PURGEC

Indicates that the task or common global variables with the specified variable types or names should be purged from external storage.

variable [,...]

For REXX, specifies the 1- to 31-character name or names of the task or common global variables to be purged from external storage. For NetView command list language, specifies the 1- to 11-character name or names of the task or common global variables to be purged from external storage. See “Usage Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Coding Variables for GLOBALV PURGE

The following examples show how you can code variables for the GLOBALV PURGET or PURGEC command:

Use either of the following examples to purge all variables from external storage:

```
'GLOBALV PURGET *'  
'GLOBALV PURGEC *'
```

Use either of the following examples to purge all variables beginning with VAR1 and LU and the variable ABC from external storage:

```
'GLOBALV PURGET VAR1* ABC LU*'
```

```
'GLOBALV PURGEC VAR1* ABC LU*'
```

Use either of the following examples to purge all of the listed variables from external storage:

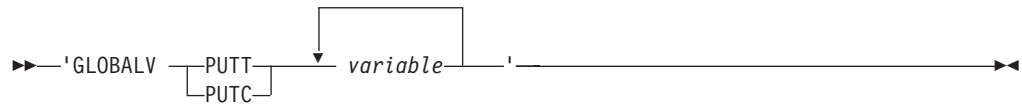
```
'GLOBALV PURGET VAR1 VAR2 VAR3'
```

```
'GLOBALV PURGEC VAR1 VAR2 VAR3'
```

GLOBALV PUT (REXX)

Syntax

GLOBALV PUTTIPUTC



Purpose of Command

Use the GLOBALV PUTT or PUTC command to set a task or common global variable from a REXX or NetView command list. The GLOBALV PUTT or PUTC command creates a task or common global variable with the specified variable name and places it in the task or common global variable dictionary.

When a GLOBALV PUTT or PUTC command is processed, NetView determines if a REXX or NetView command list language variable already exists with the specified variable name. If a variable with the same name already exists, the task or common global variable is created and its value is set to the value currently assigned to the variable.

To delete the value of a task or common global variable, set the value of the variable name to NULL (") and use the GLOBALV PUT command to replace the value. A GLOBALV GET request for the variable causes a null value to be returned.

If a NetView command list language variable with the specified variable name does not exist, a task or common global variable with the specified name is not created. Any attempt to retrieve the variable causes a null value to be returned.

The function of the GLOBALV PUT command is influenced by any previous &TGLOBAL, &CGLOBAL, or GLOBALV defined command issued within the same NetView command list for the same named variable. For example, if a variable has been defined as common (&CGLOBAL *variable*), a subsequent GLOBALV PUTT *variable* moves the common global value to the task global variable by that name instead of moving a value from the NetView command list language variable. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more information about &TGLOBAL and &CGLOBAL.

If a task or common global variable already exists with the specified variable name, the value of the task or common global variable is updated in the global variable dictionary.

Note: If you have more than one command list running under different tasks and accessing the same common global variable, the last value to which the variable is set is the value that is retrieved by any command list accessing the variable.

For example, a command list accesses a common global variable and, before that command list updates the variable, another command list running under a

different task accesses the variable. If both command lists update the variable, the variable assumes the value given to it by the command list that updates it last.

To avoid problems with having a common global variable updated by different command procedures at the same time, you can use PIPE VARLOAD, which can compare and set a variable at one time. See the section about UPDCGLOB (CNME1080) for an example use of VARLOAD.

Refer to *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for examples about using common global variables. Refer to *Tivoli NetView for z/OS Customization: Using Pipes* for usage of VARLOAD.

The command lists in “Command List Examples that Put, Get, Save, Restore, Purge, and Update” on page 284 show how to put, get, and update a task global variable.

Operand Descriptions

PUTT or PUTC

Indicates that task or common global variables with the specified variable names should be put into the task or common global variable dictionary.

variable

For REXX, specifies the 1 – 31-character name or names of the task or common global variables to be put. For NetView command list language, specifies the 1 – 11 character name or names of the task or command global variables to be put. See “Usage Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Coding Variables for GLOBALV PUT

Use any of the following examples to code variables for the GLOBALV PUTT or PUTC command:

```
'GLOBALV PUTT VAR1 VAR2 VAR3'
```

```
'GLOBALV PUTT VAR1,VAR2,VAR3'
```

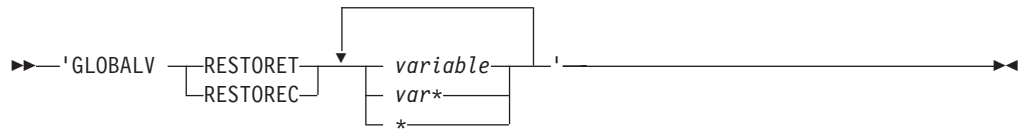
```
'GLOBALV PUTC VAR1 VAR2 VAR3'
```

```
'GLOBALV PUTC VAR1,VAR2,VAR3'
```

GLOBALV RESTORE (REXX)

Syntax

GLOBALV RESTORE|RESTOREC



Purpose of Command

Use the GLOBALV RESTORE or RESTOREC command to restore a task or common global variable in a REXX or NetView command list.

If you indicate RESTORE or RESTOREC for a variable that does not exist in external storage, the variable is restored in the global dictionary with a null value. This condition applies only if you specify the variables explicitly, not when you list all variables implicitly using an asterisk (*).

When you use an asterisk symbol, the entire global dictionary is restored from external storage or the VSAM file. If you are restoring a large number of variables, consider how much CPU time is required for the I/O. See “Example: Using the Asterisk” on page 284 for information about preventing operators from using the wildcard.

Operand Descriptions

RESTORE or RESTOREC

Indicates that the task or common global variables with the specified variable types or names should be restored from external storage to the global dictionary.

Note: You can code RESTORE and RESTOREC using parameter synonyms RESTT or RESTC, respectively.

variable

For REXX, specifies the 1- to 31-character name or names of the task or common global variables to be restored to the global dictionary. For NetView command list language, specifies the 1-11 character name or names of the task or common global variables to be restored to the global dictionary. See “Usage Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Coding Variables for GLOBALV RESTORE

The following examples show how you can code variables for the GLOBALV RESTORET or RESTOREC commands.

Use either of the following examples to restore all variables in the task or common global variable dictionary from external storage:

```
'GLOBALV RESTORET *'  
'GLOBALV RESTOREC *'
```

Use either of the following examples to restore all variables beginning with VAR1 and LU and the variable ABC in the task or common global variable dictionary from external storage:

```
'GLOBALV RESTORET VAR1* ABC LU*'  
'GLOBALV RESTOREC VAR1* ABC LU*'
```

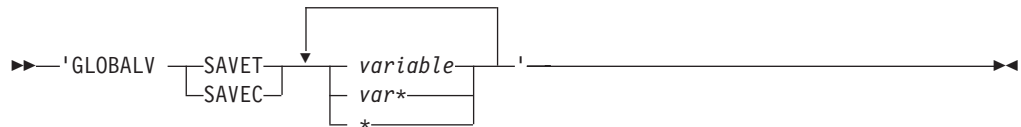
Use any of the following examples to restore all of the listed variables in the task or common global variable dictionary from external storage.

```
'GLOBALV RESTORET VAR1 VAR2 VAR3'  
'GLOBALV RESTORET VAR1,VAR2,VAR3'  
'GLOBALV RESTOREC VAR1 VAR2 VAR3'  
'GLOBALV RESTOREC VAR1,VAR2,VAR3'
```

GLOBALV SAVE (REXX)

Syntax

GLOBALV SAVETISAVEC



Purpose of Command

Use the GLOBALV SAVET or SAVEC command to save a task or common global variable in a REXX or NetView command list, or VSAM database.

If you indicate SAVET or SAVEC for a variable that does not exist in the global dictionary, the variable is saved in external storage with a null value. This condition applies only if you specify the variables explicitly, not when you list all variables implicitly using an asterisk (*).

An asterisk (*) causes the global directory to be saved in external storage or to the VSAM file. To avoid performance degradation if you are saving a large number of variables, consider how much storage space and central processing unit (CPU) time are required for the input/output (I/O). See “Example: Using the Asterisk” on page 284 for information about preventing operators from using the wildcard.

Operand Descriptions

SAVET or SAVEC

Indicates that the task or common global variables with the specified variable names should be saved from the global dictionary to external storage.

variable

For REXX, specifies the 1- to 31-character name or names of the task or common global variables to be saved to external storage. For NetView command list language, specifies the 1-11 character name or names of the task or common global variables to be saved to external storage. See “Usage Notes” on page 283 for a list of the characters you can use for a variable name. Variables can be separated by either a space or a comma.

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Coding Variables for GLOBALV SAVE

The following examples show how you can code variables for the GLOBALV SAVET or SAVEC command.

Use either of the following examples to save all variables in the task or common global variable dictionary to external storage:

```
'GLOBALV SAVET *'
```

```
'GLOBALV SAVEC *'
```

Use either of the following examples to save all variables beginning with VAR1 and LU and the variable ABC in the task or common global variable dictionary to external storage:

```
'GLOBALV SAVET VAR1* ABC LU*'
```

```
'GLOBALV SAVEC VAR1* ABC LU*'
```

Use any of the following examples to save the variables VAR1, VAR2, and VAR3 in the task or common global variable dictionary to external storage:

```
'GLOBALV SAVET VAR1 VAR2 VAR3'
```

```
'GLOBALV SAVET VAR1,VAR2,VAR3'
```

```
'GLOBALV SAVEC VAR1 VAR2 VAR3'
```

```
'GLOBALV SAVEC VAR1,VAR2,VAR3'
```

MSGREAD (REXX)

Syntax

MSGREAD

►► 'MSGREAD' ◄◄

Purpose of Command

The MSGREAD instruction causes NetView to read a trapped message from the message queue. The command list can then take action based on the message. Refer to the TRAP and WAIT instructions for information about using these with the MSGREAD instruction.

When the MSGREAD instruction is issued, the oldest message in the queue is read and removed. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for information about using TRAP in nested REXX command lists and the functions set by MSGREAD.

Note: The MSGREAD command pre-dates NetView pipelines. You can use the PIPE command to solve this type of problem. For more information about the PIPE command and its stages, refer to *Tivoli NetView for z/OS Customization: Using Pipes*.

Return Codes

MSGREAD sets the value of the return code (RC) to indicate the results of processing, as follows:

Return Code	Meaning
-2	Syntax error
-1	DSIGET failure
0	Successful completion
4	No messages in queue

Restrictions

For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Using the MSGREAD Instruction

The following is an example of a REXX command list named ACTLU. ACTLU issues a VTAM command to activate a node specified by the user. Messages sent as the result of the activated command are trapped, and the operator is notified of the result of the command list. The comments in the example explain how the command list works:

```

/*****
/* ACTLU COMMAND LIST */
/*
/* FUNCTION : TO ACTIVATE A VTAM NODE. */
/* INPUT : 1 PARAMETER, THE NAME OF THE NODE. */
/*****
SIGNAL ON ERROR /* SIGNAL IF ERROR OCCURS */
IF MSGVAR(1) = '' THEN /* NO FIRST PARAMETER ? */
DO /* THEN ISSUE REQUEST */
SAY 'PLEASE ENTER "GO NODENAME"', /* REQUEST NODENAME FROM USER */
'TO CONTINUE OR "GO STOP" TO', /* OR, ALLOW USER TO STOP */
'STOP'
PARSE PULL NODE /* NODE = NODENAME OR STOP */
END /* THEN ISSUE REQUEST */
ELSE /* FIRST PARAMETER EXISTS */
NODE = MSGVAR(1) /* ASSUME IT IS A NODE NAME */
IF NODE='STOP' THEN /* IF USER DID NOT CHOOSE STOP */
DO /* PROCESS NODENAME */
'TRAP AND SUPPRESS ONLY MESSAGES IST* ' /* TRAP ALL VTAM MSGS */
'V NET,ACT,ID='NODE /* ISSUE VTAM ACTIVATE FOR NODE */
RCSAVE=RC /* SAVE RETURN CODE IN RCSAVE */
'WAIT 30 SECONDS FOR MESSAGES' /* WAIT FOR 30 SECONDS */
SELECT
WHEN (EVENT()='M') THEN /* OUT OF WAIT - IS THERE A MSG? */
DO /* PROCESS TRAPPED MESSAGE */
'MSGREAD' /* READ IN 1ST MESSAGE */
DO WHILE (RC=0) /* IF RC=0 THEN NO MORE MSGS */
SELECT /* DETERMINE WHICH MESSAGE HIT */
WHEN (MSGID() = 'IST061I') THEN /* NODE NOT FOUND */
SAY '==> LU UNKNOWN', /* INFORM USER */
'TO YOUR VTAM <==='
WHEN (MSGID() = 'IST093I') THEN /* NODE NOW ACTIVE */
SAY '==> TERMINAL 'MSGVAR(1)' NOW', /* INFORM USER */
MSGVAR(2) '<==='
OTHERWISE /* IGNORE THE VTAM MESSAGE */
IF RCSAVE=0 THEN
'WAIT CONTINUE' /* CONTINUE WAITING */
ELSE
DO
SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED ? */
'ACTIVATE COMMAND' /* INFORM USER */
SAY MSGSTR() /* DISPLAY MESSAGE TEXT */
END

```

Figure 15. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 1 of 2)


```

        END                                /* OF SELECT FOR IST0611/IST093I */
        'MSGREAD'                          /* READ IN THE NEXT MESSAGE      */
    END                                    /* DO WHILE RC=0, LOOP BACK      */
END                                        /* PROCESS TRAPPED MESSAGE DO    */
                                           /* OUT OF DO WHILE                */

    WHEN (EVENT()='E'|RCSAVE~=0) THEN /* CHECK FOR ERROR OR           */
                                           /* TIME-OUT EVENTS               */
        SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED?          */
        'ACTIVATE COMMAND' /* INFORM USER                  */
    WHEN (EVENT()='T') THEN /* WAIT TIME-OUT ENCOUNTERED?  */
        SAY 'NO RESPONSE TO', /* INFORM USER                  */
        'ACTLU CLIST FOR 'NODE
    OTHERWISE; /* NO-OP                        */
END /* OF SELECT FOR ERROR/TIME-OUT */
END /* IF NODE~='STOP' PROCESSING */
EXIT 0;
ERROR:
IF RC = 4 THEN
    EXIT
ELSE
    SAY 'ERROR OCCURRED. RETURN CODE IS ' RC
EXIT -1; /* END COMMAND LIST FOR ERROR */

```

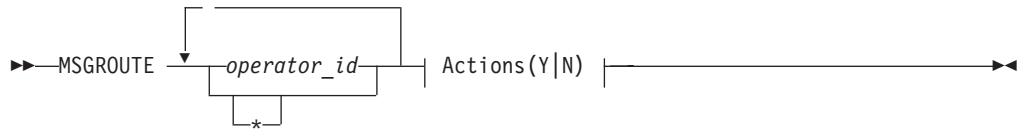
Figure 15. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 2 of 2)

Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more examples of REXX command lists for NetView.

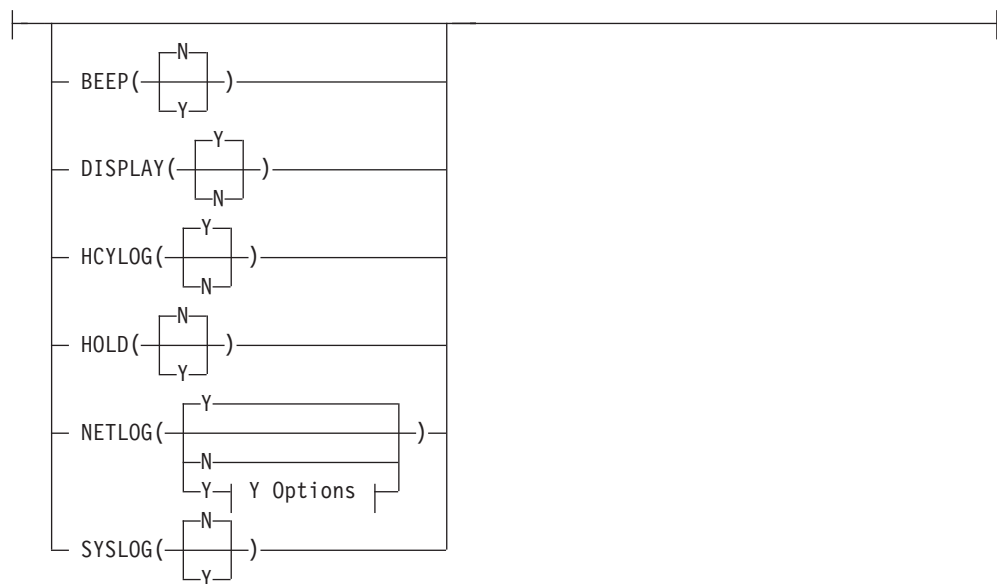
MSGROUTE (REXX)

Syntax

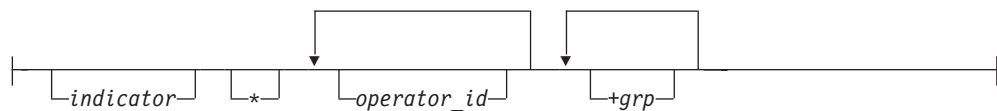
MSGROUTE



Actions(Y|N):



Y Options:



Purpose of Command

The MSGROUTE commands routes its "current message" to specified operators or operator groups. Some message attributes may be changed. The "current message" may be a message that caused the automation table to drive the command list that contains the MSGROUTE command, or it may be a message which flowed to MSGROUTE in a NetView pipeline.

After MSGROUTE routes a message, NetView automation processes the message unless it was previously automated. Messages sent by the MSGROUTE command are queued to the normal priority message queue of the target task or tasks.

Note: When you route a copy of a message to a cross-domain NetView, the message drives the message table on the receiving system. That is, the copies work the same as the copies you create using the ASSIGN command.

Operand Descriptions

BEEP	Determines whether an audible alarm is sounded when the message is displayed. The default is BEEP(N).
DISPLAY	Determines whether the message is displayed. The default is DISPLAY(Y).
HCYLOG	Determines whether the message is placed in the hardcopy log. The default is HCYLOG(Y).
HOLD	Determines whether the message is held on the operator's panel after it is displayed. The default is HOLD(N).
NETLOG	Determines whether the message is placed in the NetView log and whether the message activates a status monitor important message indicator for specified operators or groups of operators. The default is NETLOG(Y).

Specifying NETLOG(Y) can include the following variables:

- * Indicates that the *cmdstring* or message is routed to the current operator task (the task where the message is intercepted for automation checking). If the operator task is CNMCSSIR, message routing can differ. Refer to the *Tivoli NetView for z/OS Administration Reference* for more information about CNMCSSIR and NetView automation.

+grp

Specifies the group identifier of the groups of operators for whom the message is logged as important. You can code as many group identifiers as needed. The maximum length of a group identifier is 8 characters, and it must begin with a plus sign (+). Define group identifiers with the ASSIGN command. Refer to the NetView online help for more information about the ASSIGN command.

indicator

Identifies the status monitor important message indicator.

operator_id

Specifies the operator identifier of the operators for whom the message is logged as important. The maximum length of an operator identifier is 8 characters. You can code as many operator identifiers as needed.

An asterisk (*) can be specified for the *operator_id*. If MSGROUTE is issued on a virtual OST (VOST), the asterisk (*) is interpreted as the owning task's operator identifier. On any other task, the asterisk (*) is interpreted as the issuing task's operator identifier. For more information, refer to the *Tivoli NetView for z/OS Administration Reference*.

If the operator is not in the status monitor or a log browse but is logged on, message CNM039I is displayed:

```
CNM039I AN IMPORTANT MESSAGE HAS BEEN LOGGED -  
PLEASE BROWSE THE NETVIEW LOG.
```

If you specify only an *indicator*, the message is logged as important for the authorized receiver. The following example shows how NETLOG is coded with only an *indicator-number*:

```
MSGROUTE OPER1 NETLOG(Y 2)
```

The message is routed to OPER1. The message is also placed in the NetView log and is logged as an important message with a status monitor important message *indicator-number* of 2.

If you specify an *indicator-number* and a list of operators or groups of operators, the message is logged as important for the operators and groups of operators listed. The following example shows how a message is logged when you specify an *indicator-number* and a list of operators and groups of operators:

```
MSGROUTE OPER4 NETLOG(Y 2 * OPER1 +GRP5 OPER6)
```

The message is routed to OPER4. The message is also placed in the NetView log and is defined as an important message with a status monitor important message *indicator-number* of 2. The message activates a status monitor important message indicator for OPER1, OPER6, all of the operators assigned to group +GRP5, and the current operator. If operators OPER1 and OPER6 are also assigned to group +GRP5, each operator receives only one copy of message CNM039I if they are not in the status monitor.

Note: The NETLOG option cannot be specified without an operand between the parentheses. In other words, you cannot specify NETLOG().

operator_id Is the operator identifier of the operators to whom the message is routed. The maximum length of an operator identifier is 8 characters. You can code as many operator identifiers as needed.

You can also specify group identifiers for the groups of operators to whom the message is routed. You must define the group identifier to NetView with the ASSIGN command. The maximum length of a group identifier is 8 characters, and the group identifier must begin with a plus sign (+).

An asterisk (*) can be specified for the *operator_id*. If MSGROUTE is issued on a virtual OST (VOST), the asterisk (*) is interpreted as the owning task's operator identifier. On any other task, the asterisk (*) is interpreted as the issuing task's operator identifier.

The options BEEP through SYSLOG specify the actions NetView should take when routing the message. You can specify any or all of the options.

SYSLOG Determines whether the message is placed in the system log. The default value is SYSLOG(N).

Return Codes

The return code (RC or &RETCODE) indicates the processing results as follows:

Return Code	Meaning
8	Operator or group identifier is not specified, or is greater than 8 characters.
12	The value is not valid for message action.
16	MSGROUTE was not entered from a REXX or NetView command list language command list.
20	MSGROUTE was not issued from a message-driven REXX or NetView command list language command list, or from the NetView automation table.
24	Operator or group identifier, or message action is not within the operator's authority.
28	The storage request failed.
32	There is no active operator to which to route messages.
36	MSGROUTE was issued for a buffer with HDRMTYPE X'10' (MSU data). MSGROUTE does not support routing of MSU data.

Usage Notes

The DEFAULTS and OVERRIDE settings do not affect the settings in MSGROUTE.

PARSEL2R (REXX)

Syntax

PARSEL2R

►►—PARSEL2R *source_variable parsing_template*—◄◄

Purpose of Command

The PARSEL2R command enables you to extract data from the character-string value of a variable and assign the extracted data to one or more variables using a set of rules called a *parsing template*. To parse variables in REXX, use either PARSEL2R or the REXX PARSE instruction. To parse variables with the NetView command list language, use the PARSEL2R command.

Note: Refer to the REXX library for more information about the PARSE instruction.

Operand Descriptions

source_variable

Identifies a command list variable.

In REXX command lists, code *source_variable* within single quotes.

In command lists written in the NetView command list language, you must code the *source_variable* without an ampersand (&). PARSEL2R extracts data from the value of the variable you named as the *source_variable*.

The maximum character length of a source variable name depends on the following:

- 11 When invoked from a command list written in the NetView command list language.
- 31 When invoked from a REXX-format command list and is a NetView global variable name.
- 250 When invoked from a REXX-format command list and is a REXX variable name.

parsing_template

Specifies a list of symbols, patterns, or character selectors, or a combination of these, separated by blanks. PARSEL2R uses this list as a template when parsing the source variable.

Symbols are command list variable names. The rules governing the length of the symbol names in the parsing template are the same as those for the length of the following *source_variable*. In REXX command lists, code command list variable names within single quotes. In command lists written in the NetView command list language, code command list variable names without an ampersand (&). For more information about using symbols, see “Using Symbols in a Parsing Template” on page 309.

Patterns are coded using slashes (/) as delimiters. A pattern is the part of the source variable that you want to match. For more information about using patterns, see “Using Patterns in a Parsing Template” on page 310.

A character selector is coded using an asterisk (*) for each single character you want to extract from the source variable. For more information about using character selectors, see “Using Character Selectors in a Parsing Template” on page 313.

Return Codes

PARSEL2R sets the return code (RC or &RETCODE) to indicate the processing results as follows:

Return Code	Meaning
0	Processing was successful.
8	Storage is not available to continue processing.
100	There are not enough parameters, or not issued from a command list.
104	Input buffer is blank.
108	The command list dictionary lookup of source variable failed.
112	Hexadecimal data in the template is not valid.
116	The command list dictionary update failed.
120	The trailing slash (/) is missing.

Usage Notes

Consider the following topics when using the PARSEL2R command:

Using Symbols in a Parsing Template

The symbols in the parsing template identify command list variables.

In REXX command lists, code command list variables within quotes.

In command lists written in the NetView command list language, code command list variables without the ampersand (&). If only symbols appear in the parsing template, the source variable data is assigned token-by-token from left to right. Tokens are defined as a string of nonblank characters. Tokens in the source variable are separated by one or more blanks. The tokens are assigned to the command list variables you identified with symbols in the parsing template.

The following examples show three lines that use a parsing template containing only symbols:

For REXX:

```
TITLE = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D'  
'PARSEL2R TITLE A1 A2 A3 A4 A5 A6 A7 A8'  
'PARSEL2R TITLE B1 B2 B3'
```

For command list:

```
&TITLE = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D'  
PARSEL2R TITLE A1 A2 A3 A4 A5 A6 A7 A8  
PARSEL2R TITLE B1 B2 B3
```

The resulting values of the variables show how the token-by-token assignment from left to right works.

The following table shows the resulting values for the REXX and NetView command list language variables:

REXX Variable	NetView Variable	Value
A1	&A1	PROCEDURE/ACTION
A2	&A2	NOT
A3	&A3	SUPPORTED:
A4	&A4	SENSE
A5	&A5	=
A6	&A6	X'087D'
A7	&A7	null
A8	&A8	null
B1	&B1	PROCEDURE/ACTION
B2	&B2	NOT
B3	&B3	SUPPORTED: SENSE = X'087D'

Notes:

1. The value of B3 or &B3 is not a single token, but the remainder of the source variable after PARSEL2R has parsed it into the first two symbols.
2. Except for the last variable, leading blanks and trailing blanks are removed from each token in the string before it is assigned to a variable. Variable B3 or &B3 would have leading or trailing blanks if TITLE contained extra blanks before SUPPORTED, or after X'087D'.

Using Patterns in a Parsing Template

A pattern is a character or string of characters expected to appear within the source variable. Code patterns in the PARSEL2R parsing template using slashes (/) as delimiters. Use patterns within a parsing template to divide the source variable into segments.

When a pattern that you coded in the parsing template occurs in the source variable, the preceding portion of the source variable is treated as a segment. The symbols you defined in the parsing template preceding the pattern are used to parse the tokens in the corresponding segment of the source variable.

Note: If you want to use a slash (/) as a part of a pattern, code two consecutive slashes within the delimiter slashes. PARSEL2R reads this as one slash to be matched in the source variable. Two consecutive slashes by themselves (outside of delimiters) ends the parse.

The following examples show how you can use a parsing template containing patterns and symbols:

For REXX:

```
PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
'PARSEL2R PARSIT A1 A2 A3 //// B1 B2 B3 /:/ C1 C2 C3 /=/ D1 D2 D3'
```

For command list:

```
&PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
PARSEL2R PARSIT A1 A2 A3 //// B1 B2 B3 /:/ C1 C2 C3 /=/ D1 D2 D3
```

The following table shows the resulting values for the REXX and NetView command list language variables:

REXX Variable	NetView Variable	Value
A1	&A1	PROCEDURE
A2	&A2	null
A3	&A3	null
B1	&B1	ACTION
B2	&B2	NOT
B3	&B3	SUPPORTED
C1	&C1	SENSE
C2	&C2	null
C3	&C3	null
D1	&D1	X'087D'
D2	&D2	null
D3	&D3	null

The following examples are of a parsing template containing patterns and symbols:

For REXX:

```
PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
'PARSEL2R PARSIT A1 A2 A3 /:/ A4'
```

For command list:

```
&PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
PARSEL2R PARSIT A1 A2 A3 /:/ A4
```

The following table shows the resulting values for the REXX and NetView command list language variables:

REXX Variable	NetView Variable	Value
A1	&A1	PROCEDURE/ACTION
A2	&A2	NOT
A3	&A3	SUPPORTED
A4	&A4	SENSE = X'087D'

Note: Because the variable just before a pattern (A3 or &A3 in the previous examples) is treated as the last variable in a segment, the leading and trailing blanks are not removed.

To remove the blanks from the A3 or &A3 variable and make the AX or &AX variable null, specify the parsing template as follows:

```
PARSEL2R PARSIT A1 A2 A3 AX /:/ A4
```

You can use variables as part of a pattern (between the slashes). When a variable is part of a pattern, code it outside of the quotes in a REXX command list.

The following example shows three lines from a REXX command list that uses a parsing template with a pattern containing a variable:

```
A0 = SENSE
PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
'PARSEL2R PARSIT A1 A2 A3 /'A0' = / B1'
```

Because A0 is outside quotes, its value is used as part of the pattern. The pattern becomes SENSE =.

When a variable is part of a pattern, code it with an ampersand (&) in command lists written in the NetView command list language. The following example shows the NetView command list language equivalent of the REXX example:

```
&A0 = SENSE
&PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
PARSEL2R PARSIT A1 A2 A3 /&A0 = / B1
```

The following table shows the resulting values for the REXX and NetView command list language variables:

REXX Variable	NetView Variable	Value
A1	&A1	PROCEDURE/ACTION
A2	&A2	NOT
A3	&A3	SUPPORTED:
B1	&B1	X'087D'

You can also use hexadecimal codes in the parsing template pattern. Code a hexadecimal pattern using an X before the slashes. PARSEL2R matches the hexadecimal code in the template with the character in the source variable that corresponds to your system.

The following examples show how to code a parsing template containing a hexadecimal pattern, where X'E7' is an EBCDIC letter X:

For REXX:

```
PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
'PARSEL2R PARSIT A1 A2 X/E7/ A3'
```

For command list:

```
&PARSIT = 'PROCEDURE/ACTION NOT SUPPORTED: SENSE = X'087D''
PARSEL2R PARSIT A1 A2 X/E7/ A3
```

The following table shows the resulting values for the REXX and NetView command list language variables:

REXX Variable	NetView Variable	Value
A1	&A1	PROCEDURE/ACTION
A2	&A2	NOT SUPPORTED: SENSE
A3	&A3	= '087D'

Using patterns and symbols in a parsing template gives you a powerful tool to use when coding your command lists. For example, you can use PARSEL2R to code a source variable in your command list that contains a small table. You can also use the combination of symbols and patterns to search the source variable and assign a token to a variable, based on the matching pattern. This table can contain a string of alternating variables and labels. You can then use PARSEL2R to match a variable with a label to define the flow of logic within your command list.

Using Character Selectors in a Parsing Template

Character selectors in a PARSEL2R are coded as one or more asterisks (*), indicating that you must assign the preceding symbol one or more characters from the source variable. If a character selector does not follow a symbol in the template (coded at the beginning of the template or following a pattern), PARSEL2R skips that number of characters.

For example, if the source variable is:

```
DSI039I MSG FROM CNM01PPT: COMMON GLOBAL VARIABLES HAVE BEEN SET
```

And the parsing template is:

```
/FROM / DOMNAM ***** TASK /:/
```

The following values are assigned:

```
DOMNAM or &DOMNAM = CNM01  
TASK or &TASK = PPT
```

Character selectors are usually used to break up a single token into multiple variables.

Figure 16 and Figure 17 show a parsing template using character selectors to change the value of a REXX message variable or a NetView command list language control variable. &MCSFLAG, or the value returned by the MCSFLAG() function, is set to 00000110 when the command list is entered. The command list statements set bit 6 to zero (0).

```
MCSBITS = MCSFLAG()  
'PARSEL2R MCSBITS BITS1_5 ***** BIT6 * BITS7_8 **'  
BIT6 = 0  
MCSFLAG = BITS1_5 || BIT6  
MCSFLAG = MCSFLAG || BITS7_8
```

Figure 16. REXX PARSEL2R Example Using Character Selectors

```
PARSEL2R MCSFLAG BITS1T05 ***** BIT6 * BITS7T08 **  
&BIT6 = 0  
&MCSFLAG = &CONCAT &BITS1T05 &BIT6  
&MCSFLAG = &CONCAT &MCSFLAG &BITS7T08
```

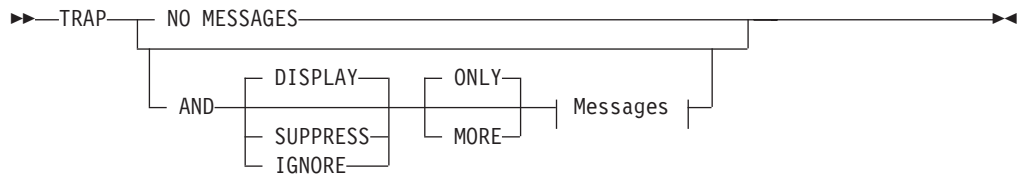
Figure 17. NetView Command List Language PARSEL2R Example Using Character Selectors

After the command list statements in the preceding figures are executed, the value of MCSFLAG or &MCSFLAG is 00000010.

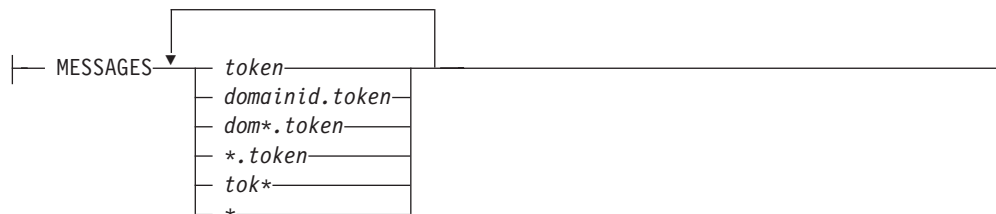
TRAP (REXX)

Syntax

TRAP



Messages:



Purpose of Command

Use the TRAP instruction to define messages that are to be trapped and to specify whether the messages should be displayed to the operator when they are trapped. You can also use the TRAP instruction to remove all messages from the list of messages to be trapped. TRAP can only be used for messages routed to the operator.

For REXX:

The TRAP instruction causes NetView to monitor the operator task for specified messages. If the messages occur, they are trapped and added to the message queue. Trapped messages can then be read using the MSGREAD instruction and can satisfy a WAIT instruction. Refer to the MSGREAD and WAIT (REXX) instructions for information about using these with the TRAP instruction.

If the trapped message satisfies the wait condition, processing of the waiting command procedure resumes. If you do not suppress the message at this point, it continues with the message flow. However, if you suppress the message, NetView marks it for deletion. In this case, automation-table processing does not occur and NetView does not display or log the message.

Note: For information about using TRAP in nested REXX command lists, refer to *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language*. For more information about message flow, refer to the *Tivoli NetView for z/OS Automation Guide*. Refer to the *Tivoli NetView for z/OS Customization Guide* for an example about using the IGNORE option.

For HLL:

The TRAP command lets you specify message trapping criteria for HLL and REXX command procedures. When the TRAP command is issued, all subsequent messages that match the conditions defined by the trapping criteria are added to the message queue (TRAPQ).

When used with the WAIT FOR MESSAGES command and the CNMGETD service routine, the TRAP command allows you to code command procedures to intercept and process, (for example, automate, display, and suppress) certain messages. The message trapping criteria specified in the TRAP command define the set of conditions that satisfy subsequent WAIT FOR MESSAGES commands. TRAP can be issued only from command procedures written in an HLL or REXX.

NetView NNTs, PPTs, OSTs, and autotasks do not process any commands or messages queued to the low priority queue of a task that is running any command list or command processor. Because of this, only messages that are queued to the high or normal priority message queue of a waiting task are checked for matches to satisfy the WAIT condition.

Operand Descriptions

AND	Specifies to make the TRAP command more readable. AND can be used between: <ul style="list-style-type: none">• TRAP and DISPLAY• TRAP and SUPPRESS• TRAP and IGNORE
<u>DISPLAY</u>	Indicates that any message matching a specified <i>token</i> is displayed on the operator's screen when received by NetView. DISPLAY is the default.
IGNORE	Indicates that any message matching a specified <i>token</i> is neither displayed on the operator's screen nor added to the message queue when received by NetView. Because an ignored message is not queued, it does not satisfy any outstanding WAIT and is not available to MSGREAD.
MORE	Indicates that the specified <i>tokens</i> are added to the list of <i>tokens</i> that were specified on a previous TRAP command. Note: Each message in the resulting list retains its individual setting of the SUPPRESS DISPLAY option. This enables some messages in the list to be suppressed while others are displayed.
NO MESSAGES	Indicates that the list of <i>tokens</i> that was specified on all previous TRAP commands is removed. TRAP NO MESSAGES clears the list of messages to trap. No other operands are valid with NO MESSAGES.
<u>ONLY</u>	Indicates that the specified <i>tokens</i> replace the list of <i>tokens</i> specified on all previous TRAP instructions in the invoking REXX command list. ONLY is the default.
SUPPRESS	Indicates that messages matching a specified <i>token</i> are neither displayed on the operator's screen nor logged when received by NetView.

Notes:

1. When using TRAP and SUPPRESS, messages are not logged or automated.
2. If you route a command list to another domain using the ROUTE command, and your command list contains a TRAP AND SUPPRESS instruction, messages resulting from your command list are logged under the NNT in the receiving domain, but are suppressed in your domain.

MESSAGES

Indicates that the trapped items are messages. The 1–8 character *domainid* is the domain ID of the message or messages to be trapped. The 1–10 character *token* identifies the first token of the message or messages to be trapped. Most special characters are valid for *token*. There is no limit on the number of *tokens* that you can specify.

You can use a trailing asterisk (*) in the *domainid* or *token* fields as a wildcard character. You can also use an asterisk (*) as a character in a *token*. If the * is followed by a character, it is considered to be part of the *token*. For example, TRAP MESSAGES A*B* traps on *tokens* beginning with A*B followed by any characters.

Following are examples of how you can specify the messages you want to trap:

<i>domainid.token</i>	The command list traps any message whose domain identifier matches the 1–5 character (for REXX) or 1–8 character (for HLL) <i>domainid</i> and whose first token matches <i>token</i> .
<i>dom*.token</i>	The command list traps any message whose domain identifier matches the partial domain identifier specified by <i>dom*</i> and whose first token matches <i>token</i> . For example, NCCF*.DSI463I means trap a DSI463I message from any domain with an identifier that starts with NCCF, such as NCCFA or NCCFB.
<i>*.token</i>	The command procedure traps any message whose first token matches <i>token</i> . The message can be from any domain.
<i>token</i>	The command list traps any message whose first token matches <i>token</i> . The message can be from any domain.
<i>tok*</i>	The command list traps any message whose first token matches the partial token specified by <i>tok*</i> . For example, DSI* means trap any messages whose first token begins with DSI, such as DSI463I or DSI386I. The message can be from any domain.
*	Specifies that the command list traps all messages routed to the operator.

Note: Using a single asterisk, or in conjunction with SUPPRESS, can cause all messages to be trapped, including those that are not a result of issued commands.

Note: If you specify a token that contains a special character such as a comma, period, asterisk, or most other nonnumeric and nonalphanumeric characters, use the DOMAIN.TOKEN format. Remember, however, that NetView does not accept commas (,) or blank spaces when you specify a token because these characters are reserved as NetView default delimiters.

Return Codes

TRAP sets the value of the return code (RC) to indicate the processing results, as follows:

Return Code Name	Value	Description
CNM_GOOD	0	Everything is OK.
CNM_BAD_INVOCATION	4	TRAP was not issued from an HLL or REXX command procedure.
CNM_BAD_SYNTAX	12	Syntax error. If DSI028I is issued, check <i>domainid.token</i> syntax. If DSI604I is issued, check the format of the TRAP command according to the operand specified.
CNM_BAD_COMMAND	144	TRAP was not issued from a command procedure running under an OST or NNT.
CNM_BAD_MRBLD + X	18000 + X	Nonzero return code X. See values for X below.

Values for X:

Return Code Value	Value for X	Description
18000 + X	8	Request for storage has failed.

Usage Notes

Consider the following when using the TRAP command for REXX:

- Multiline messages, such as multiline write-to-operator (MLWTO), are treated as one message. Therefore, only the message identifier of the first nonblank message in a multiline message is available to the TRAP, and the TRAP can be satisfied only by that message identifier. Use GETMSIZE, GETMTYPE, and GETMLINE to access the other messages in a multiline message. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for an example of the TRAP instruction in a REXX command list. Refer to the GETM commands for more information about multiline messages.
- When using a *token* event, messages not related to the command issued by the TRAP statement can be matched to the event and, depending on the options on the TRAP statement, can be suppressed with SUPPRESS when specifying a domain identifier or token. If the command list is suspended and the SUPPRESS option is in effect on the TRAP statement, any messages the task receives are suppressed before the command list is resumed. This includes any REXX trace output and messages if you are using the REXX TRACE instruction to debug the command list.
- Because NetView NNT, PPT, OST, and autotasks do not process any commands or messages queued to the low priority queue of a task that is running any command (assembler command or HLL, REXX, or NetView command list

language command procedure), only messages that are queued to the high or normal priority queue of a waiting task are checked for matches to satisfy the wait condition.

- Normally, messages queued to tasks with `assign...copy=` processing can satisfy an outstanding trap. However, a message is not sent to the trapping task with `assign...copy=` processing if the message has a message automation table entry specifying `DISPLAY(N)`. The `assign...copy=` processing requires a displayed message, but `DISPLAY(N)` specifies “no display”, which prevents that processing. The message is not passed on and, therefore, cannot satisfy the TRAP condition.

Consider the following when using the TRAP command for HLL:

- The TRAP command can only be issued from a command procedure running under an OST or NNT.
- All operands are order-dependent.
- Each TRAP command without the MORE operand cancels and replaces the previous TRAP command. To add tokens to a previous TRAP command, use the MORE operand.
- Issuing a TRAP command does not clear the queue of messages trapped by the previous TRAP command. To clear the message queue, issue a FLUSHQ.
- If you specify the same number in the TRAP command in both the waiting and nested command procedure, the message satisfies the TRAP in the nested command procedure. If the procedure ends before the message satisfies the TRAP, the message will be queued for the waiting command procedure without ending the TRAP (used with WAIT FOR MESSAGES) or ending the waiting command procedure. The message queue will continue to grow and NetView could run out of storage.
- Immediate messages are not trapped and they are not added to the message queue (TRAPQ).
- TRAP AND SUPPRESS MESSAGES * is not the same as TRAP NO MESSAGES. TRAP AND SUPPRESS MESSAGES * traps all messages but does not display them on the operator’s console.
- Message trapping (TRAP command) takes precedence over NetView automation.
- When trapping TAF messages, the domain ID is the session ID for that TAF session.

Restrictions

The following restrictions apply to the TRAP command:

- Operands must be entered in the order shown in the syntax diagram.
- For REXX, the instruction is enclosed in single quotes to prevent variable substitution.
- The TRAP instruction does not clear the queue of messages trapped by the previous TRAP. To clear the message queue, issue a FLUSHQ instruction. Refer to the FLUSHQ instruction for more information.
- Messages issued by DSIPSS with `TYPE=FLASH` cannot satisfy an outstanding TRAP.

Examples

Example: Using the TRAP Command

The following is an example of a REXX command list named ACTLU. ACTLU issues a VTAM command to activate a node specified by the user. Messages sent as

the result of the activated command are trapped, and the operator is notified of the result of the command list. The comments in the example explain how the command list works:

```

/*****
/* ACTLU COMMAND LIST */
/*
/* FUNCTION : TO ACTIVATE A VTAM NODE. */
/* INPUT : 1 PARAMETER, THE NAME OF THE NODE. */
/*****
SIGNAL ON ERROR /* SIGNAL IF ERROR OCCURS */
IF MSGVAR(1) = '' THEN /* NO FIRST PARAMETER ? */
DO /* THEN ISSUE REQUEST */
SAY 'PLEASE ENTER "GO NODENAME"', /* REQUEST NODENAME FROM USER */
'TO CONTINUE OR "GO STOP" TO', /* OR, ALLOW USER TO STOP */
'STOP'
PARSE PULL NODE /* NODE = NODENAME OR STOP */
END /* THEN ISSUE REQUEST */
ELSE /* FIRST PARAMETER EXISTS */
NODE = MSGVAR(1) /* ASSUME IT IS A NODE NAME */
IF NODE-='STOP' THEN /* IF USER DID NOT CHOOSE STOP */
DO /* PROCESS NODENAME */
'TRAP AND SUPPRESS ONLY MESSAGES IST* ' /* TRAP ALL VTAM MSGS */
'V NET,ACT,ID='NODE /* ISSUE VTAM ACTIVATE FOR NODE */
RCSAVE=RC /* SAVE RETURN CODE IN RCSAVE */
'WAIT 30 SECONDS FOR MESSAGES' /* WAIT FOR 30 SECONDS */
SELECT
WHEN (EVENT()='M') THEN /* OUT OF WAIT - IS THERE A MSG? */
DO /* PROCESS TRAPPED MESSAGE */
'MSGREAD' /* READ IN 1ST MESSAGE */
DO WHILE (RC=0) /* IF RC=0 THEN NO MORE MSGS */
SELECT /* DETERMINE WHICH MESSAGE HIT */
WHEN (MSGID() = 'IST061I') THEN /* NODE NOT FOUND */
SAY '==> LU UNKNOWN', /* INFORM USER */
'TO YOUR VTAM <==='
WHEN (MSGID() = 'IST093I') THEN /* NODE NOW ACTIVE */
SAY '==> TERMINAL 'MSGVAR(1)' NOW', /* INFORM USER */
MSGVAR(2) '<==='
OTHERWISE /* IGNORE THE VTAM MESSAGE */
IF RCSAVE=0 THEN
'WAIT CONTINUE' /* CONTINUE WAITING */
ELSE
DO
SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED ? */
'ACTIVATE COMMAND' /* INFORM USER */
SAY MSGSTR() /* DISPLAY MESSAGE TEXT */
END

```

Figure 18. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 1 of 2)

```

        END                                /* OF SELECT FOR IST061I/IST093I */
        'MSGREAD'                          /* READ IN THE NEXT MESSAGE      */
    END                                    /* DO WHILE RC=0, LOOP BACK      */
END                                        /* PROCESS TRAPPED MESSAGE DO    */
                                        /* OUT OF DO WHILE               */

    WHEN (EVENT()='E'|RCSAVE~=0) THEN /* CHECK FOR ERROR OR           */
                                        /* TIME-OUT EVENTS              */
        SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED?          */
        'ACTIVATE COMMAND' /* INFORM USER                  */
    WHEN (EVENT()='T') THEN /* WAIT TIME-OUT ENCOUNTERED?  */
        SAY 'NO RESPONSE TO', /* INFORM USER                  */
        'ACTLU CLIST FOR 'NODE
    OTHERWISE; /* NO-OP                        */
END /* OF SELECT FOR ERROR/TIME-OUT */
END /* IF NODE~='STOP' PROCESSING */
EXIT 0;
ERROR:
IF RC = 4 THEN
    EXIT
ELSE
    SAY 'ERROR OCCURRED. RETURN CODE IS ' RC
EXIT -1; /* END COMMAND LIST FOR ERROR */

```

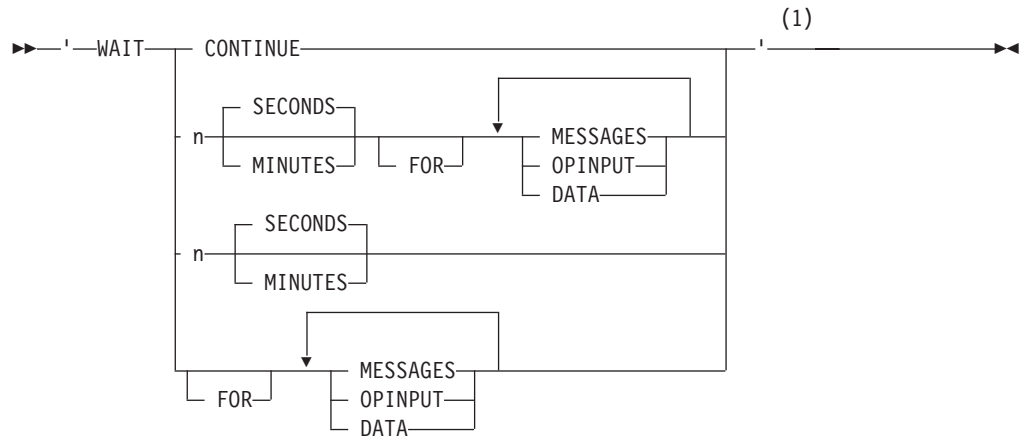
Figure 18. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 2 of 2)

Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more examples of REXX command lists for NetView.

WAIT (REXX)

Syntax

WAIT



Notes:

- 1 OPINPUT and DATA are only valid for HLL.

Purpose of Command

When issued from a command procedure, the WAIT command temporarily suspends processing of that command procedure until a specified event occurs. For an HLL command procedure, the event can be one or more messages, operator input, data, a certain period of time, or any combination of these. The first occurrence of one of these events satisfies the wait and processing is resumed.

When NetView encounters a WAIT instruction in a REXX command list, the letter W is displayed in the upper right corner of the current command facility panel if the screen is refreshed because a message is received or the ENTER key is pressed. This W notifies the operator that the command list has halted its processing and is waiting for a message or group of messages or for a specific period of time. The first of these events that occurs satisfies the WAIT.

WAIT can be used in nested REXX command lists. For more information, refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language*.

Operand Descriptions

CONTINUE Specifies to continue waiting for additional messages, operator input, or data before command procedure processing is resumed.

The options specified on the previous TRAP and WAIT instructions remain in effect for the WAIT CONTINUE instruction. When processing resumes, the next instruction after the WAIT CONTINUE is executed.

For example, if you code the following instructions in a command list and one of the messages specified on the TRAP instruction is received in five seconds, the WAIT instruction is satisfied:

```
'TRAP AND SUPPRESS MESSAGES MSG1, MSG2, MSG3'  
'WAIT 20 SECONDS FOR MESSAGES'  
'MSGREAD'  
.  
.  
.  
'WAIT CONTINUE'  
'MSGREAD'
```

The WAIT CONTINUE instruction waits up to 15 seconds to receive one of the messages specified on the TRAP instruction before resuming command list processing. (Fifteen seconds is the difference between the 20 seconds specified on the WAIT instruction and the five seconds already used to satisfy the WAIT instruction.) When processing resumes, the MSGREAD instruction following WAIT CONTINUE is executed.

DATA (HLL only)

The command procedure waits for data sent by CNMSMSG with message type of DATA.

FOR

Can be used to make the WAIT instruction syntax more readable.

MESSAGES

Specifies the command list waits for a trapped message to be added to the message queue before resuming processing. You define the specific messages for which the command list should wait using the TRAP instruction. If you also specify a time interval, the command list waits for up to that amount of time and then resumes, even if one of the specified messages is not received. If one of the specified messages is already on the message queue, the command list resumes without waiting. Refer to the TRAP instruction for more information.

MINUTES

Specifies the command list waits *n* minutes before resuming processing.

n

Specifies the number of SECONDS or MINUTES that the command procedure waits for receipt of the messages specified on the TRAP command, for receipt of operator input, or for receipt of data before processing is resumed. The command procedure can also wait only for the specified time (no specified events) before processing is resumed.

Valid numbers for *n* are in the range of 0–2678400 seconds and 0–44640 minutes. The equivalent of 2678400 seconds or 44640 minutes is 31 days.

OPINPUT (HLL only)

The command procedure waits for operator input. Data from the QUEUE or GO command can satisfy this wait.

SECONDS

Specifies the command list waits *n* seconds before resuming processing. This is the default.

Return Codes

WAIT sets the value of the return code (RC) to indicate the results of processing.

For REXX:

Return Code	Meaning
-1	There was a DSIGET failure.
0	Processing completed successfully.
4	WAIT is valid only from HLL or REXX command lists.
8	There are too many operands.
12	A syntax error occurred.
144	Not in OST or NNT.
152	WAIT was issued without a previous TRAP.
248	WAIT CONTINUE was issued without a previous valid WAIT.

For HLL:

Return Code Name	Value	Description
CNM_BAD_INVOCATION	4	WAIT was not issued from an HLL or REXX command procedure.
CNM_TOO_MANY	8	Too many operands.
CNM_BAD_SYNTAX	12	Syntax error.
CNM_BAD_COMMAND	144	WAIT was not issued from a command procedure running under an OST or NNT.
CNM_NO_TRAP	152	WAIT was issued but TRAP was not issued. You must issue valid TRAP before issuing WAIT for messages.
CNM_NO_PREV_WAIT	248	No previous WAIT; WAIT CONTINUE is not valid.
CNM_STRG_FAILURE	616	DSIGET failure in a service routine.
CNM_TIME_OUT_WAIT	224	WAIT timed out.
CNM_GO_ON_WAIT	228	GO satisfied wait.
CNM_MSG_ON_WAIT	232	Message received during wait.
CNM_OPINPUT_ON_WAIT	236	OPINPUT received during wait.
CNM_DATA_ON_WAIT	240	DATA received during wait.

Note: Return codes 224–240 are issued when a wait is satisfied. They are generated in place of a 0 return code to inform the operator which event satisfied the wait.

Usage Notes

Consider the following when using the WAIT command:

- The WAIT command can only be issued from an HLL or REXX command procedure running under an OST or NNT.
- A REXX WAIT and a HLL WAIT have some syntactical differences.
- Unlike REXX, an HLL command procedure that issues the WAIT CONTINUE command can continue waiting for operator input and data. As shown in this example, operator input received within 12 seconds satisfies the first wait and

the command procedure continues to wait for operator input for the remaining 18 seconds. The same is true when waiting for data.

```
CNMCOMMAND DATA('WAIT 30 SECONDS FOR OPINPUT');
      /* Wait up to 30 seconds for operator input */
CNMGETDATA
  FUNC(FLUSHQ) QUEUE(OPERQ);
      /* Remove the operator input from the queue */
CNMCOMMAND DATA('WAIT CONTINUE');
      /* Continue waiting for more operator input */
```

- A WAIT CONTINUE command satisfies the previous valid WAIT command. Here the command procedure continues to wait using the wait criteria from the initial WAIT command because the second WAIT command is not valid. If the initial WAIT command is satisfied after 3 seconds, the command procedure continues to wait for messages for the remaining 7 seconds (because the WAIT 20 SECONDS FOR DATAA was not valid).


```
CNMCOMMAND DATA('TRAP MESSAGES *');
      /* Trap all messages */
CNMCOMMAND DATA('WAIT 10 SECONDS FOR MESSAGES');
      /* Wait up to 10 seconds for the first trapped message */
CNMGETDATA FUNC(FLUSHQ) QUEUE(TRAPQ);
      /* Remove messages from the message queue */
CNMCOMMAND DATA('WAIT 20 SECONDS FOR DATAA');
      /* A not valid wait for data (misspelled data) */
CNMGETDATA FUNC(FLUSHQ) QUEUE(DATAQ);
      /* Remove data from the data queue */
CNMCOMMAND DATA('WAIT CONTINUE');
      /* Continue waiting for the next trapped message */
```
- CNMGETD must successfully complete (HLBRC = 0) before a WAIT CONTINUE processes correctly.
- Refer to the NetView online help for a full description of the REXX WAIT command.
- The NetView operator's screen is refreshed whenever a message arrives or the ENTER key is pressed. If the screen is refreshed while an HLL command procedure is in a wait state, the pause and wait status indicators (P and W) are displayed in the upper right corner of the current command facility panel. The W indicator notifies the operator that the command procedure has halted its processing and is waiting for one or more messages, data, or a specified period of time. The P indicator notifies the operator that the command procedure has halted its processing and that one of the events for which it is waiting is operator input. When the wait is satisfied, processing starts again and the indicators are cleared from the screen.
- Flush the operator input queue (FLUSHQ) after a WAIT command is satisfied with operator input. Otherwise, subsequent WAIT FOR OPINPUT commands will be satisfied by data already in the operator input queue. Refer to the NetView online help for more information about FLUSHQ.
- When an HLL command processor is in a wait or pause state, operator commands that are entered can be deferred. The commands are deferred based on the NetView DEFAULTS, OVERRIDE, and CMD commands. Refer to the NetView online help for information about these commands. If a command is not deferred, and it specifies the same message number on a TRAP instruction as the waiting command, the message satisfies the WAIT in the nondeferred command.
- The WAIT command with no operands is not valid.
- The following example illustrates the use of the TRAP and WAIT commands. If the initial WAIT command is satisfied after five seconds, the WAIT CONTINUE

command gets control and the command procedure continues to wait for the remainder of the time specified (15 seconds) in the previous WAIT command.

```
CNMCOMMAND DATA('TRAP MESSAGES MSG1 MSG2 MSG3');
/* Trap messages with tokens MSG1, MSG2, MSG3 */
CNMCOMMAND DATA('WAIT 20 SECONDS FOR MESSAGES');
/* Wait up to 20 seconds for the first trapped message */
CNMGETDATA FUNC(FLUSHQ) QUEUE(TRAPQ);
/* Remove messages from the message queue */
CNMCOMMAND DATA('WAIT CONTINUE');
/* Continue waiting for the next trapped message */
```

Restrictions

The following restrictions apply to the WAIT instruction:

- WAIT cannot be used for PPT tasks.
- You must enter the operands in the order shown in the syntax diagram.
- You must code a time interval, MESSAGES, or both.
- For REXX, the instruction is enclosed in single quotes to prevent variable substitution.

Examples

Example: Using the WAIT Instruction

The following is an example of a REXX command list named ACTLU. ACTLU issues a VTAM command to activate a node specified by the user. Messages sent as the result of the activate command are trapped, and the operator is notified of the result of the command list. The comments in the example explain the command list:

```

/*****
/* ACTLU COMMAND LIST */
/*
/* FUNCTION : TO ACTIVATE A VTAM NODE. */
/* INPUT : 1 PARAMETER, THE NAME OF THE NODE. */
/*****
SIGNAL ON ERROR /* SIGNAL IF ERROR OCCURS */
IF MSGVAR(1) = '' THEN /* NO FIRST PARAMETER ? */
DO /* THEN ISSUE REQUEST */
SAY 'PLEASE ENTER "GO NODENAME"', /* REQUEST NODENAME FROM USER */
'TO CONTINUE OR "GO STOP" TO', /* OR, ALLOW USER TO STOP */
'STOP'
PARSE PULL NODE /* NODE = NODENAME OR STOP */
END /* THEN ISSUE REQUEST */
ELSE /* FIRST PARAMETER EXISTS */
NODE = MSGVAR(1) /* ASSUME IT IS A NODE NAME */
IF NODE='STOP' THEN /* IF USER DID NOT CHOOSE STOP */
DO /* PROCESS NODENAME */
'TRAP AND SUPPRESS ONLY MESSAGES IST* ' /* TRAP ALL VTAM MSGS */
'V NET,ACT,ID='NODE /* ISSUE VTAM ACTIVATE FOR NODE */
RCSAVE=RC /* SAVE RETURN CODE IN RCSAVE */
'WAIT 30 SECONDS FOR MESSAGES' /* WAIT FOR 30 SECONDS */
SELECT
WHEN (EVENT()='M') THEN /* OUT OF WAIT - IS THERE A MSG? */
DO /* PROCESS TRAPPED MESSAGE */
'MSGREAD' /* READ IN 1ST MESSAGE */
DO WHILE (RC=0) /* IF RC=0 THEN NO MORE MSGS */
SELECT /* DETERMINE WHICH MESSAGE HIT */
WHEN (MSGID() = 'IST061I') THEN /* NODE NOT FOUND */
SAY '==> LU UNKNOWN', /* INFORM USER */
'TO YOUR VTAM <==='
WHEN (MSGID() = 'IST093I') THEN /* NODE NOW ACTIVE */
SAY '==> TERMINAL 'MSGVAR(1)' NOW', /* INFORM USER */
MSGVAR(2) '<==='
OTHERWISE /* IGNORE THE VTAM MESSAGE */
IF RCSAVE=0 THEN
'WAIT CONTINUE' /* CONTINUE WAITING */
ELSE
DO
SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED ? */
'ACTIVATE COMMAND' /* INFORM USER */
SAY MSGSTR() /* DISPLAY MESSAGE TEXT */
END

```

Figure 19. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 1 of 2)


```

        END                                /* OF SELECT FOR IST0611/IST093I */
        'MSGREAD'                          /* READ IN THE NEXT MESSAGE      */
    END                                     /* DO WHILE RC=0, LOOP BACK      */
END                                         /* PROCESS TRAPPED MESSAGE DO    */
                                           /* OUT OF DO WHILE                */

    WHEN (EVENT()='E'|RCSAVE≠0) THEN /* CHECK FOR ERROR OR           */
                                           /* TIME-OUT EVENTS               */
        SAY 'ERROR PROCESSING', /* ERROR ENCOUNTERED?          */
        'ACTIVATE COMMAND' /* INFORM USER                  */
    WHEN (EVENT()='T') THEN /* WAIT TIME-OUT ENCOUNTERED?  */
        SAY 'NO RESPONSE TO', /* INFORM USER                  */
        'ACTLU CLIST FOR 'NODE
    OTHERWISE; /* NO-OP                        */
END /* OF SELECT FOR ERROR/TIME-OUT */
END /* IF NODE≠'STOP' PROCESSING */
EXIT 0;
ERROR:
IF RC = 4 THEN
    EXIT
ELSE
    SAY 'ERROR OCCURRED. RETURN CODE IS ' RC
EXIT -1; /* END COMMAND LIST FOR ERROR */

```

Figure 19. Sample Command List Using TRAP, WAIT, MSGREAD, and WAIT CONTINUE (Part 2 of 2)

Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more examples of REXX command lists for NetView.

Example: Checking the Result of a WAIT Instruction

The NetView event that satisfied the WAIT is determined by the value of the REXX EVENT() function. The REXX command list can use the EVENT() function to set a variable and take appropriate action based on the set value. The possible returned values from EVENT() are as follows:

Value Meaning

- M** The message for which the command list is waiting has arrived. The message can be read using the MSGREAD instruction.
- T** The time period for which the command list was waiting has expired, and processing is resumed.
- G** You entered the GO command, and processing is resumed.
- E** You did not code the WAIT or TRAP instructions correctly. For example, you entered the operands in the incorrect order or issued a WAIT for messages instruction without a matching TRAP instruction. The command list resumes processing.

If you do not issue a WAIT instruction in a command list, the value of the EVENT() function is replaced with a value of null.

Example: Processing One Message

The following is an example of a skeleton command list showing how you can use the WAIT instruction to process one message at a time:

```

/* Command List Comments - Include message ID and text */
'TRAP AND SUPPRESS MESSAGES ..... '
/* issue command here */
'WAIT nn SECONDS FOR MESSAGES'
SELECT
    WHEN (EVENT()='M') THEN

```

```

DO
  'MSGREAD'
  /* process message */
END
WHEN (EVENT()='T') THEN
  /* process WAIT time-out */
WHEN (EVENT()='E') THEN
  /* process WAIT error */
WHEN (EVENT()='G') THEN
  /* process GO entered */
OTHERWISE
  /* process anything else */
END
'TRAP NO MESSAGES'
'FLUSHQ MESSAGES'

```

Example: Using NetView Commands with WAIT

The following is an example of a skeleton command list that processes several messages:

```

/* REXX Command List Comments - Include message ID and text */
END_OF_WAIT = 'NO'
'TRAP AND SUPPRESS MESSAGES ..... '
/* issue command here */
IF RC = 0 THEN
DO
  'WAIT nn SECONDS FOR MESSAGES'
  DO WHILE END_OF_WAIT = 'NO'
    'MSGREAD'
    SELECT
      WHEN (EVENT()='M') THEN
        DO
          /* process messages */
          /* call another routine to process the messages */
          CALL PROCESS_MSG
        END
      WHEN (EVENT()='T') THEN
        DO
          /* process WAIT time-out */
          END_OF_WAIT = 'YES'
        END
      WHEN (EVENT()='E') THEN
        DO
          /* process WAIT error */
          END_OF_WAIT = 'YES'
        END
      WHEN (EVENT()='G') THEN
        DO
          /* process GO entered */
          END_OF_WAIT = 'YES'
        END
      OTHERWISE
        /* process anything else */
    END
  END
END
END

```

Figure 20. Using NetView Commands with WAIT (Part 1 of 2)

```

'TRAP NO MESSAGES'
'FLUSHQ MESSAGES'
PROCESS_MSG:
/*****
/* PROCESS_MSG ROUTINE
/* This skeleton routine can be used to identify what types
/* of messages can occur and select an action based on those
/* messages.
/*
/* This skeleton shows how to handle the following conditions:
/* (1) a message that you want to act upon
/* (see MSGID()='xxxxxxx')
/* (2) a message that you want to ignore
/* (see MSGID()='yyyyyyy')
/* (3) a message that you did not plan for in your logic
/* (see OTHERWISE statement)
/*
/* You might have multiples of the above conditions based on what
/* you're doing with the command list and the messages.
/*
*****/
SELECT
  WHEN (MSGID()='xxxxxxx') THEN
    DO
      /* this is a message that you want to act upon and then
      end the TRAP/WAIT for any further messages
      /*
      /* insert message processing here */
      END_OF_WAIT = 'YES'
    END
  WHEN (MSGID()='yyyyyyy') THEN
    DO
      /* this is a message that you want to ignore and have suppressed
      by the TRAP command. You only need to continue
      waiting for the next message...
      /*
      'WAIT CONTINUE'
    END
  OTHERWISE
    DO
      /* process any other message that is on the trapped message
      queue. You can choose one of 2 options:
      (1) continue the wait using a 'WAIT CONTINUE' statement
      (2) end the wait by coding END_OF_WAIT='YES'
      In either case, recommend issuing an error message to the
      task running this command list using the SAY command.
      /*
    END
END

```

Figure 20. Using NetView Commands with WAIT (Part 2 of 2)

When a REXX command list is in a wait or pause state, operator commands that are entered can be delayed. Whether the commands are delayed is based on the NetView DEFAULTS, OVERRIDE, and CMD commands. Refer to the NetView online help for information about these commands.

If a command is not delayed, and it specifies the same message number on a TRAP command as the waiting command list, the message satisfies the WAIT in the nondelayed command.

The GO, STACK, UNSTACK, and RESET commands affect the processing of command lists in a wait state as follows:

GO Ends a WAIT.

STACK

Suspends command list processing and causes any commands that are subsequently queued to be processed. You can enter any command or command list for normal processing while a command list is suspended. The WAIT is not suspended, and events are still matched as they occur. The W does not remain in the upper right corner of the NetView panel. The GO command is rejected until the command list resumes processing.

UNSTACK

Resumes command list processing. The WAIT resumes processing events that were matched while the command list was suspended.

RESET

Ends a command list and all related nested command lists. RESET also invokes HALT when you code SIGNAL ON HALT.

For more information about the GO, STACK, UNSTACK, and RESET commands, refer to the NetView online help.

Sending Messages to the Operator Console

Introduction

You can use the following NetView commands to send messages to, and remove messages from, one or more MVS system consoles:

WTO (write-to-operator)

Send messages to one or more MVS consoles.

WTOR (write-to-operator with reply)

Send messages to the MVS operator console and wait for replies.

DOM

Remove WTO or NetView held messages. The DOM command can also be used to remove NetView held messages from NetView terminals, even when they are not MVS messages.

For REXX:

When a command list is invoked from a NetView automation table, specific system message information is available as returned values when their respective REXX message functions are invoked. This specific system message information is also assigned when a MSGREAD instruction is issued. These assignments occur after every MSGREAD.

The WTO and WTOR commands use the values of the message functions as input when the commands are processed. However, before issuing a WTO or WTOR command, you can override the system values with your own user-assigned values.

You cannot assign a value to a REXX function. You must assign a value to a variable with the same name as the function, but without the parentheses at the end. For example, a MSGREAD instruction reads a message which assigns MCSFLAG() a value of 10000100. To change the value before issuing a WTO command, use the following assignment statement to change the value of MCSFLAG:

```
MCSFLAG = '01000100'
```

Invoking a REXX message function does not set the variable of the same name to a value. These REXX variables are null until a user assignment is made to them. They are not affected by their respective REXX message functions.

If you assign values to any variables that are input to the WTO or WTOR commands, the values you assign are used instead of system values of the REXX functions.

The MSGREAD instruction does not change the user-assigned variable values. If you want to return to the system values that MSGREAD assigns to the functions, use the REXX DROP instruction to drop the variables before issuing the WTO or WTOR command, for example, DROP MCSFLAG. Refer to the appropriate manual in the REXX library for information about the DROP instruction.

For NetView command list language:

When a command list is invoked from a NetView automation table, specific system message information is assigned to NetView command list language message control variables when the command list starts. This specific system message

information is also assigned when an &WAIT control statement is satisfied within the command list. These assignments occur after every &WAIT.

The WTO and WTOR commands use the values of the control variables as input when the commands are processed. However, before issuing a WTO or WTOR command, you can override the system values with your own user-assigned values.

If you assign values to any variables that are input to the WTO or WTOR commands, the values you assign are used instead of system values of the control variables.

With the NetView command list language, you can use assignment statements to directly change the values of the user variables. For example, use the following assignment statement to change the value of &MCSFLAG:

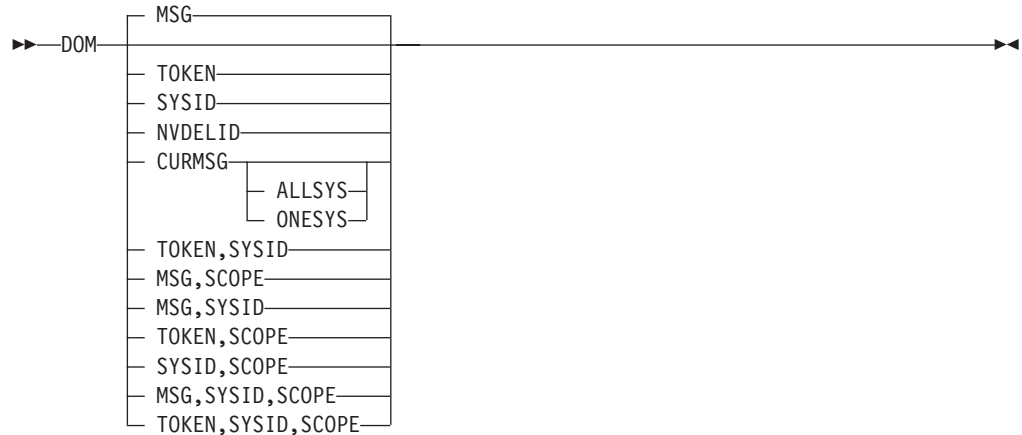
```
&MCSFLAG = 01000100
```

When you have assigned a user-defined value to the control variable, any subsequent references to the control variable will substitute the user-defined value. There is no mechanism to return to the system value within the same NetView command list language command list.

DOM (REXX)

Syntax

DOM



Purpose of Command

The delete operator message (DOM) command is used in a command list to remove a WTO or WTOR message from one or more MVS consoles, including MVS consoles that are in use by NetView. You can delete NetView held messages from NetView terminals using the NVDELID or CURMSG options. This is similar to the DOM function of MVS, and is useful for messages that were held by automation, and not issued from MVS as WTOs or WTORs.

Use the DOM command to remove action messages after verifying that the action was completed. The DOM command uses the command options MSG, TOKEN, SYSID, SCOPE, NVDELID, or CURMSG to determine which messages should be removed. You can set a value for these variables as input to the DOM command, as shown for SMSGID in the following examples:

For REXX:

```
SMSGID = value
```

For command list:

```
&SMSGID = value
```

Parentheses are not used in either the NetView command list language or REXX versions.

If a value is not set for the variables that are used as input to the DOM command, the current system values are used by default.

For REXX, the current system values reside in the SMSGID(), MSGTOKEN(), MSGCSYID(), and NVDELID() functions.

For NetView command list language, the current system values reside in the &SMSGID, &MSGTOKEN, &MSGCSYID, and &NVDELID control variables. If your command list was invoked by automation or issued a WAIT that was

satisfied by a message, then the current system value of REXX message functions and command list control variables such as SMSGID are derived from the driving message.

Operand Descriptions

CURMSG

The NetView deletion ID in the current message identifies which message is to be deleted. This option is useful for removing held messages that are not WTOs or WTORs. It is also useful in a REXX procedure when you want to delete the message that is currently being automated. For example, the message whose values are set by MSGREAD.

If no parameter is specified, the message will be deleted, within NetView, without issuing an MVS system DOM signal. Optionally, you can specify one of the following parameters:

ALLSYS

Generates an MVS system DOM signal to all systems.

ONESYS

Generates an MVS system DOM signal to a single system.

DOM

When specified without parameters, DOM provides compatible function to DOM MSG.

MSG

The &SMSGID value determines the message to be deleted.

The &SCOPE value defaults to SYSTEMS.

MSG,SCOPE

The &SMSGID value determines the message to be deleted.

The &SCOPE value determines whether messages are deleted from only the issuing processor or all communicating MVS systems (JES3 only).

MSG,SYSID

The &SMSGID and &MSGCSYID values determine the message to be deleted.

The &SCOPE value defaults to SYSTEMS.

MSG,SYSID,SCOPE

The &SMSGID and &MSGCSYID values determine the message to be deleted.

The &SCOPE value determines whether messages are deleted from only the issuing processor or all communicating MVS systems (JES3 only).

NVDELID

The NetView deletion ID for the NVDELID REXX variable or &NVDELID NetView command list variable identifies which messages are deleted. This option provides a NetView console equivalent to the MVS DOM function. It is useful for removing held messages that are not WTOs or WTORs. The message is removed from all NetView tasks receiving the message. Use this option to delete messages for which the NVDELID value was saved, for example, in a global variable. This option does not use the MVS DOM function within NetView.

SYSID

All messages issued by system &MSGCSYID are deleted.

SYSID,SCOPE

All messages issued by system &MSGCSYID are deleted.

The &SCOPE value determines whether messages are deleted from only the issuing processor or from all communicating MVS systems (JES3 only).

TOKEN

The &MSGTOKEN value determines the message to be deleted.

The &SCOPE value defaults to SYSTEMS.

TOKEN,SCOPE

The &MSGTOKEN value determines the message to be deleted.

The &SCOPE value determines whether messages are deleted from only the issuing processor or all communicating MVS systems (JES3 only).

TOKEN,SYSID

The &MSGTOKEN and &MSGCSYID values determine the message to be deleted.

The &SCOPE value defaults to SYSTEMS.

TOKEN,SYSID,SCOPE

The &MSGTOKEN and &MSGCSYID values determine the message to be deleted.

The &SCOPE value determines whether messages are deleted from only the issuing processor or all communicating MVS systems (JES3 only).

Command List Variables

Table 7 lists the DOM command variables:

Table 7. DOM Command List Variable Reference Table

DOM Macro Keyword	Function Variable Name	Usage	Applicable DOM command parameters
COUNT	Not required	Not used	None
DOMCBLK	Not required	Not used	None
TOKEN	MSGTOKEN() &MSGTOKEN	1- to 10-character decimal value, with a maximum value of 4294967295.	DOM TOKEN ¹ DOM TOKEN,SYSID DOM TOKEN,SCOPE ² DOM TOKEN,SYSID,SCOPE ²
SCOPE ²	(none)	Send DOM to SYSTEM or SYSTEMS	DOM MSG,SCOPE DOM TOKEN,SCOPE DOM SYSID,SCOPE DOM MSG,SYSID,SCOPE DOM TOKEN,SYSID,SCOPE
MSG MSGLIST ³	SMSGID() &SMSGID	Value used on DOM of a message	DOM DOM MSG DOM MSG,SYSID DOM MSG,SCOPE ² DOM MSG,SYSID,SCOPE ²
REPLY=YES	Not required	Not used	None
SYSID	MSGCSYID() &MSGCSYID	MVS system identifier	DOM SYSID DOM MSG,SYSID DOM TOKEN,SYSID DOM SYSID,SCOPE ² DOM MSG,SYSID,SCOPE DOM TOKEN,SYSID,SCOPE
(none)	NVDELID() &NVDELID	24-character values saved from a message (NetView deletion ID)	DOM NVDELID

Table 7. DOM Command List Variable Reference Table (continued)

DOM Macro Keyword	Function Variable Name	Usage	Applicable DOM command parameters
(none)	(none)	NetView deletion ID for current message	DOM NVDELID

Table Notes:

1. DOM by TOKEN with a TOKEN value of zero is not recommended. See "Usage Notes" for more information.
2. JES3 only
3. Not required

Usage Notes

Consider the following when using the DOM command:

- There is neither a SCOPE() function nor an &SCOPE control variable.
- After you issue a WTO or WTOR command, the SMSGID value, corresponding to the message issued, is available in the SMSGID or &SMSGID command list variables.

Consider the following when using DOM TOKEN:

- You can use TOKEN to group a series of WTOs and issue a DOM TOKEN to delete the group of messages. However, use caution when selecting the token value for the WTOs that are to be deleted with DOM TOKEN. If possible, plan the use of TOKENs so that the TOKEN used by your command procedure will not intersect with the TOKEN used by other command procedures in your system.
- WTOs issued without specifying TOKEN will default to a value of zero (0). A DOM TOKEN with MSGTOKEN equal to zero can delete many messages that did not originate from your command processor and is not recommended.
- In a sysplex environment, a DOM with a TOKEN issued by one system in the complex will be forwarded to all of the communicating MVS systems. If there is not a corresponding message to be deleted on the receiving systems (other than the system that issued the DOM), MVS might apply the DOM to a subsequent message that arrives with a valid TOKEN and ASID.

To ensure that token values are as close to unique as possible, they should be planned or generated. The example REXX EXEC shown in Figure 21 on page 337 can be used to generate a TOKEN value from a store clock value. If you use the NetView command list language, you can call a REXX EXEC similar to this example from your command list to return the generated token value in a global variable.

```

/*****
/* Generate a 1-10 digit decimal number from the store clock value. */
/*
/* This method can be used in REXX to generate suitable values */
/* for TOKENs used in the WTO and DOM command processors. */
/*
NUMERIC DIGITS 20          /* STCKGMT converts to 20 digits */
STCKVAL = STCKGMT()        /* Obtain the store clock value */
NUMSTCK = C2D(STCKVAL)     /* Convert store clock to decimal */
TOKVAL = NUMSTCK // 4294967295 /* Reduce to 4 byte number */
SAY 'Example of value which could be used for token: ' TOKVAL
EXIT
*****/

```

Figure 21. REXX EXEC to Generate a Token Value from a Store Clock Value

- DOM TOKEN will not delete operator messages from the MVS subsystem allocatable consoles obtained by NetView if the subsystem interface is being used for MVS messages.
DOM TOKEN will delete operator messages from the EMCS consoles obtained by NetView.

Return Codes

The return code (RC or &RETCODE) indicates the processing results as follows:

Return Code	Meaning
4	There is a syntax error.
8	Storage is not available to continue processing.
100	SMSGID length is not valid.
104	SMSGID value is not valid.
108	Not invoked from a command procedure.
112	MSGTOKEN length is not valid.
116	MSGTOKEN value is not valid.
120	MSGCSYID length is not valid.
124	MSGCSYID value is not valid.
128	SCOPE length (JES3 only) is not valid.
132	SCOPE value (JES3 only) is not valid.
136	Specified DOM option is not valid.
140	NVDELID length is not valid.
144	NVDELID value is not valid.
148	NetView DOM routing failed.
152	There is no current message from which to copy NVDELID.

Examples

Example: Issuing DOM Commands

The following example illustrates how to issue DOM commands:

```

/*****
/* WTOEXAM COMMAND LIST */
/* ----- */
/*
/* FUNCTION: THIS NETVIEW REXX COMMAND LIST DEMONSTRATES THE ABILITY */
/* TO ISSUE WTO, WTO, AND DOM COMMANDS. IT SHOWS HOW */
/* TO CONTROL VARIOUS CONTROL VARIABLES THAT */
/* AFFECT ATTRIBUTES OF THE WTO. */
/*
/* STEPS: */
/* WTO A SERIES OF MESSAGES TO THE SYSTEM CONSOLE THAT HAVE */
*****/

```

```

/*      DIFFERENT ATTRIBUTES, AS DEFINED IN MVS/XA SYSTEM      */
/*      MACROS AND FACILITIES VOLUME 2 UNDER THE              */
/*      WTO MACRO.                                           */
/*      SAVE THE MESSAGE IDS FOR THE WTO'ED MESSAGES.        */
/*      DELETE THE MESSAGES FROM THE CONSOLE.                */
/*      ISSUE AND RETRIEVE THE RESPONSE TO A WTOR COMMAND.    */
/*      */
/*****
/*****
/* ISSUE SOME WTOS                                           */
/*****
/*****
ROUTCDE = '(1)'                                           /* MASTER CONSOLE ACTION */
DESC = '(1)'                                             /* SYSTEM FAILURE        */
'WTO THIS MESSAGE IS URGENT'                             /* ISSUE THE MESSAGE     */
MSGID_FOR_FIRST_MESSAGE=MSGID                           /* SAVE THE MESSAGE ID   */
ROUTCDE = '(2)'                                           /* MASTER CONSOLE INFO   */
DESC = '(3)'                                             /* EVENTUAL ACTION REQUIRED */
'WTO THIS MESSAGE CAN BE HANDLED LATER'                 /* ISSUE THE MESSAGE     */
MSGID_FOR_SECOND_MESSAGE=MSGID                           /* SAVE THE MESSAGE ID   */
/*****
/* ISSUE A MLWTO                                           */
/*****
ROUTCDE = '(2)'                                           /* MASTER CONSOLE INFO   */
DESC = '(3)'                                             /* EVENTUAL ACTION REQUIRED */
LINETYPE='C'                                             /* THIS IS A CONTROL LINE */
'WTO THE FIRST LINE OF A MLWTO'                         /* ISSUE THE FIRST LINE  */
LINETYPE='D'                                             /* THIS IS A DATA LINE  */
'WTO THE SECOND LINE OF A MLWTO'                       /* ISSUE THE SECOND LINE */
LINETYPE='DE'                                           /* THIS IS THE LAST DATA LINE */
/*
'WTO THE THIRD LINE OF A MLWTO'                         /* ISSUE THE THIRD LINE  */
MSGID_FOR_THIRD_MESSAGE=MSGID                           /* SAVE THE MESSAGE ID   */
/*****
/* DEMONSTRATE THE ABILITY TO DELETE OPERATOR MESSAGES (DOM) */
/*****
'WAIT 5 SECONDS'                                         /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_FIRST_MESSAGE                           /* RESTORE THE MESSAGE ID */
'DOM'                                                    /* DELETE THE MESSAGE     */
'WAIT 5 SECONDS'                                         /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_SECOND_MESSAGE                         /* RESTORE THE MESSAGE ID */
'DOM'                                                    /* DELETE THE MESSAGE     */
'WAIT 5 SECONDS'                                         /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_THIRD_MESSAGE                          /* RESTORE THE MESSAGE ID */
'DOM'                                                    /* DELETE THE MESSAGE     */
/*****
/* ASK THE OPERATOR FOR INFORMATION VIA A WTOR              */
/*****
'WTOR ENTER SOME DATA:'                               /* ISSUE THE WTOR        */
SAY 'THE DATA ENTERED AT THE CONSOLE WAS: "WTOREPLY"'
EXIT

```

WTO (REXX)

Syntax

WTO (MVS only)

►—WTO *message_text*—◄

Purpose of Command

The write-to-operator (WTO) command sends a message to one or more MVS consoles. In an automation task command list, use the WTO command as an alternative to automatic processing. For example, instances that require operator intervention, such as adding paper to a printer or choosing among several processing alternatives.

Operand Descriptions

message_text

Specifies the message to send to one or more MVS consoles. This can include the master console and MVS consoles in use by NetView. You can send a character string or set a variable name to the value of the message you want to send.

For REXX command lists, enclose character strings within quotes along with the command. If you use a variable, insert a blank after the command, close the quotes, and enter the name of the variable outside the quotes. For example, if the message is contained in a variable named MSG1, code the following:

```
'WTO 'MSG1
```

Command List Variables

This section describes how to override system values with user-defined values for the control variables and REXX functions that are used as input to WTO. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more information about the NetView command list language control variables and REXX functions. Refer to the MVS library for more information about the MVS WTO macro.

In Table 8, the WTO command uses the values of the REXX functions or NetView command list language control variables (respectively) as input. To override the REXX function value, set a REXX variable with the same name as the function, but without using the parentheses.

Table 8. WTO Command List Variable Reference Table

WTO Macro Keyword	Function Variable Name	Usage
AREAID	AREAID() &AREAID	One-letter console panel area.
CART	CART() &CART	8-byte command and response token (CART)
DESC	DESC() &DESC	Descriptor codes
KEY	KEY() &KEY	8-byte key

Table 8. WTO Command List Variable Reference Table (continued)

WTO Macro Keyword	Function Variable Name	Usage
LINETYPE	LINETYPE() &LINETYPE	MLWTO line type
MCSFLAG	MCSFLAG() &MCSFLAG	Multiple console support flags
TOKEN	MSGTOKEN() &MSGTOKEN	4-byte decimal token
MSGTYP	MSGTYP() &MSGTYP	SESS, JOBNAMEs, STATUS bits
ROUTCDE	ROUTCDE() &ROUTCDE	Routing codes
CONNECT	SMSGID() &SMSGID	Value used to connect a multiline message and a DOM of an action message. In NetView, specify the correct linetype to indicate that you want to build an MLWTO.
CONSID CONSNAME	SYSCONID() &SYSCONID	Console number in decimal, or console name

AREAID, &AREAID

Is a 1-letter (A–Z) identifier for the area on the console panel that displays the message.

For REXX:

AREAID

▶▶—AREAID = 'letter'—▶▶

For command list:

&AREAID

▶▶—&AREAID = letter—▶▶

Note: MVS respects the AREAID specification only for MLWTOs on multiple console support consoles.

CART, &CART

Is an 8-character command and response token (CART) represented by 8 bytes of hexadecimal or EBCDIC characters in the form of:

For REXX:

CART

▶▶—CART = $\left[\begin{array}{l} \text{'8-byte-hex-value'} \\ \text{'8-byte-character-value'} \end{array} \right]$ —▶▶

For command list:

&CART

►► &CART = $\left[\begin{array}{l} \text{X'8-byte-hex-value'} \\ \text{8-byte-character-value} \end{array} \right]$

The following two examples use the hexadecimal notation:

CART = 'C1C2C3C4F0F1F2F3'X (REXX)

&CART = X'C1C2C3C4F0F1F2F3' (Command list)

The following example uses EBCDIC character notation:

CART = ABCD1234

This variable can be used as input to the WTO command on systems running MVS/SP™ 4.1 or a later release.

Note: All 8 bytes must be specified.

DESC, &DESC

The descriptor codes for the WTO, specified as a list of numbers in parentheses:

For REXX:

DESC

►► DESC = $\left(\left[\begin{array}{l} \text{number}_1\text{-}16 \\ \text{number}_1\text{-}16 \end{array} \right] \right)$

For command list:

&DESC

►► &DESC = $\left(\left[\begin{array}{l} \text{number}_1\text{-}16 \\ \text{number}_1\text{-}16 \end{array} \right] \right)$

You can also specify specific numbers and a range of numbers, for example:

&DESC = '(3,5-7)' (command list)

Notes:

1. For compatibility with previous releases of NetView, DESC can also be specified as a string of ones and zeros. For example, to turn on descriptor codes 3, 5, and 7:

&DESC = '001010100000' (command list)

When specifying a string of ones and zeros, omit the opening and closing parentheses.

2. In general, do not use descriptor codes that are reserved in MVS. Descriptor code 13 is used by NetView to flag messages that have already been automated on the MVS system.

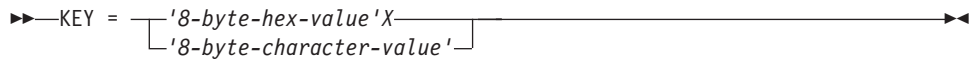
Setting descriptor code 13 will cause your messages to be ineligible for automation or display by NetView on your system. In a sysplex, however, these messages could still be eligible for automation and display on the other systems.

KEY, &KEY

Specifies a retrieval key represented by 8 bytes of hexadecimal or EBCDIC characters in the form of:

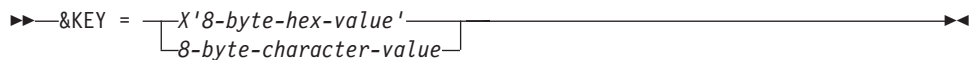
For REXX:

KEY



For command list:

&KEY



The following two examples use the hexadecimal notation:

`KEY = X'C1C2C3C4F0F1F2F3'` (REXX)

`&KEY = X'C1C2C3C4F0F1F2F3'` (command list)

The following example uses character notation:

`KEY = ABCD1234`

Notes:

1. All 8 bytes must be specified.
2. To use the MVS D R, KEY command, specify only EBCDIC characters for KEY.

LINETYPE, &LINETYPE

Provides the multiline write-to-operator (MLWTO) line type in the form of:

For REXX:

LINETYPE



For command list:

&LINETYPE



Notes:

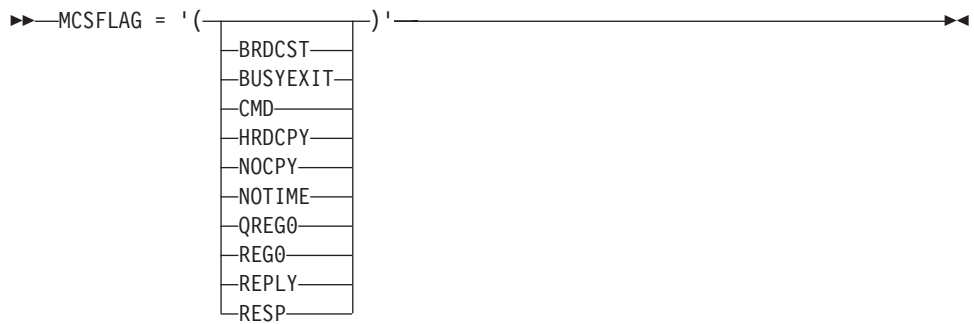
1. When you use the WTO command to issue multiline messages, assign the REXX variable LINETYPE or the NetView command list language variable &LINETYPE and issue a WTO command for each line.
2. Issue the WTO commands for all lines before the command list completes execution. NetView saves the lines until an end line is issued, or purges an unfinished multiline WTO when the command list ends. This helps prevent console hangs due to incomplete multiline WTOs.

MCSFLAG, &MCSFLAG

The MCSFLAGs to be associated with the WTO, specified as a list of keywords in parentheses:

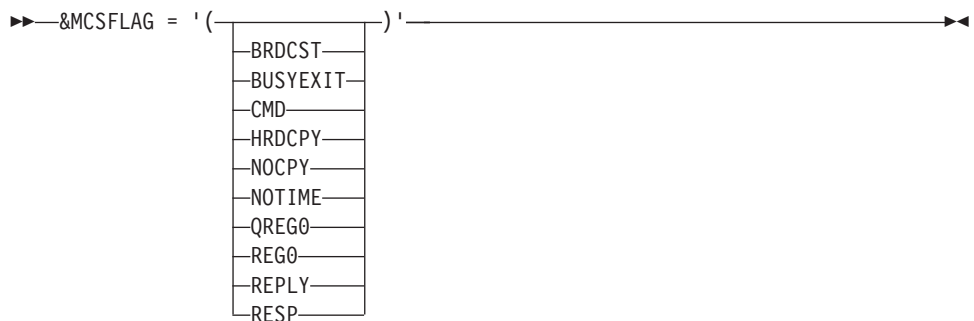
For REXX:

MCSFLAG



For command list:

&MCSFLAG



Notes:

1. More than one keyword can be specified. The delimiter is a comma or blank.

2. CMD and BUSYEXIT cannot be retrieved when messages are received by NetView.
3. For compatibility with previous releases of NetView, MCSFLAG can also be specified as a string of ones and zeros.

For example, to set MCSFLAG to QREG0 and NOCPY:

```
MCSFLAG = '01000001' (REXX)
```

```
&MCSFLAG = '01000001' (command list)
```

When specifying a string of ones and zeros, omit the opening and closing parentheses.

MSGTOKEN, &MSGTOKEN

Is a numeric token associated with the WTO in the form of:

For REXX:

MSGTOKEN

```
▶▶—MSGTOKEN = number_1-4294967295—▶▶
```

For Command List:

&MSGTOKEN

```
▶▶—&MSGTOKEN = number_1-4294967295—▶▶
```

This token can be used to subsequently delete the operator message using the DOM command with the TOKEN option. See "Usage Notes" on page 336 for more information.

MSGTYP, &MSGTYP

The system message type flags for the WTO, specified as a list of keywords in parentheses:

For REXX:

MSGTYP

```
▶▶—MSGTYP = '(JOBNAMES SESS STATUS)'—▶▶
```

For Command List:

&MSGTYP

```
▶▶—&MSGTYP = '(JOBNAMES SESS STATUS)'—▶▶
```

Notes:

1. More than one keyword can be specified. The delimiter is a comma or blank.
2. For compatibility with previous releases of NetView, MSGTYP can also be specified as a string of ones and zeros.

For example:

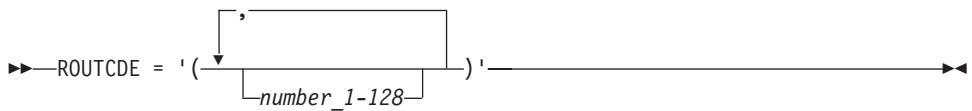
&MSGTYP = '010' (Command List)

ROUTCDE, &ROUTCDE

The system routing codes, specified as a list of numbers in parentheses:

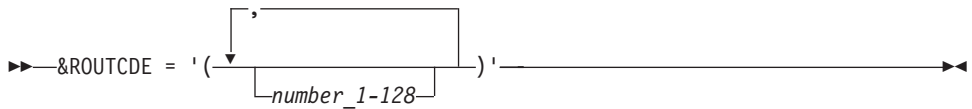
For REXX:

ROUTCDE



For Command List:

&ROUTCDE



You can also specify specific numbers and a range of numbers, for example:

&ROUTCDE = '(2,4-7)' (Command List)

For compatibility with previous releases of NetView, ROUTCDE can also be specified as a string of ones and zeros. For example, to turn on codes 3, 5, and 7:

&ROUTCDE = '001010100000' (Command List)

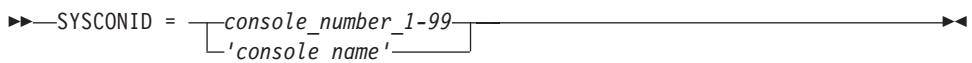
When specifying a string of ones and zeros, omit the opening and closing parentheses.

SYSCONID, &SYSCONID

The destination console identifier for the WTO. This can be either a name or a decimal number in the range of 1-99, as shown:

For REXX:

SYSCONID



For Command List:

&SYSCONID

►►&SYSCONID = $\left\{ \begin{array}{l} \text{console_number_1-99} \\ \text{console_name} \end{array} \right.$ ◀◀

Notes:

1. Set on MCSFLAG bits 1 (REG0) or 2 (QREG0), or both, if you want SYSCONID to be used or checked.
2. The console name you specify must be active. If it is not, a “nonvalid SYSCONID” return code is received.
3. MVS reserves some console names. Currently, the following names are reserved:
 - HC
 - INSTREAM
 - INTERNAL
 - UNKNOWN

These reserved names, and the corresponding console IDs, cannot be used as SYSCONID values for input to the WTO command.

4. Console names can be used only if your MVS system is release 4.1 or later and you are using NetView for MVS systems.

Usage Notes

Consider the following when using the WTO command:

- The values of these variables determine how the WTO command is processed. The variables provide the same input as the keywords on an MVS WTO macro. If a command list does not set the variables before issuing the WTO command, their values default to the current system values.
- For REXX command lists, the current system values are contained in the functions that correspond to the variable names. For example, the current system value for the REXX SYSCONID variable is contained in the SYSCONID() function.
- The WTO command returns values in the following REXX variables:
 - RC
 - SMSGID
- The WTO command returns values in the following NetView command list language control variables:
 - RC, &RETCODE
 - SMSGID, &SMSGID

Return Codes

The return code (RC or &RETCODE) indicates the processing results as follows:

Return Code	Meaning
4	WTO was sent with truncated text.
8	Storage is not available to continue processing.
16	Internal processing failed.
100	AREAID is not valid.
112	SYSCONID length was not valid, or SYSCONID was not specified but is required because MCSFLAG bit 1 (REG0) or bit 2 (QREG0), or both, were set on.

116	SYSCONID value is not valid.
120	Internal decimal conversion failed.
124	The command list dictionary update failed.
132	The command is not allowed under PPT.
136	LINETYPE is not valid.
138	Not invoked from a command procedure.
140	KEY length is not valid.
148	MSGTOKEN length is not valid.
152	MSGTOKEN value is not valid.
156	CART length is not valid.
164	ROUTCDE value is not valid.
168	MSGTYP value is not valid.
172	MCSFLAG value is not valid.
176	DESC value is not valid.

A return code with a value greater than 200 indicates that the return code was passed from the MVS WTO macro. Subtract 200 from the value of the return code. The new value corresponds to the return code that was passed from the MVS WTO macro. For example, if you receive a return code of 208, refer to the MVS library for the meaning of return code 8 from the MVS WTO macro.

Examples

Example: Issuing WTO Commands

The following example displays how to issue WTO commands:

```

/*****/
/*  WTOEXAM  COMMAND LIST                               */
/*  -----  -----  -----                           */
/*                                                    */
/*  FUNCTION: THIS NETVIEW REXX COMMAND LIST DEMONSTRATES THE ABILITY */
/*            TO ISSUE WTO, WTOR, AND DOM COMMANDS.  IT SHOWS HOW    */
/*            TO CONTROL VARIOUS CONTROL VARIABLES THAT              */
/*            AFFECT ATTRIBUTES OF THE WTO.                          */
/*                                                    */
/*  STEPS:                                                                 */
/*  WTO A SERIES OF MESSAGES TO THE SYSTEM CONSOLE THAT HAVE        */
/*  DIFFERENT ATTRIBUTES, AS DEFINED IN MVS/XA SYSTEM                */
/*  MACROS AND FACILITIES VOLUME 2 UNDER THE                        */
/*  WTO MACRO.                                                       */
/*  SAVE THE MESSAGE IDS FOR THE WTO'ED MESSAGES.                   */
/*  DELETE THE MESSAGES FROM THE CONSOLE.                            */
/*  ISSUE AND RETRIEVE THE RESPONSE TO A WTOR COMMAND.              */
/*                                                    */
/*****/
/*****/
/*  ISSUE SOME WTOS                                                 */
/*****/
ROUTCDE = '(1)' /* MASTER CONSOLE ACTION */
DESC = '(1)' /* SYSTEM FAILURE */
'WTO THIS MESSAGE IS URGENT' /* ISSUE THE MESSAGE */
MSGID_FOR_FIRST_MESSAGE=MSGID /* SAVE THE MESSAGE ID */
ROUTCDE = '(2)' /* MASTER CONSOLE INFO */
DESC = '(3)' /* EVENTUAL ACTION REQUIRED */
'WTO THIS MESSAGE CAN BE HANDLED LATER' /* ISSUE THE MESSAGE */
MSGID_FOR_SECOND_MESSAGE=MSGID /* SAVE THE MESSAGE ID */
/*****/
/*  ISSUE A MLWTO                                                 */
/*****/
ROUTCDE = '(2)' /* MASTER CONSOLE INFO */
DESC = '(3)' /* EVENTUAL ACTION REQUIRED */
LINETYPE='C' /* THIS IS A CONTROL LINE */
'WTO THE FIRST LINE OF A MLWTO' /* ISSUE THE FIRST LINE */

```

```

LINETYPE='D'                                /* THIS IS A DATA LINE */
'WTO THE SECOND LINE OF A MLWTO'           /* ISSUE THE SECOND LINE */
LINETYPE='DE'                               /* THIS IS THE LAST DATA LINE
*/
'WTO THE THIRD LINE OF A MLWTO'           /* ISSUE THE THIRD LINE */
MSGID_FOR_THIRD_MESSAGE=MSGID             /* SAVE THE MESSAGE ID */
/*****/
/* DEMONSTRATE THE ABILITY TO DELETE OPERATOR MESSAGES (DOM) */
/*****/
'WAIT 5 SECONDS'                           /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_FIRST_MESSAGE             /* RESTORE THE MESSAGE ID */
'DOM'                                       /* DELETE THE MESSAGE */
'WAIT 5 SECONDS'                           /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_SECOND_MESSAGE           /* RESTORE THE MESSAGE ID */
'DOM'                                       /* DELETE THE MESSAGE */
'WAIT 5 SECONDS'                           /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_THIRD_MESSAGE            /* RESTORE THE MESSAGE ID */
'DOM'                                       /* DELETE THE MESSAGE */
/*****/
/* ASK THE OPERATOR FOR INFORMATION VIA A WTOR */
/*****/
'WTOR ENTER SOME DATA:'                   /* ISSUE THE WTOR */
SAY 'THE DATA ENTERED AT THE CONSOLE WAS: ''WTOREPLY'' '
EXIT

```

WTOR (REXX)

Syntax

▶—WTOR *message_text*—▶

Purpose of Command

The write-to-operator with reply (WTOR) command sends a message to one or more MVS consoles and requests a reply. Command lists that use the WTOR command do not complete until an operator replies. Therefore, use WTOR with care. Because command list processing is suspended while waiting for the reply from the WTOR, other command lists, including command lists that are run with a low priority, can be scheduled and executed before the original command list completes.

If the command list is written in REXX, the operator reply is stored in the WTOREPLY variable, and the ID of the system console that replied is stored in the SYSCONID variable.

If the command list is written in the NetView command list language, the operator reply is stored in the control variable &WTOREPLY, and the ID of the system console that replied is stored in &SYSCONID.

Operand Descriptions

message_text

The message to send to one or more MVS consoles. This can include the master console as well as MVS consoles in use by NetView. You can send a character string or set a variable name to the value of the message you want to send.

For REXX command lists, enclose character strings within quotes along with the command. If you use a variable, put a blank after the command, close the quotes, and put the name of the variable outside the quotes. For example, if the message is contained in a variable named MSG1, code the following:

```
'WTOR 'MSG1
```

Command List Variables

This section describes how to override system values with user-defined values for the control variables and REXX functions that are used as input to WTOR. Refer to the *Tivoli NetView for z/OS Customization: Using REXX and the NetView Command List Language* for more information about the NetView command list language control variables and REXX functions. Refer to the MVS library for more information about the MVS WTOR macro.

In Table 9, the WTOR command uses the values of the REXX functions or NetView command list language control variables (respectively) as input. To override a REXX function, set a REXX variable with the same name as the function, but without the parentheses.

Table 9. WTOR Command List Variable Reference Table

WTOR Macro Keyword	Function Variable Name	Usage
CART	CART() &CART	8-byte command and response token (CART)
DESC	DESC() &DESC	Descriptor codes
KEY	KEY() &KEY	8-byte key
MCSFLAG	MCSFLAG() &MCSFLAG	Multiple console support flags
TOKEN	MSGTOKEN() &MSGTOKEN	1- to 10-character decimal value, with a maximum value of 4294967295.
MSGTYP	MSGTYP() &MSGTYP	SESS, JOBNAMEs, STATUS bits
ROUTCDE	ROUTCDE() &ROUTCDE	Routing codes
CONSID CONSNAME	SYSCONID() &SYSCONID	Console number in decimal, or console name
WQEBLK	Not required	N/A
none	WTOREPLY &WTOREPLY	Variable that returns an operator's reply to a WTOR

CART, &CART

Is an 8-character command and response token (CART) represented by 8 bytes of hexadecimal or EBCDIC characters in the form of:

For REXX:

CART

►► CART = $\left[\begin{array}{l} \text{'8-byte-hex-value'} \\ \text{'8-byte-character-value'} \end{array} \right]$ ◀◀

For Command List:

&CART

►► &CART = $\left[\begin{array}{l} \text{'X'8-byte-hex-value'} \\ \text{8-byte-character-value} \end{array} \right]$ ◀◀

The following two examples use the hexadecimal notation:

CART = 'C1C2C3C4F0F1F2F3' (REXX)

&CART = 'C1C2C3C4F0F1F2F3' (Command List)

The following example uses EBCDIC character notation:

CART = ABCD1234

This variable can be used as input to the WTOR command on systems running MVS/SP 4.1 or a later release.

Note: All 8 bytes must be specified.

DESC, &DESC

The descriptor codes for the WTOR, specified as a list of numbers in parentheses:

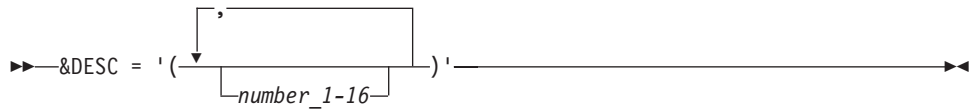
For REXX:

DESC



For Command List:

&DESC



You can also specify specific numbers and a range of numbers, for example:

```
&DESC = '(3,5-7)'
```

Notes:

1. For compatibility with previous releases of NetView, **DESC** can also be specified as a string of ones and zeros. For example, to turn on descriptor codes 3, 5, and 7:

```
&DESC = '001010100000' (Command List)
```

When specifying a string of ones and zeros, omit the opening and closing parentheses.

2. In general, do not use descriptor codes that are reserved in MVS. Descriptor code 13 is used by NetView to flag messages that have already been automated on the MVS system.

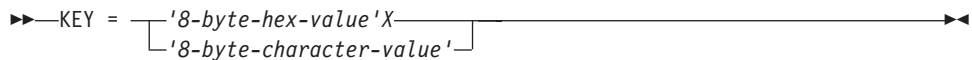
Setting descriptor code 13 will cause your messages to be ineligible for automation or display by NetView on your system. In a sysplex, however, these messages could still be eligible for automation and display on the other systems.

KEY, &KEY

Specifies a retrieval key represented by 8 bytes of hexadecimal or EBCDIC characters in the form of:

For REXX:

KEY



For Command List:

&KEY

►► &KEY = $\left[\begin{array}{l} \text{X'8-byte-hex-value'} \\ \text{8-byte-character-value} \end{array} \right]$

The following two examples use the hexadecimal notation:

KEY = 'C1C2C3C4F0F1F2F3' (REXX)

&KEY = 'C1C2C3C4F0F1F2F3' (Command List)

The following example uses character notation:

KEY = ABCD1234

Notes:

1. All 8 bytes must be specified.
2. To use the MVS D R, KEY command, specify only EBCDIC characters for KEY.

MCSFLAG, &MCSFLAG

The MCSFLAGs to be associated with the WTOR, specified as a list of keywords in parentheses:

For REXX:

MCSFLAG

►► MCSFLAG = '(

BRDCST
CMD
HRDCPY
NOCPY
NOTIME
QREG0
REG0
REPLY
RESP

)'

For Command List:

&MCSFLAG

►► &MCSFLAG = '(

BRDCST
CMD
HRDCPY
NOCPY
NOTIME
QREG0
REG0
REPLY
RESP

)'

Notes:

1. More than one keyword can be specified. The delimiter is a comma or blank.
2. CMD cannot be retrieved when messages are received by NetView.

- For compatibility with previous releases of NetView, MCSFLAG can also be specified as a string of ones and zeros.

For example, to set MCSFLAG to QREG0 and NOCPY:

```
MCSFLAG = '01000001' (REXX)
```

```
&MCSFLAG = '01000001' (Command List)
```

When specifying a string of ones and zeros, omit the opening and closing parentheses.

MSGTOKEN, &MSGTOKEN

Is a numeric token associated with the WTOR in the form of:

For REXX:

MSGTOKEN

```
▶▶—MSGTOKEN = number_1-4294967295—▶▶
```

For Command List:

&MSGTOKEN

```
▶▶—&MSGTOKEN = number_1-4294967295—▶▶
```

This token can be used to subsequently delete the operator message using the DOM command with the TOKEN option. See “Usage Notes” on page 336 for more information.

MSGTYP, &MSGTYP

The system message type flags for the WTO, specified as a list of keywords in parentheses:

For REXX:

MSGTYP

```
▶▶—MSGTYP = '(

|          |
|----------|
| JOBNAMES |
| SESS     |
| STATUS   |

)'—▶▶
```

For Command List:

&MSGTYP

```
▶▶—&MSGTYP = '(

|          |
|----------|
| JOBNAMES |
| SESS     |
| STATUS   |

)'—▶▶
```

Notes:

- More than one keyword can be specified. The delimiter is a comma or blank.

- For compatibility with previous releases of NetView, MSGTYP can also be specified as a string of ones and zeros. For example:

```
&MSGTYP = '010' (Command List)
```

ROUTCDE, &ROUTCDE

The system routing codes, specified as a list of numbers in parentheses:

For REXX:

ROUTCDE

```
▶▶—ROUTCDE = '(  
                  └──┬──┘  
                  number_1-128  
                  └──┬──┘  
                  , )'
```

For Command List:

&ROUTCDE

```
▶▶—&ROUTCDE = '(  
                  └──┬──┘  
                  number_1-128  
                  └──┬──┘  
                  , )'
```

You can also specify specific numbers and a range of numbers, for example:

```
&ROUTCDE = '(2,4-7)' (Command List)
```

Note: For compatibility with previous releases of NetView, ROUTCDE can also be specified as a string of ones and zeros. For example, to turn on codes 3, 5, and 7:

```
&ROUTCDE = '001010100000' (Command List)
```

When specifying a string of ones and zeros, omit the opening and closing parentheses, as shown in the example above.

SYSCONID, &SYSCONID

The destination console identifier for the WTOR. This can be either a name or a decimal number in the range of 1–99, as shown:

For REXX:

SYSCONID

```
▶▶—SYSCONID = └──console_number_1-99──┘  
                  └──'console_name'──┘
```

For Command List:

&SYSCONID

```
▶▶—&SYSCONID = └──console_number_1-99──┘  
                  └──console_name──┘
```

Notes:

1. Set on MCSFLAG bit 1 (REG0) or bit 2 (QREG0), or both, if you want SYSCONID to be used or checked.
2. The WTOR command updates SYSCONID to indicate which operator replied to the WTOR.
3. The console name you specify must be active. If it is not, a “nonvalid SYSCONID” return code is received.
4. MVS reserves some console names. Currently, the following names are reserved:
 - HC
 - INSTREAM
 - INTERNAL
 - UNKNOWN

These reserved names, and the corresponding console IDs, cannot be used as SYSCONID values for input to the WTOR command.

5. Console names can be used only if your MVS system is release 4.1 or later and you are using NetView for MVS.

Usage Notes

Consider the following when using the WTOR command:

- The values of these variables determine how the WTOR command is processed. The variables provide the same input as the keywords on an MVS WTOR macro. If a command list does not set the variables before issuing the WTOR command, their values default to the current system values.
- For REXX command lists, the current system values are contained in the functions that correspond to the variable names. For example, the current system value for the REXX SYSCONID variable is contained in the SYSCONID() function.
- The WTOR command returns values in the following REXX variables:
 - RC
 - SMSGID
 - SYSCONID
 - WTOREPLY
- The WTOR command returns values in the following NetView command list language control variables:
 - &RETCODE
 - &SMSGID
 - &SYSCONID
 - &WTOREPLY

Return Codes

The return code (RC or &RETCODE) indicates the processing results as follows:

Return Code	Meaning
4	The message was sent with truncated text.
8	Storage is not available to continue processing.
16	Internal processing failed.
100	There was no message text, or the command was running under a PPT.
104	SYSCONID length was not valid, or SYSCONID

	was not specified but is required because MCSFLAG bit 1 (REG0) or bit 2 (QREG0), or both, were set on.
108	SYSCONID value is not valid.
112	The task is posted to terminate.
116	The WTOREPLY command list dictionary update failed.
120	An internal decimal conversion failed.
124	The command list dictionary update failed.
138	The command was not invoked from a command procedure.
140	KEY length is not valid.
148	MSGTOKEN length is not valid.
152	MSGTOKEN value is not valid.
156	CART length is not valid.
164	ROUTCDE value is not valid.
168	MSGTYP value is not valid.
172	MCSFLAG value is not valid.
176	DESC value is not valid.

A return code with a value greater than 200 indicates that the return code was passed from the MVS WTOR macro. Subtract 200 from the value of the return code. The new value corresponds to the return code that was passed from the MVS WTOR macro. For example, if you receive a return code of 208, refer to the MVS library for the meaning of return code 8 from the MVS WTOR macro.

Examples

Example: Issuing WTOR Commands

The following example displays how to issue WTOR commands:

```

/*****
/*  WTOEXAM  COMMAND LIST                               */
/*  -----  -----  -----                          */
/*
/*  FUNCTION: THIS NETVIEW REXX COMMAND LIST DEMONSTRATES THE ABILITY */
/*              TO ISSUE WTO, WTOR, AND DOM COMMANDS.  IT SHOWS HOW   */
/*              TO CONTROL VARIOUS CONTROL VARIABLES THAT            */
/*              AFFECT ATTRIBUTES OF THE WTO.                          */
/*
/*  STEPS:                                                 */
/*  WTO A SERIES OF MESSAGES TO THE SYSTEM CONSOLE THAT HAVE        */
/*  DIFFERENT ATTRIBUTES, AS DEFINED IN MVS/XA SYSTEM              */
/*  MACROS AND FACILITIES VOLUME 2 UNDER THE                       */
/*  WTO MACRO.                                                     */
/*  SAVE THE MESSAGE IDS FOR THE WTO'ED MESSAGES.                 */
/*  DELETE THE MESSAGES FROM THE CONSOLE.                          */
/*  ISSUE AND RETRIEVE THE RESPONSE TO A WTOR COMMAND.             */
/*
/*****
/*****
/*  ISSUE SOME WTOS                                             */
/*****
ROUTCDE = '(1)'          /* MASTER CONSOLE ACTION */
DESC = '(1)'            /* SYSTEM FAILURE        */
'WTO THIS MESSAGE IS URGENT' /* ISSUE THE MESSAGE    */
MSGID_FOR_FIRST_MESSAGE=MSGID /* SAVE THE MESSAGE ID  */
ROUTCDE = '(2)'        /* MASTER CONSOLE INFO  */
DESC = '(3)'           /* EVENTUAL ACTION REQUIRED */
'WTO THIS MESSAGE CAN BE HANDLED LATER' /* ISSUE THE MESSAGE    */
MSGID_FOR_SECOND_MESSAGE=MSGID /* SAVE THE MESSAGE ID  */
/*****

```

```

/* ISSUE A MLWTO */
/*****/
ROUTCDE = '(2)' /* MASTER CONSOLE INFO */
DESC = '(3)' /* EVENTUAL ACTION REQUIRED */
LINETYPE='C' /* THIS IS A CONTROL LINE */
'WTO THE FIRST LINE OF A MLWTO' /* ISSUE THE FIRST LINE */
LINETYPE='D' /* THIS IS A DATA LINE */
'WTO THE SECOND LINE OF A MLWTO' /* ISSUE THE SECOND LINE */
LINETYPE='DE' /* THIS IS THE LAST DATA LINE */
/*
'WTO THE THIRD LINE OF A MLWTO' /* ISSUE THE THIRD LINE */
MSGID_FOR_THIRD_MESSAGE=MSGID /* SAVE THE MESSAGE ID */
/*****/
/* DEMONSTRATE THE ABILITY TO DELETE OPERATOR MESSAGES (DOM) */
/*****/
'WAIT 5 SECONDS' /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_FIRST_MESSAGE /* RESTORE THE MESSAGE ID */
'DOM' /* DELETE THE MESSAGE */
'WAIT 5 SECONDS' /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_SECOND_MESSAGE /* RESTORE THE MESSAGE ID */
'DOM' /* DELETE THE MESSAGE */
'WAIT 5 SECONDS' /* LEAVE MESSAGE FOR 5 SECS */
MSGID=MSGID_FOR_THIRD_MESSAGE /* RESTORE THE MESSAGE ID */
'DOM' /* DELETE THE MESSAGE */
/*****/
/* ASK THE OPERATOR FOR INFORMATION VIA A WTOR */
/*****/
'WTOR ENTER SOME DATA:' /* ISSUE THE WTOR */
SAY 'THE DATA ENTERED AT THE CONSOLE WAS: "WTOREPLY"'
EXIT

```




File Number: S370/4300/30XX-50
Program Number: 5697-B82



Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC31-8858-00

