Tivoli® IBM Tivoli NetView for z/OS

Version 5 Release 4

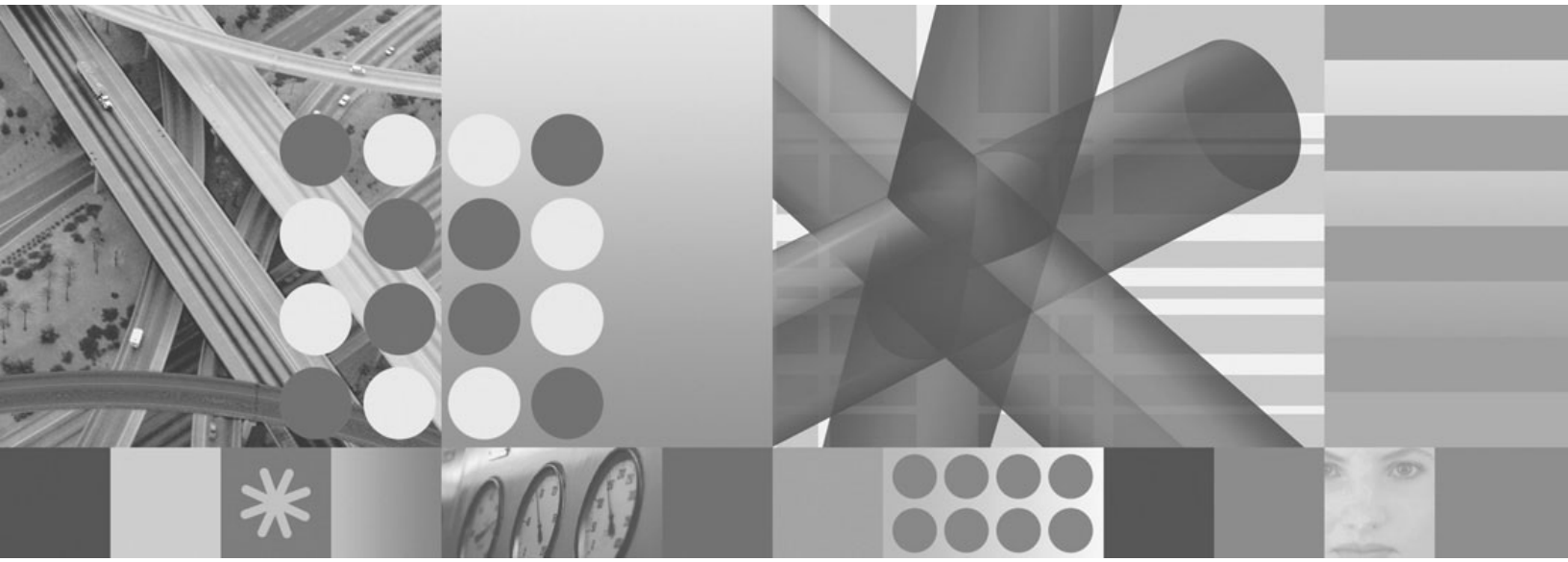
User's Guide: NetView Management Console

GC31-8852-03

Tivoli® IBM Tivoli NetView for z/OS

IBM

Version 5 Release 4



User's Guide: NetView Management Console

GC31-8852-03

Note
Before using this information and the product it supports, read the information in "Notices" on page 203.

This edition applies to version 5, release 4 of IBM Tivoli NetView for z/OS (product number 5697-ENV) and to all subsequent versions, releases, and modifications until otherwise indicated in new editions.

This edition replaces GC31-8852-02.

Contents

Figures

About this publication

The IBM® Tivoli® NetView® for z/OS® product provides advanced capabilities that you can use to maintain the highest degree of availability of your complex, multi-platform, multi-vendor networks and systems from a single point of control. This publication, the *IBM Tivoli NetView for z/OS User's Guide: NetView Management Console*, provides information about the NetView management console interface to the NetView product. For more detailed information about specific functions, see the NetView management console online help.

Intended audience

This publication is for network operators and system programmers who use the NetView management console.

Publications

This section lists publications in the IBM Tivoli NetView for z/OS library and related documents. It also describes how to access Tivoli publications online and how to order Tivoli publications.

IBM Tivoli NetView for z/OS library

The following documents are available in the IBM Tivoli NetView for z/OS library:

- *Administration Reference*, SC31-8854, describes the NetView program definition statements required for system administration.
- *Application Programmer's Guide*, SC31-8855, describes the NetView program-to-program interface (PPI) and how to use the NetView application programming interfaces (APIs).
- *Automation Guide*, SC31-8853, describes how to use automated operations to improve system and network efficiency and operator productivity.
- *Command Reference Volume 1 (A-N)*, SC31-8857, and *Command Reference Volume 2 (O-Z)*, SC31-8858, describe the NetView commands, which can be used for network and system operation and in command lists and command procedures.
- *Customization Guide*, SC31-8859, describes how to customize the NetView product and points to sources of related information.
- *Data Model Reference*, SC31-8864, provides information about the Graphic Monitor Facility host subsystem (GMFHS), SNA topology manager, and MultiSystem Manager data models.
- *Installation: Configuring Additional Components*, SC31-8874, describes how to configure NetView functions beyond the base functions.
- *Installation: Configuring Graphical Components*, SC31-8875, describes how to install and configure the NetView graphics components.
- *Installation: Configuring the Tivoli NetView for z/OS Enterprise Management Agent*, SC31-6969, describes how to install and configure the NetView for z/OS Enterprise Management Agent.
- *Installation: Getting Started*, SC31-8872, describes how to install and configure the base NetView functions.

- *Installation: Migration Guide*, SC31-8873, describes the new functions provided by the current release of the NetView product and the migration of the base functions from a previous release.
- *IP Management*, SC27-2506, describes how to use the NetView product to manage IP networks.
- *Messages and Codes Volume 1 (AAU-DSI)*, SC31-6965, and *Messages and Codes Volume 2 (DUI-IHS)*, SC31-6966, describe the messages for the NetView product, the NetView abend codes, the sense codes that are included in NetView messages, and generic alert code points.
- *Programming: Assembler*, SC31-8860, describes how to write exit routines, command processors, and subtasks for the NetView product using assembler language.
- *Programming: Pipes*, SC31-8863, describes how to use the NetView pipelines to customize a NetView installation.
- *Programming: PL/I and C*, SC31-8861, describes how to write command processors and installation exit routines for the NetView product using PL/I or C.
- *Programming: REXX and the NetView Command List Language*, SC31-8862, describes how to write command lists for the NetView product using the Restructured Extended Executor language (REXX) or the NetView command list language.
- *Resource Object Data Manager and GMFHS Programmer's Guide*, SC31-8865, describes the NetView Resource Object Data Manager (RODM), including how to define your non-SNA network to RODM and use RODM for network automation and for application programming.
- *Security Reference*, SC31-8870, describes how to implement authorization checking for the NetView environment.
- *SNA Topology Manager Implementation Guide*, SC31-8868, describes planning for and implementing the NetView SNA topology manager, which can be used to manage subarea, Advanced Peer-to-Peer Networking, and TN3270 resources.
- *Troubleshooting Guide*, GC27-2507, provides information about documenting, diagnosing, and solving problems that might occur in using the NetView product.
- *Tuning Guide*, SC31-8869, provides tuning information to help achieve certain performance goals for the NetView product and the network environment.
- *User's Guide: Automated Operations Network*, GC31-8851, describes how to use the NetView Automated Operations Network (AON) component, which provides event-driven network automation, to improve system and network efficiency. It also describes how to tailor and extend the automated operations capabilities of the AON component.
- *User's Guide: NetView*, GC31-8849, describes how to use the NetView product to manage complex, multivendor networks and systems from a single point.
- *User's Guide: NetView Management Console*, GC31-8852, provides information about the NetView management console interface of the NetView product.
- *User's Guide: Web Application*, SC32-9381, describes how to use the NetView Web application to manage complex, multivendor networks and systems from a single point.
- *Licensed Program Specifications*, GC31-8848, provides the license information for the NetView product.
- *Program Directory for IBM Tivoli NetView for z/OS US English*, GI10-3194, contains information about the material and procedures that are associated with installing the IBM Tivoli NetView for z/OS product.

- *Program Directory for IBM Tivoli NetView for z/OS Japanese*, GI10-3210, contains information about the material and procedures that are associated with installing the IBM Tivoli NetView for z/OS product.
- *IBM Tivoli NetView for z/OS V5R4 Online Library*, SK2T-6175, contains the publications that are in the NetView for z/OS library. The publications are available in PDF, HTML, and BookManager® formats.

Related publications

You can find additional product information on the NetView for z/OS Web site:

http://www.ibm.com/software/tivoli/products/netview-zos/

For information about the NetView Bridge function, see *Tivoli NetView for OS/390 Bridge Implementation*, SC31-8238-03 (available only in the V1R4 library).

Accessing terminology online

The *Tivoli Software Glossary* includes definitions for many of the technical terms related to Tivoli software. The *Tivoli Software Glossary* is available at the following Tivoli software library Web site:

http://publib.boulder.ibm.com/tividd/glossary/tivoliglossarymst.htm

The IBM Terminology Web site consolidates the terminology from IBM product libraries in one convenient location. You can access the Terminology Web site at the following Web address:

http://www.ibm.com/software/globalization/terminology/

For a list of NetView for z/OS terms and definitions, refer to the IBM Terminology Web site. The following terms are used in this library:

NetView
For the following products:
- Tivoli NetView for z/OS version 5 release 4
- Tivoli NetView for z/OS version 5 release 3
- Tivoli NetView for z/OS version 5 release 2
- Tivoli NetView for z/OS version 5 release 1
- Tivoli NetView for OS/390® version 1 release 4

MVS™ For z/OS operating systems

MVS element
For the BCP element of the z/OS operating system

CNMCMD
For the CNMCMD member and the members that are included in it using the %INCLUDE statement

CNMSTYLE
For the CNMSTYLE member and the members that are included in it using the %INCLUDE statement

PARMLIB
For SYS1.PARMLIB and other data sets in the concatenation sequence

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.

Table 1 defines the Windows® and UNIX® terms as they are used with the NetView management console topology console and NetView management console topology server.

Table 1. NetView Management Console Topology Server and Topology Console Platforms

Term	NetView Management Console Topology Console	NetView Management Console Topology Server
Windows	• Windows XP Professional • Windows Server 2003 • Windows Server 2008 • Windows Vista Desktop	• Windows Server 2003 • Windows Server 2008
UNIX	• Solaris • Red Hat Enterprise Linux® Desktop • SuSE Linux Enterprise Desktop	• AIX® • Red Hat Enterprise Linux for IBM eSeries System z® and S/390® • SUSE Linux Enterprise Server for System z

Using NetView for z/OS online help

The following types of NetView for z/OS mainframe online help are available, depending on your installation and configuration:

- General help and component information
- Command help
- Message help
- Sense code information
- Recommended actions

Using LookAt to look up message explanations

LookAt is an online facility that you can use to look up explanations for most of the IBM messages you encounter, and for some system abends and codes. Using LookAt to find information is faster than a conventional search because, in most cases, LookAt goes directly to the message explanation.

You can use LookAt from the following locations to find IBM message explanations for z/OS elements and features, z/VM®, VSE/ESA, and Clusters for AIX and Linux systems:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ .
- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e system to access IBM message explanations, using LookAt from a TSO/E command line (for example, TSO/E prompt, ISPF, or z/OS UNIX System Services running OMVS).
- Your Microsoft® Windows workstation. You can install LookAt directly from the *z/OS Collection* (SK3T-4269) or the *z/OS and Software Products DVD Collection* (SK3T-4271) and use it from the resulting Windows graphical user interface (GUI). The command prompt (also known as the DOS command line) version can still be used from the directory in which you install the Windows version of LookAt.

- Your wireless handheld device. You can use the LookAt Mobile Edition from http://www.ibm.com/systems/z/os/zos/bkserv/lookat/lookatm.html with a handheld device that has wireless access and an Internet browser.

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from the following locations:

- A CD in the *z/OS Collection* (SK3T-4269).
- The *z/OS and Software Products DVD Collection* (SK3T-4271).
- The LookAt Web site. Click **Download** and then select the platform, release, collection, and location that you want. More information is available in the LOOKAT.ME files that is available during the download process.

Accessing publications online

The documentation DVD, *IBM Tivoli NetView for z/OS V5R4 Online Library*, SK2T-6175, contains the publications that are in the product library. The publications are available in PDF, HTML, and BookManager formats. Refer to the readme file on the DVD for instructions on how to access the documentation.

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli Information Center Web site at http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp.

Note: If you print PDF documents on other than letter-sized paper, set the option in the **File → Print** window that enables Adobe® Reader to print letter-sized pages on your local paper.

Ordering publications

You can order many Tivoli publications online at http://www.elink.ibmlink.ibm.com/publications/servlet/pbi.wss

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, contact your software account representative to order Tivoli publications. To locate the telephone number of your local representative, perform the following steps:

1. Go to http://www.elink.ibmlink.ibm.com/publications/servlet/pbi.wss.
2. Select your country from the list and click **Go**.
3. Click **About this site** to see an information page that includes the telephone number of your local representative.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully. 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.

For additional information, see the Accessibility appendix in the *User's Guide: NetView*.

Tivoli technical training

For Tivoli technical training information, refer to the following IBM Tivoli Education Web site at http://www.ibm.com/software/tivoli/education.

Downloads

Clients and agents, NetView product demonstrations, and several free NetView applications can be downloaded from the NetView for z/OS support Web site:

http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliNetViewforzOS.html

In the "IBM Tivoli for NetView for z/OS support" pane, click **Download** to go to a page where you can search for or select downloads.

These applications can help with the following tasks:

- Migrating customization parameters and initialization statements from earlier releases to the CNMSTUSR member and command definitions from earlier releases to the CNMCMDU member.
- Getting statistics for your automation table and merging the statistics with a listing of the automation table
- Displaying the status of a job entry subsystem (JES) job or canceling a specified JES job
- Sending alerts to the NetView program using the program-to-program interface (PPI)
- Sending and receiving MVS commands using the PPI
- Sending Time Sharing Option (TSO) commands and receiving responses

Support for problem solving

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

Online
Go to the IBM Software Support site at http://www.ibm.com/software/support/probsub.html and follow the instructions.

IBM Support Assistant
The IBM Support Assistant (ISA) is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. The ISA provides quick access to support-related information and serviceability tools for problem determination. To install the ISA software, go to http://www.ibm.com/software/support/isa/.

Troubleshooting information
For more information about resolving problems with the NetView for z/OS product, see the *IBM Tivoli NetView for z/OS Troubleshooting Guide*. Additional support for the NetView for z/OS product is available through the NetView user group on Yahoo at http://groups.yahoo.com/group/NetView/. This support is for NetView for z/OS customers only, and registration is required. This forum is monitored by NetView developers who answer questions and provide guidance. When a problem with the code is found, you are asked to open an official problem management record (PMR) to obtain resolution.

Conventions used in this publication

This publication uses several conventions for special terms and actions, operating system-dependent commands and paths, and command syntax.

Typeface conventions

This publication uses the following typeface conventions:

Bold

- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:**, and **Operating system considerations**:)
- Keywords and parameters in text

Italic

- Citations (examples: titles of publications, diskettes, and CDs
- Words defined in text (example: a nonswitched line is called a *point-to-point line*)
- Emphasis of words and letters (words as words example: "Use the word *that* to introduce a restrictive clause."; letters as letters example: "The LUN address must start with the letter *L*.")
- New terms in text (except in a definition list): a *view* is a frame in a workspace that contains data.
- Variables and values you must provide: ... where *myname* represents...

`Monospace`

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

Operating system-dependent variables and paths

For workstation components, this publication uses the UNIX convention for specifying environment variables and for directory notation.

When using the Windows command line, replace **$***variable* with **%***variable***%** for environment variables and replace each forward slash (**/**) with a backslash (\) in directory paths. The names of environment variables are not always the same in the Windows and UNIX environments. For example, %TEMP% in Windows environments is equivalent to $TMPDIR in UNIX environments.

Note: If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Syntax diagrams

Read syntax diagrams from left-to-right, top-to-bottom, following the horizontal line (the main path). This section describes how syntax elements are shown in syntax diagrams.

Symbols

The following symbols are used in syntax diagrams:

►► Marks the beginning of the command syntax.

► Indicates that the command syntax is continued.

| Marks the beginning and end of a fragment or part of the command syntax.

►◄ Marks the end of the command syntax.

Parameters

The following types of parameters are used in syntax diagrams:

Required Required parameters are shown on the main path.

Optional Optional parameters are shown below the main path.

Default Default parameters are shown above the main path. In parameter descriptions, default parameters are underlined.

Syntax diagrams do not rely on highlighting, brackets, or braces. In syntax diagrams, the position of the elements relative to the main syntax line indicates whether an element is required, optional, or the default value.

Parameters are classified as keywords or variables. Keywords are shown in uppercase letters. Variables, which represent names or values that you supply, are shown in lowercase letters and are either italicized or, in NetView help and BookManager publications, displayed in a differentiating color.

In the following example, the USER command is a required keyword parameter, *user_id* is a required variable parameter, and *password* is an optional variable parameter.

Punctuation and parentheses

You must include all punctuation that is shown in the syntax diagram, such as colons, semicolons, commas, minus signs, and both single and double quotation marks.

When an operand can have more than one value, the values typically are enclosed in parentheses and separated by commas. For a single value, the parentheses typically can be omitted. For more information, see “Multiple operands or values” on page xx.

If a command requires positional commas to separate keywords and variables, the commas are shown before the keywords or variables.

When examples of commands are shown, commas are also used to indicate the absence of a positional operand. For example, the second comma indicates that an optional operand is not being used:

```
COMMAND_NAME opt_variable_1,,opt_variable_3
```

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

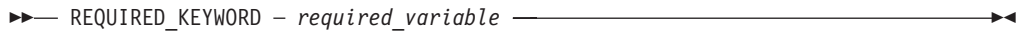
Abbreviations

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

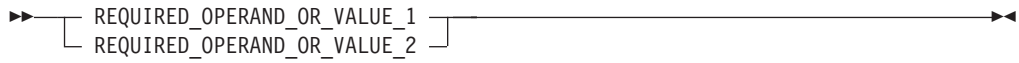
Syntax examples

This section show examples for the different uses of syntax elements.

Required syntax elements: Required keywords and variables are shown on the main syntax line. You must code required keywords and variables.

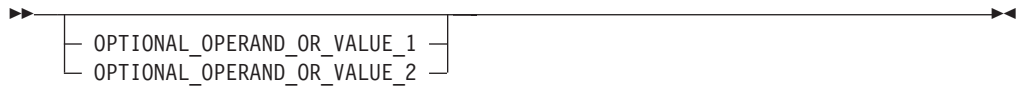


If multiple mutually exclusive required keywords or variables are available to choose from, they are stacked vertically in alphanumeric order.



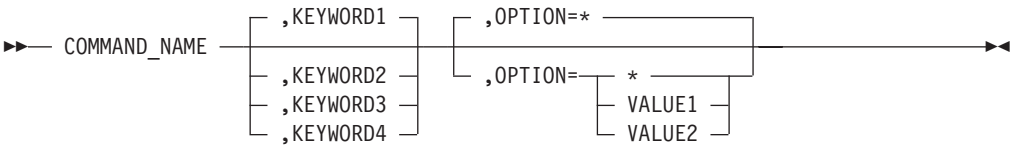
Optional syntax elements: Optional keywords and variables are shown below the main syntax line. You can choose not to code optional keywords and variables.

If multiple mutually exclusive optional keywords or variables are available to choose from, they are stacked vertically in alphanumeric order below the main syntax line.

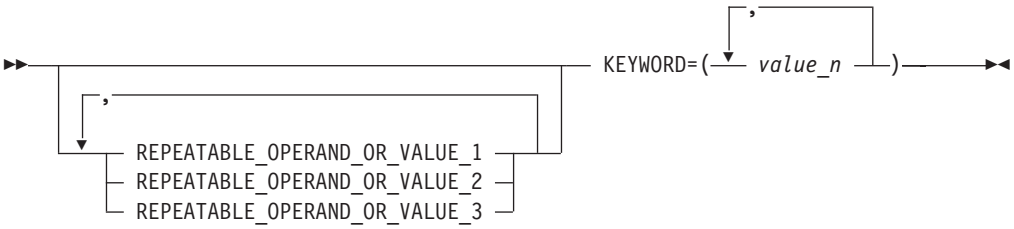


Default keywords and values: Default keywords and values are shown above the main syntax line in one of the following ways:

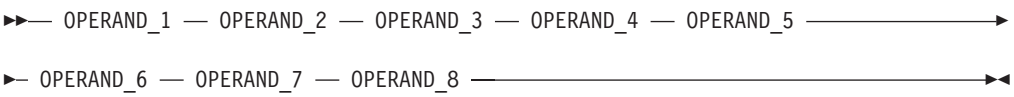
- A default keyword is shown only above the main syntax line. You can specify this keyword or allow it to default. The following syntax example shows the default keyword KEYWORD1 above the main syntax line and the rest of the optional keywords below the main syntax line.
- If an operand has a default value, the operand is shown both above and below the main syntax line. A value below the main syntax line indicates that if you specify the operand, you must also specify either the default value or another value shown. If you do not specify the operand, the default value above the main syntax line is used. The following syntax example shows the default values for operand OPTION=* above and below the main syntax line.



Multiple operands or values: An arrow returning to the left above a group of operands or values indicates that more than one can be selected or that a single one can be repeated.



Syntax that is longer than one line: If a diagram is longer than one line, each line that is to be continued ends with a single arrowhead and the following line begins with a single arrowhead.



Syntax fragments: Some syntax diagrams contain syntax fragments, which are used for lengthy, complex, or repeated sections of syntax. Syntax fragments follow the main diagram. Each syntax fragment name is mixed case and is shown in the main diagram and in the heading of the fragment. The following syntax example shows a syntax diagram with two fragments that are identified as Fragment1 and Fragment2.

Fragment1

Fragment2

Part 1. Overview

Chapter 1. Introduction to the NetView Management Console

The NetView management console of NetView for z/OS graphically displays the resources that represent a network, a portion of a network, or a group of networks at various levels of detail. These views show the network and systems resources that you are monitoring. When you monitor a network, resource status changes are reflected graphically in the views.

What You Can Do with NetView Management Console

You can use the NetView management console to perform the following tasks:

- Monitor and control large portions of complex business systems.
- View the topology and connectivity of your network graphically.
- Monitor the overall state of a network or a portion of a network through aggregates, which represent the combined status of a group of related applications and resources.
- Navigate easily from an aggregate to a real resource that is failing.
- Mark resources for your own purposes; for example, to show that they are being serviced.
- Display a list of events or status changes received for a selected resource.
- Issue predefined commands from context menus, or issue your own commands.
- Stop and restart selected resources.
- Monitor and manage multiple NetView programs.
- Cycle through open views automatically, at intervals.
- Build custom view and aggregate resource collections.
- Monitor resources by *exception*, to show on the screen only when the resources need the attention of the operator.

NetView management console provides the added capability to create demonstrations from your live NetView management console views. These demonstrations can be used to aid in a variety of activities including:

- Operator training within your corporate setting
- Showing your customers what you can offer them
- NetView management console advocacy

You can create these demonstrations by capturing live NetView management console views and integrating them into the demonstration. This makes your demonstrations look and feel like your real NetView management console system. For complete information about creating demonstrations, see Chapter 6, "Creating a Demonstration," on page 43.

How the NetView Management Console Works

The NetView management console consists of a *server* and a Java-based *console*, which are generically referred to in this book and the online help as the *topology server* and *topology console*.

The topology console graphically displays systems and networking information provided by Tivoli NetView for z/OS. This information is displayed as Resource

Object Data Manager (RODM) based views and is only available if a conversation is set up between the topology server and the NetView host. See Chapter 9, “Operating the NetView Management Console,” on page 77 for information about setting up this conversation using the NETCONV command.

Topology Server

The topology server interacts with Graphic Monitor Facility host subsystem (GMFHS) and RODM and provides information for display on the topology console.

The topology server furnishes the topology console with a set of tasks that are applicable for a resource. These tasks are shown in context-sensitive menus on the topology console.

The topology server also stores files on behalf of the topology console, including icons, backgrounds, help files, log files, customized views, and settings.

Note that you must have a conversation between the topology server and the NetView host set up using the NETCONV command. See Chapter 9, “Operating the NetView Management Console,” on page 77 for information about setting up communications with the NetView host.

Topology Console

The topology console graphically displays network information from the topology server. It displays systems and networking views. It uses color to indicate the status of each resource, and the status of the entire network. As the topology server receives configuration and status updates about the network, it updates the topology console. This enables the topology console to always display the real-time configuration of the network.

The topology console is a Java-based, platform-independent application.

Real and Aggregate Resources

The topology console can display both real and aggregate resources. A real resource is a single component or link (connection) in a network. An aggregate resource represents a collection of real or aggregate resources. When displayed on the topology console, an aggregate resource has a plus (+) in the lower-right corner, and an aggregate link has a plus (+) in the center of the link. The status of an aggregate resource is a reflection of the status of its underlying real resources. When you monitor an aggregate resource, you are monitoring the overall status of a portion of the network.

You can define real resources as critical and noncritical using aggregation priority. Critical resources are resources that are considered important to the operation of the network and are assigned a high aggregation priority (1 or greater). If the status of a critical resource changes to unsatisfactory, for example, the status of the aggregate resource also changes to degraded. Noncritical resources have low aggregation priorities (0).

You can set the aggregation priority on a resource from the Resource Properties notebook for a selected resource.

When a real resource changes status, the status of an aggregate of which the resource is a part is determined in the following way:

- The status of the parent aggregate of the resource is determined by the statuses of the real resource and its siblings.
- The status of the grandparent aggregate of the resource is determined by the statuses of the real resources under the parent.

Chapter 2. Understanding Views

The NetView management console graphically displays systems and networking information provided by the NetView host. This information is displayed as Resource Object Data Manager (RODM) based views and is available only if a conversation is set up between the topology server and the NetView host.

RODM-Based Views

RODM-based views are predefined in RODM or are dynamically built based on definitions in RODM. The Graphic Monitor Facility host subsystem (GMFHS) must be available to display RODM-based views and can include the following views:

- Network views
- Exception views
- Configuration views
- More detail views
- Locate failing resource views

RODM-based views contain resources which are defined by the SNA topology manager, MultiSystem Manager, user applications, loader files, and so on.

Network Views

Network views and the resources displayed in them are defined in RODM. Figure 1 shows a network view.

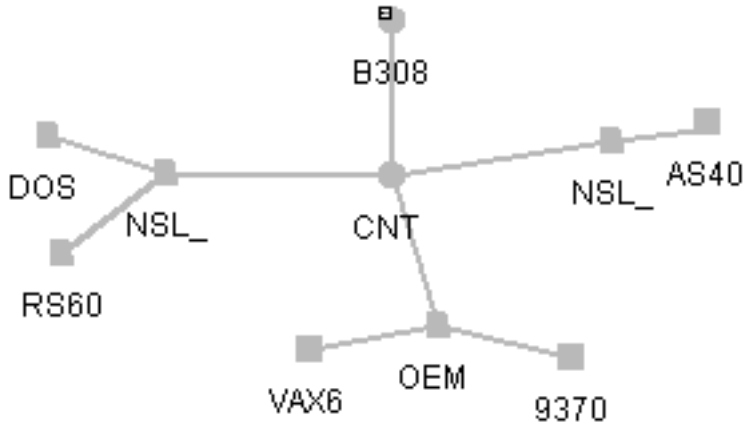


Figure 1. Network View

Exception Views

An exception view is a view that typically shows only resources that are not functioning properly, as defined by the exception criteria you defined in RODM. Figure 2 on page 8 shows an exception view.

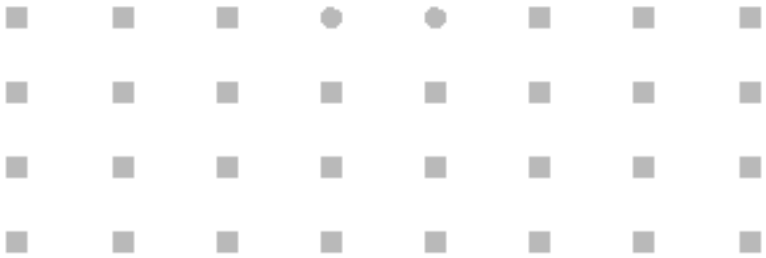


Figure 2. Exception View

Configuration Views

You can request these types of configuration views: parents, children, peers, logical, physical, logical and physical, and backbone. All relationships must have been previously defined in RODM.

Configuration Parents View

Figure 3 displays the configuration of a resource (not the entire connectivity) to its owning node.

Figure 3. Configuration Parents View

Configuration Children View

Figure 4 on page 9 shows the selected resource and all of its children.

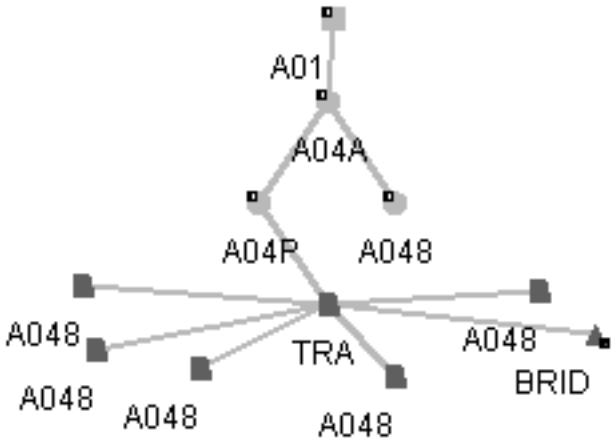


Figure 4. Configuration Children View

Configuration Peers View

Figure 5 shows a view containing resources in the network that are arranged in a configuration based on a peer relationship between resources.

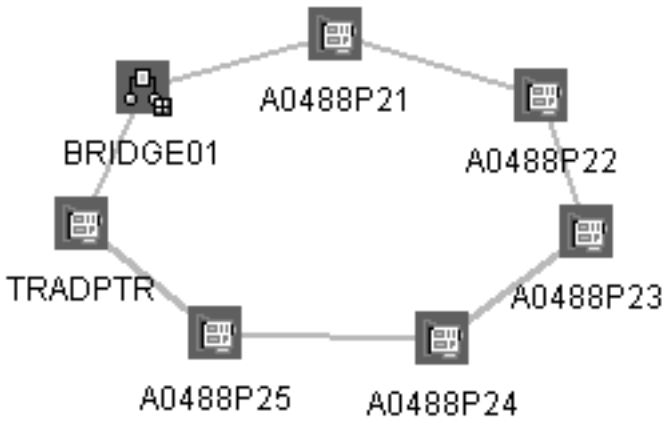


Figure 5. Configuration Peers View

Configuration Logical View

Figure 6 on page 10 shows a view containing resources in the network that are arranged in a configuration based on a logical relationship between resources.

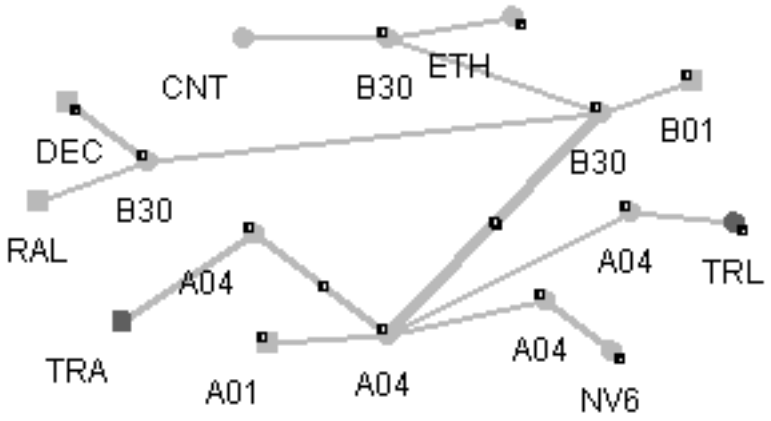


Figure 6. Configuration Logical View

Configuration Physical View

Figure 7 shows a view containing resources in the network that are arranged in a configuration based on a physical relationship between resources.

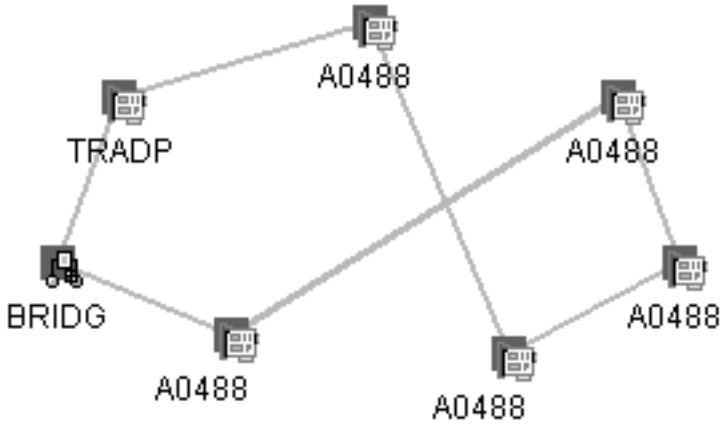


Figure 7. Configuration Physical View

Configuration Logical and Physical

Figure 8 on page 11 shows a view containing resources in the network that are arranged in a configuration based on a logical and physical relationship between resources.

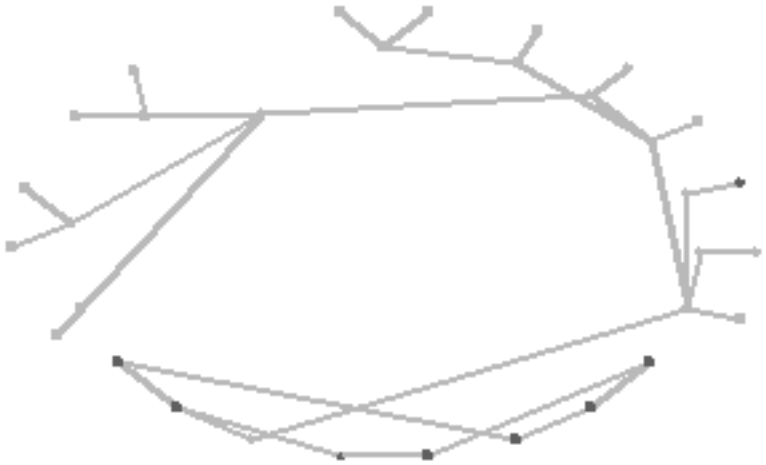


Figure 8. Configuration Logical and Physical View

Configuration Backbone View

Figure 9 shows a view containing resources in the network that are arranged in a configuration based on a subarea backbone relationship.

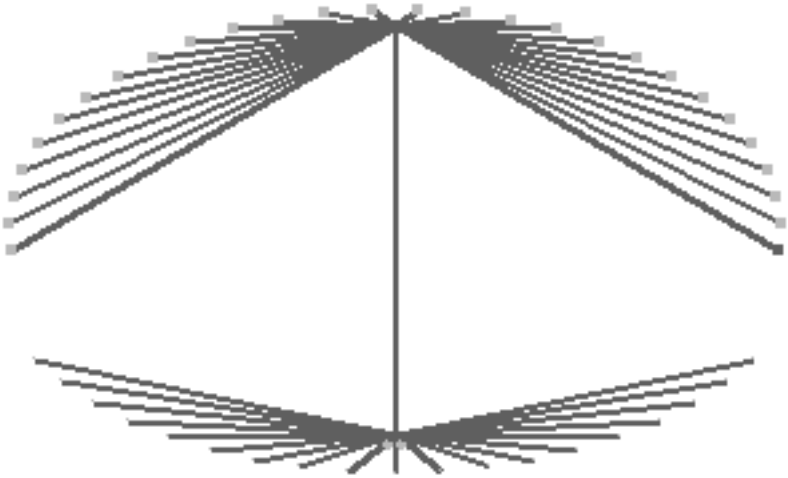


Figure 9. Configuration Backbone View

More Detail Views

When you request more detail about a selected resource, a view is displayed showing lower-level resources related to the selected resource. The More Detail function lets you navigate from high-level views to lower-level views. Figure 10 on page 12 shows the contents of a resource, or *more details* about the resource.

Figure 10. More Detail Views

Locate Failing Resources

Figure 11 shows a view which was created by selecting **Locate Failing Resources** on the context menu of an aggregate resource. This view displays all child real resources currently in an exception status.

Figure 11. Locate Failing Resource View

Customized Views

A customized view is a view that has been opened, changed, and saved using the Save View Customization function. You can use this function to save changes to views that are created dynamically in response to certain requests and to predefined RODM-based network views. If you are signed on as administrator, this function is available for the following types of views:

- Network views (predefined)
- Configuration views (both predefined and dynamically-built)
- More detail views (both predefined and dynamically-built)
- Locate failing resource views (dynamically-built)

If sign on as administrator, and dynamic views are customized, the Customized Dynamic Views node is displayed in the business tree.

Double-clicking a customized dynamic view in the business tree opens a snapshot of the view, but this snapshot is not an active view with real status and the latest topology changes. You can use it to see how dynamic views have been customized and to change the customization.

Note: Customized network views are shown in the business tree under Network Views.

Views Containing Resources for Which You Are Not Authorized

If you have defined span of control, some views can contain resources that you are not authorized to display because of your span authorization. When this occurs, the view is affected in one of the following ways, depending on your NetView customization:

- The unauthorized resources are not visible.
- The unauthorized resources are displayed as null nodes or null links, or both.

For more information about span of control, refer to the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

Views Containing Scheduled Resources

If you have defined NMCSTATUS policy definitions, a view can contain resources that are suspended from aggregation or are no longer receiving system status updates at the NetView management console. For more information about `NMCSTATUS` policy definitions, see the *IBM Tivoli NetView for z/OS Administration Reference*.

A resource that is suspended from aggregation because of a NMCSTATUS policy definition has a textual note attached to the suspend flag indicating why the resource was suspended. The note is displayed when either a Resource Properties or List Suspended Resources request is made.

A resource that is no longer receiving system status updates at the NetView management console topology console has a scheduled system status. The resource continues to receive system status updates in RODM but they are not sent to the NetView management console topology console while the resource is scheduled. As with other system statuses, you can customize the color of the scheduled system status on the Console Properties window.

Displaying Views in a Web Browser

You can configure the NetView management console topology console to function as a Web server. This enables the console to capture Topographic or Details NetView management console views and convert them into HTML and GIF files, which you can view in any Web browser.

Note: You can only display views that are open on the console.

If you want information about...	Refer to...
Displaying views in a Web browser	"Configuring a Web Browser to Display Views" on page 26.

Part 2. Installing and Customizing the NetView Management Console

Chapter 3. Installing the NetView Management Console

Installation instructions are provided in the installation README files. This chapter provides information on selecting the correct README file based on whether you are installing the topology server or topology console and information about configuring some parts of the NetView management console. For additional installation information about the various features of the NetView management console, see *IBM Tivoli NetView for z/OS Installation: Configuring Graphical Components*.

The README files contain installation instructions, hardware and software requirements, and late-breaking news. They are available in the following locations:

- The NetView CD-ROM in the README directory under the ENU or JPN subdirectories, where ENU represents English and JPN represents Japanese.
- The NetView product support Web page. From the NetView Web page, (http://www.ibm.com/software/tivoli/products/netview-zos/), click **Technical support**, which is on the left under IBM Tivoli NetView for z/OS. On the NetView product support page, go to the **Download** section.

Installing the Topology Console

For complete installation instructions for the topology console, see the EGVREAD1 README file (English) or the EGVREAD3 README file (Japanese).

Installing the Topology Server

For complete installation instructions for the topology server, see the EGVREAD2 README file (English) or the EGVREAD4 README file (Japanese).

Defining the NetView for z/OS User ID and Password on the Topology Server

Using the optional **hostcmd** command to send commands from the topology server (using a command prompt on the topology server workstation) to the z/OS environment requires a Tivoli NetView for z/OS operator ID and password or password phrase. If the operator ID and password or password phrase can be determined in any of the following three ways and in the following order, the **hostcmd** command does not prompt you for them:

1. The **tserver hostcmd** command is issued with the **-u** and **-p** options, which specify the user (operator) ID and password or password phrase. This method overrides the other 2 methods.
2. The **tserver hostcmdoper** command is used to set the operator ID and password or password phrase. This command encrypts the password or or password phrase and stores it on disk. This method is overridden by the previous method and overrides the following method.
3. The `ihsshstc.cfg` file is customized by using the following steps. This method is overridden by either of the other 2 methods.
   a. In a workstation command window, change to one of the following directories:
      - For Windows systems: `%BINDIR%\TDS\server\config`

- For UNIX systems: $BINDIR/TDS/server/config

b. Edit the ihsshstc.cfg file to specify the operator ID and the password or password phrase of the NetView operator for which the commands are to be run.
   - Specify the NetView operator ID in the OPER_ID parameter.
   - Specify the password or password phrase for that operator ID in the OPER_PW parameter.

Chapter 4. Customizing the NetView Management Console Topology Server

This chapter describes customization of the NetView management console topology server to perform specific functions.

Modifying the Server Properties File

You can customize the NetView management console topology server by modifying the `server.properties` file, which is provided with the topology server. One reason to customize this file is to set the defaults for the audit log, as described in "Auditing Functions and the Server.Properties File."

The `server.properties` file is a plain text configuration file that enables control of the following functions:

- Communications timeouts
- Performance tuning
- View appearance
- Audit log attributes

These functions are used by the server at run time and affect all consoles attached to the server. Usually, the settings in the `server.properties` file should be left at the default levels. The file contains a detailed description of the items available for configuration. The server.properties file is located in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\config`
- For UNIX: `$BINDIR/TDS/server/config`

To customize topology server functions, perform the following steps:

1. Make a backup copy of the `server.properties` file.
2. Open the `server.properties` file in a text editor and make the necessary changes.
3. Restart the topology server so that the changes can take affect.

Auditing Functions and the Server.Properties File

The `server.properties` file contains auditing function specifications that can be set to control the output and format of the auditing log.

Customizing the Functions to be Audited

You can determine which functions to audit in the `server.properties` file. The attribute for each of the functions that can be audited can have two possible values. If the attribute value is 1, the auditing is performed. If the attribute value is 0 (zero) then auditing is not performed. If a value that is not valid is assigned to the attribute, then the default value is used. Table 2 on page 20 describes the attributes and gives the default value for each.

Table 2. Auditing function attributes.

Attribute in `server.properties` file	Description of function to be (or not to be) audited	Default value
auditAggregationChange	Aggregation priority and threshold change requests	0
auditCommand	Listing commands to be run	1
auditCommandResponse	Listing responses to commands	0
auditConsoleSignonSignoff	Signing on and off of the NetView management console topology console	1
auditCPESignonSignoff	Signing on and off of the command profile editor (CPE) batch utility	1
auditFlagChange	Flag change and list suspended resources requests	0
auditNetconvUpDown	Running the NETCONV command to start and stop communication between the host and server	1
auditSendMessage	Sending messages	0
auditServerStartStop	Starting and stopping the NetView management console topology server	1
auditViewClose	Closing views	0
auditViewCustomizationSaveRemove	Saving, removing, and deleting customized views	0
auditViewOpenRequest	Requesting views from the NetView host	0
auditViewOpenResponse	Opening views	0

Customizing the Audit Log for Viewing

The ihsaudit.xml file is the log file which contains the data specified by the attributes in the `server.properties` file as described in the section "Customizing the Functions to be Audited" on page 19.

For example, an `ihsaudit.xml` file can contain audit entries indicating the start and initialization of a NetView management console topology server, netconv communication from the NetView management console topology server to a host, and a NetView management console topology console signing on to the NetView management console topology server. The following example contains such audit entries:

```
<?xml version="1.0" encoding="utf-8" ?>
<?xml-stylesheet href="../sample/ihsaudit.css" type="text/css" ?>

<auditLog>

<auditEntry>
  <date>2001-12-11 15:22:24</date>
  <nmcServer>
    <action>start</action>
  </nmcServer>
</auditEntry>

<auditEntry>
  <date>2001-12-11 15:23:09</date>
```

```
  <netconv>
    <action>up</action>
    <nvDomain>CNM01</nvDomain>
    <ipAddr>1.2.3.4</ipAddr>
    <hostname>netview.hostname</hostname>
  </netconv>
</auditEntry>

<auditEntry>
  <date>2001-12-11 15:23:10</date>
  <nmcServer>
    <action>initialized</action>
  </nmcServer>
</auditEntry>

<auditEntry>
  <date>2001-12-11 15:23:22</date>
  <nmcConsole>
    <action>signon</action>
    <id>netop1</id>
    <ipAddr>5.6.7.8</ipAddr>
    <hostname>server.hostname</hostname>
    <admin>yes</admin>
  </nmcConsole>
</auditEntry>

</auditLog>
```

Refer to Appendix E, "Auditing and the ihsaudit.xml File," on page 173 for information about elements in the `ihsaudit.xml` file.

The audit log can be viewed by a Web browser, or other application, which supports XML. To enable a Web browser or other XML application to display the contents of the audit log properly, header lines must be inserted in the ihsaudit.xml log file. These lines are set in the `server.properties` file and determine the appropriate style sheet to be used for the browser. The header lines are inserted only when the topology server creates a new ihsaudit.xml file, not every time an audit entry is written to the file.

Note: Ensure that your Web browser or XML application supports the format of the style sheet that you choose.

The audit log file is located in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\log`
- For UNIX: `$BINDIR/TDS/server/log`

Table 3 describes the XML attributes in the `server.properties` file.

Table 3. XML attributes set in the server.properties file.

Attribute in `server.properties` file	**Description of function**	**Default value**
`auditLogHeaderLine.1`	Sets the XML version	1.0
`auditLogHeaderLine.2`	Sets the style sheet to be used to enable the appropriate Web browser or XML application	ihsaudit.css

Example:

```
auditLogHeaderLine.1   =<?xml version="1.0" encoding="utf-8" ?>
auditLogHeaderLine.2   =<?xml -stylesheet href="../sample/ihsaudit.css"
                          type="text/css" ?>
```

Customizing the Size of the Audit Log

When the audit log file reaches the size specified with the auditLogFileMaxSize attribute in the server.properties file, it is renamed ihsaudit.bak. A new ihsaudit.xml file is then created. It is possible for the audit log to be slightly greater in size than specified by the auditLogFileMaxSize attribute. Audit entries are written to the log until the file size is equal to or greater than the value of auditLogFileMaxSize. For example, if the actual size of the file is 999 999 and the auditLogFileMaxSize is set to 1 000 000, then one more audit log entry is written to the file. Table 4 describes the auditLogFileMaxSize attribute in the server.properties file.

Table 4. Audit log size attribute set in the server.properties file.

Attribute in server.properties file	Description	Default value	Valid values
auditLogFileMaxSize	Determines the size of the audit log file, in megabytes	20	1–1000 (1 megabyte to 1 gigabyte)

If the value for the auditLogFileMaxSize attribute is set to 20, then the maximum file size is actually 20 000 000 bytes.

Notes:

1. If an incorrect value is entered for the auditLogFileMaxSize attribute, the default value is used.
2. If there was a previous ihsaudit.bak file, that file is deleted and replaced with the most current ihsaudit.xml backed up and renamed ihsaudit.bak.

Customizing the Length of Command Responses within the Audit Log

Audit log entries for command responses are written to the audit log if the auditCommandResponse value specified in the server.properties file is set to 1. The auditCommandResponse attribute is described in "Customizing the Functions to be Audited" on page 19. The maximum line length of these command responses can be specified in the server.properties file with the auditCommandResponseMaxSize attribute. When command responses exceed the length specified by this attribute, they are truncated in the audit log. Table 5 describes the auditCommandResponseMaxSize attribute in the server.properties file.

Table 5. Command response length attribute set in the server.properties file.

Attribute in server.properties file	Description	Default value	Valid values
auditCommandResponseMaxSize	Determines the maximum line length of command responses, in bytes, written to the audit log.	1000	0–10 000

Notes:

1. The 0 (zero) value indicates that the command response is not truncated.
2. If an incorrect value is entered, the default value 1 000, is used.
3. If the command response contains an ampersand, (&), then the truncated string can be a few characters shorter or longer than the auditCommandResponseMaxSize indicates. If an ampersand (&) has been

translated to a string of characters that can be browsed (&), for example, then the complete string is included in the truncated command response, regardless of the length of the string.

Customizing Special Characters to be Browsed

Without using the `auditTranslateToBrowseableChars` attribute in the `server.properties` file, the following set of characters can cause an error so that the ihsaudit.xml file cannot be opened in a browser:

- & (ampersand)
- < (less than)
- > (greater than)

The `auditTranslateToBrowseableChars` changes these characters to a string of characters that can be browsed within the following XML tags:

- `<cmd>`
- `<cmdResp>`
- `<locateName>`
- `<menuText>`
- `<msg>`
- `<note>`
- `<viewName>`

The `auditTranslateToBrowseableChars` attribute in the `server.properties` file translates these characters so that the ihsaudit.xml file can be viewed by a browser. The `auditTranslateToBrowseableChars` attribute in the `server.properties` file is described in Table 6.

Table 6. Characters to be browsed attribute set in the server.properties file.

Attribute in `server.properties` file	**Description**	**Default value**	**Valid values**
`auditTranslateToBrowseableChars`	When set to 1, this attribute translates the characters in the following way: • & to `&` • < to `<` • > to `>` When set to 0, these characters are not translated.	1	0 or 1

Note: If a value other than 0 or 1 is entered for this attribute, the default (1) is used.

Customizing the Replacement Character for Carriage Returns and Line Feeds

Command responses often contain carriage return and line feed characters. Some browsers are unable to display an audit entry that contains these characters. However, you can choose to replace the carriage return and line feed characters with a character or multiple characters of your choice. To insert characters for the

carriage return and line feed characters, use the auditCarriageReturnReplacement attribute and specify the characters to be used for delineation with the auditCarriageReturnReplacementValue attribute in the server.properties file as described in Table 7.

Table 7. Line return delineation attribute set in the server.properties file.

Attribute in server.properties file	Description	Default value	Valid values
auditCarriageReturnReplacement	Determines if the carriage returns and line feeds are to be replaced with a character or characters	1	0 and 1
auditCarriageReturnReplacementValue	If the value of auditCarriage ReturnReplacement is 1, determines the characters with which to replace the carriage returns and line feeds.	null (this field is empty)	0 to 25 bytes

When the auditCarriageReturnReplacement value is set to 1, the auditCarriageReturnReplacementValue is examined and the value assigned to the auditCarriageReturnReplacementValue attribute is used as the delimiter between lines. If the auditCarriageReturnReplacementValue attribute is not assigned a value (but the auditCarriageReturnReplacement attribute value is set to 1) then the carriage return and line feed characters are replaced with a single blank character.

When the value of auditCarriageReturnReplacement is set to 0, then no replacements occur and the auditCarriageReturnReplacementValue is ignored.

Notes:

1. The value assigned to the auditCarriageReturnReplacementValue field is not interpreted but is used as it occurs. For example, if the auditCarriageReturnReplacementValue value is set to /n, then the two characters /n replace the carriage return and line feed characters; this does not cause a line return.
2. If the value assigned to the auditCarriageReturnReplacementValue attribute is longer than 25 characters, the string is truncated to the first 25 characters.

Chapter 5. Customizing the NetView Management Console Topology Console

This chapter describes basic customization of the NetView management console topology console as well as advanced customization that can be performed to change the look, feel and function.

Customizing Topology Console Functions

You can customize the NetView management console topology console by adding or changing any of the following items:

- Topology console icons
- Topology console backgrounds
- Topology console help facility
- Web browser for displaying views
- Double-click mouse actions for real or aggregate resources

Customization is performed at the topology server so that it automatically deploys to each topology console that subsequently signs on.

Adding and Customizing Topology Console Icons

To add a new icon to the topology console, create an icon in one of the following graphical interchange formats (GIFs):

32x32 pixels
This size is required. The file must be named 32_*xxxx*.gif (using only lowercase letters).

24x24 pixels
This size is optional and is used for the medium view sizes. If this size is not provided, the 32_*xxxx*.gif version is automatically scaled. The file must be named 24_*xxxx*.gif (using only lowercase letters).

16x16 pixels
This size is optional and is used for the smaller view sizes. If this size is not provided, the 32_*xxxx*.gif version is automatically scaled. The file must be named 16_*xxxx*.gif (using only lowercase letters).

Note: Animated GIFs are not supported.

Place the GIF files in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\db\current\icons`
- For UNIX: `$BINDIR/TDS/server/db/current/icons`

You can also change existing icons in these directories using any tool that enables GIF file manipulation. One example of a customized icon is the company icon. To customize this icon, use the Console Properties notebook. For details, see the icon customization procedures in the NetView management console online help.

Adding and Customizing Topology Console Backgrounds

To add a new background image to the topology console, create an image file following these guidelines:

- The name of the file must contain only lowercase letters.
- For a GIF image, use **gif** for the extension (or file type).
- For a JPEG image, use **jpg** for the extension (or file type).
- Use a size of 300x500 pixels.

Store the image file in the appropriate topology server directory:

- For Windows: `%BINDIR%\TDS\server\db\current\backgrounds`
- For UNIX: `$BINDIR/TDS/server/db/current/backgrounds`

You can also change the existing backgrounds in these directories using any tool that enables GIF or JPG manipulation.

Customized backgrounds are associated with a view. See the NetView management console online help for specific steps on how to customize a view background.

Note: Removing unused backgrounds reduces the install download time for the topology console.

Customizing Your Online Help Facility

As an alternative to using the built-in NetView management console help facility, you can specify your own Web browser to display HTML help pages. Select the **General** tab on the **Console Properties** notebook. In the group box labeled **Configure help facility**, select one of the following options:

- Use the built-in help facility.
- Use my default Web browser.
- Let me specify my own browser.

To specify your own Web browser, use lower case only, and specify the full drive and path name for the Web browser executable file. Click**Browse** to navigate to the directory and locate the executable file. If you specify $URL anywhere in the specified path, the Web address of the help is substituted for $URL automatically. If you do not specify a $URL, the Web address of the help is appended to the end of the given command path. If you use a blank in your path name, enclose the entire command in quotation marks.

When online help is selected from the menu, the corresponding HTML help pages are sent to your browser. An HTML message is displayed in either case (if problems occur either in launching the Web browser or after successfully launching the Web browser).

Note: If you request help at the **Sign On** dialog box, the help is displayed in the NetView management console help facility, regardless of your choice of help facility in this Console Properties settings page. At the time you sign on, the values specified on the console settings page are not yet available.

Configuring a Web Browser to Display Views

Designating a Console as a Web Server

When you first install the NetView management console, the Web server function is not enabled. Use the Web Server page in the Console Properties notebook to designate the console as a Web server. Click the **Help** button on the Web Server page for detailed descriptions of the page settings.

Designating Multiple Consoles as Web Servers

You can designate multiple consoles on the same system as Web servers, so that each console can set up its own set of views. Supply the data on the Web Server page in the Console Properties notebook to designate each console as a Web server, ensuring that you set the port number ranges so that each console on the same system can have a unique port number. Web servers on different systems can have the same port number.

Using the Web Browser

After designating the console system as a Web server, enter the fully qualified host name or IP address of the console system as a Web address in the Web browser. You do not need to add prefixes (http, www, and so on), although you might need to fully qualify the host name. If the port number for the Web server is not 80, follow the host name with a colon (:) and also the port number in the browser. The port number for the Web server is displayed on the Web Server page of the Console Properties notebook. If the port number is 80, the Web address looks like this:

```
Clientname
```

If the port number is not 80, the Web address looks like this:

```
Clientname:Port number
```

You cannot see a view in the Web browser until you add that view to the set of available views on the console. The view is displayed in the Web browser after the next Web browser refresh interval and the following information is displayed above the view:

- The number of available views
- The name of the current view
- The time and date that the view was captured

The refresh interval is displayed below the view.

Web browser views are not dynamic: they are *snapshots* in time of a view that is opened on the console system. If the status of a view changes, the change is shown in the browser after the next browser refresh interval. As views are captured for the Web server by the console system, they are added to a list of views. To capture the view, it must be visible on the console.

If your console is configured to use the Cycle Views function, the views on the console are automatically displayed for a certain time interval, enabling the views to be captured for the Web server automatically. To use the Cycle Views function to select the views and define the time interval to display them, from the Windows menu, select **Cycle Views**.

Topology views in the Web browser are the same size as the views captured on the console system. To change the size of a topology view in the Web browser, resize the view on the console and click **Add View to Web Server** or **Update View to Web Server** on the pop-up menu. The re-sized view is shown in the Web browser at the next refresh, or you can manually reload the view in the Web browser by clicking the **Reload** button.

A list of available views is provided in the browser window. Select a view, and click **Open**. As additional views are added to the set of available views on the console, they are added to the view list. Both Topology and Details views can be

displayed as they are displayed on the console. Icons in the Details view do not contain flags, background color, or an aggregate symbol.

You can manually refresh a view by right-clicking the view background on the console. From the pop-up menu, select one of the following actions:

- **Add View to Web Server**
- **Update View on Web Server**
- **Refresh Now** (if the view was added)

The view is captured and sent to the browser. Click **Reload** from the Web browser or wait for the next refresh interval to see the refreshed view. If you did not select **Make Views Available When They Are Opened** from the **Web Server** page, you must manually add views to the Web browser. To do this, right-click the view background and select **Add View To Web Server** from the pop-up menu. The view is shown in the list of views available to the Web browser at the next browser refresh interval.

Logging Web Server Messages

You can indicate on the Web Server page in the Console Properties notebook to create a record of each view that is opened on the web browser and send the record to the console log. Do this only if you need to closely track your views, since it creates a record every time a view is refreshed from the attached browsers.

Customizing Double-click Mouse Actions

You can customize the double-right-click and double-left-click mouse actions for real or aggregate resources using the Console Properties notebook. Open the notebook by clicking **Console Properties** on the Options menu item, and go to the Mouse page. For details about the page settings, click the **Help** button on the Mouse page.

Note: When you use the console in demonstration mode, the customized mouse actions are not in effect.

Advanced Topology Console Customization

Several advanced customization tasks can be used for the topology console. Most of these tasks can be performed without using the topology console interface.

Displaying Customized Help

You can create context menu help that displays a Hypertext Markup Language (HTML) document at the topology console using the following instructions.

1. Create your document file using basic HTML tags.

   Note: Use only lowercase characters in the file name and extension.
2. Place the help file into the appropriate directory on the topology server workstation:
   - For Windows: `%BINDIR%\TDS\server\db\current\help`
   - For UNIX: `$BINDIR/TDS/server/db/current/help`

   Note: You can also change existing help files which are located in these directories.
3. Add the new context command to the topology server using the command profile editor batch utility. Create the command with the following values:

Command string field: `com.tivoli.ihs.client.action.IhsShowDocument document_file_name`

Command exit field: `IHSXTJAM`

Add the command to the default profile. For an example of a command profile editor batch utility response file, see sample `ihsscp.xxx.rsp` (where *xxx* is a country code indicator such as en_US) in the sample topology server directory.

Enabling User Flags

Thirty-two flags are shipped with the NetView management console; eight of these can be customized. When they are shipped from the factory, the values for all of them are disabled. Therefore, if one of these flags is set for a resource, it does not display on any NetView management console user interface component. To display these flags, they must first be enabled. After a customized flag is enabled, it displays on the appropriate NetView management console user interface component (such as, in the Resource Properties window, or the Filter Bar).

To enable any of the flags that can be customized, update the default operational scheme by performing the following steps from the server workstation:

1. Make a backup copy of the NetView management console default operational scheme files with one of the following methods:
   - Use a packaging tool such as PKZip, WinZip, or tar.
   - Create a backup directory and copy the default scheme files to it.
2. Determine the hexadecimal value of the specific user flag that you want to enable by using the following steps:
   a. Open the defaultscheme.properties file in a text editor.
   b. Search for the string `f1` to locate the section of the file where the flags are defined.
   c. Scroll down until you find the comment for the flag you want to enable.

      For example:

      ```
      * User 1
      f25.value =0x00000080
      ```

      The corresponding value of the attribute is the hexadecimal value for the flag. In this example, `0x00000080` is the `User 1 flag` value.
3. In the defaultscheme.properties file, search for the string `FLAG values` to locate the section of the file where the flags are defined. The flags are defined in descending order.
4. Scroll down until you see the value for the user flag that you want to enable.
5. To enable this flag, change the value of the `defDefine` attribute from `false` to `true`.

   Depending on how you want this flag to work, you might need to define additional attributes. All flag attributes are documented by the `com.tivoli.ihs.client.view.IhsUserStatus` class as shown in the following example.

   ```
   #  "Flag" Definition Values:
   #
   #    fX.tag         Reference tag (required).
   #    fX.defDefine   Is this item defined? (optional, true).
   #                   Set to false to disable this item.
   #    fX.isDefault   Does this item contain default values for all other items?
   #                   (optional, false).
   #    fX.weight      Orders an item relative to other items by "weight"
   #                   (optional, 100).
   ```

```
#    fX.value       Status value (required, only 1 bit can be on).
#    fX.defFilter   Currently not used.
#    fX.defDisplay  Currently not used.
#
#    fX.onView      Display sub-icon on topology view? (optional, false)
#    fX.color       Color of sub-icon on topology view.  Flag with highest
#                   weight is used. (optional, gray)
#    fX.reqAdmin    Administrator required to set/clear (optional, false).
#    fX.canSet      Is this flag allowed to be set (turned on)? (optional,
#                   true)
#    fX.canClear    Is this flag allowed to be cleared (turned off)? (optional,
#                   true)
#    fX.forAgg      Applicable for an aggregate resource? (optional, false)
#    fX.forReal     Applicable for a real resource? (optional, true)
#    fX.relatedTo   Mask of "related" flags (optional, none).
#
```

For more information about self-documenting data classes for the NetView management console topology console, see “Running a Console Class” on page 35.

6. Save your changes.
7. To define the wording for this user flag, open the defaultschemetext.properties file in a text editor.
8. Search for the `f.`*xxxxxxx* string where *xxxxxxx* is the hexadecimal value of the specific user flag you are enabling.
9. Change the value of the `f.`*xxxxxxx*`.label` attribute to include all text for this flag.
10. Change the value of the `f.`*xxxxxxx*`.abbrev` attribute to the abbreviation you are assigning to this flag.
11. Save the changes.

To verify that the flag you have enabled exists, perform the following steps:

1. Start an NetView management console topology console.
2. Open a view.
3. Select a resource and right-click to display the context menu.
4. Click **Resource Properties**.
5. Verify that the user flag is present on the **Resource** window.
6. Open the **Console Properties** notebook, select the **Status** page, and verify that the flag is present.

The example in Table 8 shows the changes that enable the `User 1 flag` and name it `Retired`.

Table 8. Changing the User 1 Flag to Retired

File	Before	After
defaultscheme.properties	* User 1 f25.value =0*x*00000080 f25.weight =2000 f25.defDefine =false	* User 1 f25.value = 0*x*00000080 f25.weight = 2000 f25.defDefine = true
defaultschemetext.properties	f.00000080.label = User 1 f.00000080.abbrev = User1	f.00000080.label =Retired f.00000080.abbrev =Retd

Adding a Flag to the Context Menu

The following flags can be directly manipulated from resource specific context menus:

- Suspend, Manually Clear
- Suspend, Automatically Clear
- Clear Suspended
- Clear Child Suspended

You can enable context menu items for additional flags. From the server workstation, add a new flag context menu item to the NetView management console default operational scheme. The properties files, in which the NetView management console operational scheme is defined, are described briefly in Table 9.

Table 9. NetView Management Console Properties Files

Scheme File Name	Description
defaultscheme.properties	Provides most of the operational definitions (except required multicultural support enabled text). 1. COLOR Definitions (Not currently used) 2. FLAG Definitions 3. FLAG (User status) menu items 4. STATUS SCHEME values 5. STATUS values 6. STATUS MAPPING values (Not currently used) 7. MODE values (NetView management console only supports Control mode) 8. LAYER values 9. Miscellaneous Control Values
defaultschemetext.properties	Provides the English language set of multicultural support enabled text that is required by the `defaultscheme.properties` file. *(Status text and colors not currently used.)*
defaultschemetext_ja.properties	Provides the Japanese language set of multicultural support enabled text that is required by `defaultscheme.properties` file. *(Colors not currently used.)*

The disk location of the scheme properties files is determined by the NetView management console operational mode (as shown in Table 10).

Table 10. Location of Scheme Files

NetView Management Console Operational Mode	Location of Scheme Files
Signed on to a server	<installRoot>\bin\<*interp*>\TDS\Server\db\current\settings
Demonstration	<installRoot>\bin\generic_unix\TDS\client\settings

To add a new flag context menu item to the NetView management console default operational scheme, perform the following steps:

1. Make a backup copy of the NetView management console default operational scheme with one of the following methods:
   - Use a packaging tool such as PKZip, WinZip, or tar.
   - Create a backup directory and copy the default scheme files to it.
2. Determine the hexadecimal value of the specific user flag that you want to enable by performing the following steps.
   a. Open the `defaultscheme.properties` file in a text editor.
   b. Search for the string `f1` to locate the section of the file where the flags are defined.
   c. Scroll down until you find the comment for the flag.

      For example:

      ```
      * Marked
      f1.value =0x80000000
      ```

      The corresponding value for the attribute is the hexadecimal value for the flag. In this example, `0x80000000` is the `Marked flag` value.
3. In the defaultscheme.properties file, search for the string `User status menu items` to locate the section of the file where the context menu items for the flag are defined.
4. The menu items are defined in the order they are displayed in a context menu, so scroll down to the bottom of the list of menu items.
5. Create a new set of menu item attributes with a `menu`*x* prefix (where *x* is the next sequence number). For example:

   ```
   menux.tag   = <unique identifier for this menu item>
   menux.value = <hex value of the flag that you want to enable>
   menux.setTo = <true=set flag -or- false=Clear flag>
   ```
6. Depending on how you want this context menu item to work, you might also have to define additional attributes.

   The `com.tivoli.ihs.client.view.IhsFlagMenuScheme` class documents all menu item attributes, as shown in the following example.

   ```
   #  "User Status Menu Item" Definition Values:
   #
   #  defaultscheme.properties:
   #    menuX.tag         Reference tag (required).
   #    menuX.defDefine   Is this item defined? (optional, true).
   #                      Set to false to disable this item.
   #    menuX.isDefault   Does this item contain default values for all other
   #                      items?
   #                      (optional, false).
   #    menuX.weight      Orders an item relative to other items by "weight"
   #                      (optional, 100).
   #    menuX.value       Associated flag value (required, only 1 bit can be on).
   #    menuX.setTo       When selected, set flag on (true) or off (false)
   #                      (required).
   #    menuX.setRel      If this flag is set, should a related flag also be set?
   #                      (optional, false).
   #                      See flag definitions relatedTo setting for the related
   #                      flag.
   #    menuX.setRelTo    If the related flag is to be set, set it on (true) or
   #                      off (false) (optional).
   #  defaultschemetext.properties:
   #    menuX.text        Menu text (required)
   #    menuX.help        Menu help file (required)
   #
   ```

   For more information about self-documenting data classes for the NetView management console topology console, see "Running a Console Class" on page 35.

7. Save the changes.
8. Open the `defaultschemetext.properties` file in a text editor and search for the string `User status menu items text` to locate the section of the file where the flag menu items are defined.
9. Scroll down to the end of the list of menu items which are currently defined.
10. Create a new attribute with prefix that is the same as the `menu`*x* value you used in the defaultscheme.properties file.

    For example:

    ```
    menux.text= <text displayed for this menu item>
    ```
11. Save the changes.

To verify these changes, perform the following steps:

1. Start an NetView management console topology console.
2. Verify that the new context menu is displayed properly:
   a. Open a view.
   b. Locate a resource for which your new menu item should be available.
   c. Right click on that resource and verify that your new menu item is present.
   d. Select the menu item and verify that the flag was updated.
3. Verify that the definition of the default note was updated:
   a. Open the **Console Properties** notebook and select the **Notes** tab.
   b. Verify that the new menu item is present.

For Resource Object Data Manager (RODM)-based resources, the NetView management console flags are the 4-byte `UserStatus` fields. For details, see the *IBM Tivoli NetView for z/OS Data Model Reference*.

The example in Table 11 on page 34 shows the changes made to enable context menu items for `Marked`*`flag`* in the defaultscheme.properties file.

Table 11. The Markedflag Item in the defaultscheme.properties File

Before	After
**********	**********
* User status menu items	* User status menu items
**********	**********
menu1.setRelTo = false	menu1.setRel = true
menu2.tag = SuspAut	menu1.setRelTo = false
menu2.value = 0*x*40000000	menu2.tag = SuspAut
menu2.setTo = true	menu2.value = 0*x*40000000
menu2.setRel = true	menu2.setTo = true
menu2.setRelTo = true	menu2.setRel = true
menu3.tag = ClearSusp	menu2.setRelTo = true
menu3.value = 0*x*40000000	menu3.tag = ClearSusp
menu3.setTo = false	menu3.value = 0*x*40000000
menu3.setRel = true	menu3.setTo = false
menu3.setRelTo = false	menu3.setRel = true
menu4.tag = ClearChildSusp	menu3.setRelTo = false
menu4.value = 0*x*00800000	menu4.tag = ClearChildSusp
menu4.setTo = false	menu4.value = 0*x*00800000
	menu4.setTo = false
	menu5.tag = SetMarked
	menu5.value = 0*x*80000000
	menu5.setTo = true
	menu6.tag = ClearMarked
	menu6.value = 0*x*80000000
	menu6.setTo = false

The example in Table 12 on page 35 shows the changes that were made to enable context menu items for the `Marked`*`flag`* in the defaultschemetext.properties file.

Table 12. The Markedflag Item in the defaultschemetext.properties File

Before	After
**************************** * User status menu items text **************************** menu1.text = Suspend, Manually Clear menu1.help = ihs_mi_clermansusp_*xxx*.html menu2.text = Suspend, Automatically Clear menu2.help = ihs_mi_clerautosusp_*xxx*.html menu3.text = Clear Suspended menu3.help = ihs_mi_clersuspended_*xxx*.html menu4.text = Clear Child Suspended menu4.help = ihs_mi_clerchldsusp_*xxx*.html	****************************** * User status menu items text ****************************** menu1.text = Suspend, Manually Clear menu1.help = ihs_mi_clermansusp_*xxx*.html menu2.text = Suspend, Automatically Clear menu2.help = ihs_mi_clerautosusp_*xxx*.html menu3.text = Clear Suspended menu3.help = ihs_mi_clersuspended_*xxx*.html menu4.text = Clear Child Suspended menu4.help = ihs_mi_clerchldsusp_*xxx*.html menu5.text = Set Marked menu6.text = Clear Marked

Running a Console Class

Many of the NetView management console topology console data classes are self-documenting. This means that if you run the data class, it tells you all of the attributes that you can define in a properties file to create an instance of the class. All of the NetView management console topology console classes are contained in the `\bin\generic_unix\tds\client\lib\ihseuc.jar` file. To obtain information for a specific data class, issue the following command from a console workstation:

```
java -classpath %classpath%;<installRoot>\bin\generic_unix\tds\client\lib\ihseuc.jar
<package qualified class name>
```

This command generates all of the attributes for a flag, for example:

```
java -classpath %classpath%;<installRoot>\bin\generic_unix\tds\client\lib\ihseuc.jar
com.tivoli.ihs.client.view.IhsUserStatus
```

Customizing Web Server Enablement

You can enable or disable the Web server function (as a tab on the Console Properties notebook). Table 13 shows the values for enabling or disabling the Web browser function; these values are set in the defaultscheme.properties file.

Table 13. Enabling and Disabling the Web Server Function in the defaultscheme.properties File

defaultscheme.properties Attribute	**Default Value**	**Other Valid Value**
webServerPage.enable	true	false

To customize the Web server enablement, the NetView management console operational scheme must be updated, as shown in Table 13. To customize Web server enablement, perform the following steps.

1. Make a backup copy of the default NetView management console operational scheme.
2. Open the `defaultscheme.properties` file in a text editor.
3. Search for the string `webServerPage`.
4. Change the value on this line to false if you want to disable the Web Server function.
5. Save the changes.

To verify this change, perform the following steps:

1. Start an NetView management console topology console
2. Open the Console Properties notebook.
3. Verify that the Web Server tab is no longer displayed on the Console Properties notebook page.

Note: Setting the attribute in the defaultscheme.properties file overrides the automatic start setting. For example, if you previously set the Web Server to automatically start and then set `webServerPage.enable=false`, then the Web Server does not automatically start.

For more information about the NetView management console operational scheme, see Table 9 on page 31.

Customizing the View Bar Layout

You can adjust the number of rows or columns of view buttons displayed on the View Bar of the main NetView management console window. The minimum size of the view buttons can also be adjusted. Table 14 shows the attributes used to adjust these settings. These attributes are set in the defaultscheme.properties file.

Note: Either the number of rows *or* the number of columns can be customized, but not both. The value for one of these settings must remain 0. The 0 value indicates that no maximum value exists for the attribute.

Table 14. Changing View Bar Attributes and Values in the defaultscheme.properties File

Attribute	Description	Default Value	Minimum Value	Maximum Value
`viewbar_max_rows = <int>`	Maximum number of rows in the view bar grid. New columns are added to display additional views.	0		None
`viewbar_ max_columns = <int>`	Maximum number of columns in the view bar grid. New rows are added to display additional views.	0		None
`viewbar_min_view_size = <int>`	Minimum size to make a view when sizing the views to fit on the view bar.	100	100	

To to change the view buttons on the View Bar, update the NetView management console operational scheme from the console or server workstation, depending on the mode in which you are running. See Table 10 on page 31 to determine if the

updates need to be performed from the server or console workstation. To customize the view bar layout, perform the following steps:

1. Make a backup copy of the default NetView management console operational scheme.
2. Open the defaultscheme.properties file in a text editor.
3. Search for the string `viewbar` and locate the attribute to be changed.
4. Change the value of the attribute.
5. Save the changes.

The example in Table 15 shows the changes made to the maximum number of rows in the View Bar grid from 0 to 5.

Table 15. Changing the Number of Rows in a View Bar in the defaultscheme.properties File

Before	After
`viewbar_max_rows  = 0`	`viewbar_max_rows  = 5`

For more information about the NetView management console operational scheme, see Table 9 on page 31.

Customizing the Automatic Download of Files At Log On

When the console logs on to the server, the time stamp of the files (as stored on the console) and the time stamp of the files on the server are compared. If the console consistently logs on to the same server, these time stamps are the same and the files are not downloaded. See Appendix F, "Automatic File Download at Console Log On," on page 197 for more information. If the console logs on to servers on different platforms, however, then these time stamps are slightly different. Use the defaultscheme.properties file to set the tolerance level of the time difference between time stamps. The tolerance level is a value, in minutes, between the times on the time stamps. If it is set to 0 (zero), for example, then the files are automatically downloaded from the server unless the server and console time stamps match exactly. Table 16 shows the attribute and the range of valid values for this attribute.

Table 16. Changing the Tolerance for Time Stamp Differences in the defaultscheme.properties File

`defaultscheme.properties` attribute	**Default Value**	**Minimum Value**	**Maximum Value**
`file_download_timestamp_tolerance = int`	720 (12 hours)	0	43,200 (30 days)

The example in Table 17 shows two scenarios in which the time stamp tolerance can be used and the associated values.

Table 17. Time Stamp Tolerance Settings in the defaultscheme.properties File

Scenario	Value to Use
The console is logging on to the same server and you want to automatically download any files that have changed each time you log on.	Set the value to 0 (zero) in the following way: `file_download_timestamp_tolerance = 0`

Table 17. Time Stamp Tolerance Settings in the defaultscheme.properties File (continued)

Scenario	Value to Use
You are logging on to multiple servers on various platforms and you do not want the files to download each time you log on	Perform one of the following actions: • Leave the default value of 12 hours: `file_download_timestamp_tolerance = 720` • Set the value higher, 3 days in this example: `file_download_timestamp_tolerance = 4320` • Set the value lower, 6 hours in this example: `file_download_timestamp_tolerance = 360`

Overriding the Default Date and Time Format

By default, the Java™ run time obtains display formats for the date and time from the operating system. On Windows systems, for example, the date and time display formats are defined by selecting **Control Panel**, then **Regional Settings**. Unfortunately, certain changes, such as the time format, that are made to these values are not propagated to Java 2 Runtime Environment, Standard Edition (J2RE). If you change the `Time` format in **Regional Settings** in Windows, to display in 24-hour format, this change is not propagated to the J2RE. The topology console, therefore, seems to be out of sync with the rest of the local applications.

You can force the console to override the use of the operating system formats and display the dates and times you want. Table 18 shows the attributes (in the defaultschemetext.properties file) used to make this change.

Table 18. Changing the Date and Time Format in the defaultschemetext.properties File

Attributes	Description
`override.date = <format>`	Format used for formatting a date
`override.time = <format>`	Format used for formatting a time
`override.datetime = <format>`	Format used for formatting a composite date and time

The example in Table 19 shows how a time stamp of 6:45:07 p.m. on March 30, 2001 displays for various format specifications.

Table 19. Example of Time Stamp Format Specifications.

Format Specification	Resulting Display Text
yyyy.MM.dd	2001.03.30
yyyy.MMM.dd	2001.Mar.30
yyyy.MMMM.dd	2001.March.30
hh:mm:ss a	06:45:07 PM
hh:mm:ssa z	06:45:07PM EST
HH:mm:ss z	18:45:07 EST

To customize the date and time format, update the default operational scheme from the console or the server workstation, depending on the mode in which you are running. See Table 10 on page 31 to determine if the updates need to be performed from the server or console workstation. To customize the date/time format, perform the following steps.

1. Make a backup copy of the default NetView management console operational scheme.
2. Open the defaultschemetext.properties file in a text editor.
3. Search for the string `override.date`.
4. Uncomment the override item by removing the leading * from the beginning of the line.
5. Change the value on the `<value>` line to the format specification you want.

   The `<value>` field defines the format of the displayed item. It can contain both literal and substitution symbols.
6. Save the changes.

To verify the changes, perform the following steps:

1. Start an NetView management console topology console.
2. Open a view.
3. Verify the `override.datetime` attribute change by checking the view information area in the status bar.
4. Verify the `override.date` and `override.time` attribute changes by placing the view in *details* mode and checking the Time/Date column.

The example in Table 20 shows the changes made to override all of the date and time display formats. These changes were made in the defaultschemetext.properties file.

Table 20. Overriding the Date and Time Display Formats in the defaultschemetext.properties File

Before	After
*************************	*************************
* Date/Time Overrides Formats	* Date/Time Overrides Formats
*************************	*************************
* override.date =yyyy.MMM.dd	override.date =yyyy.MMM.dd
* override.time =HH:mm:ss z	override.time =HH:mm:ss z
* override.datetime=yyyy.MMM.dd @ HH:mm:ss z	override.datetime=yyyy.MMM.dd @ HH:mm:ss z

For more information about the NetView management console operational scheme, see Table 9 on page 31.

Time Format Syntax

Table 21 is an excerpt from the javadoc of the `java.text.SimpleDateFormat` class. It provides details about all of the possible formatting capabilities. To specify the time format, use a time pattern string. In this pattern, all ASCII letters are reserved as pattern letters, which are defined in Table 21.

Table 21. Time Format Syntax

Symbol	Meaning	Presentation	Example
G	era designator	(Text)	AD
y	year	(Number)	2001
M	month in year	(Text & Number)	April & 01
d	day in month	(Number)	10

Table 21. Time Format Syntax (continued)

Symbol	Meaning	Presentation	Example
h	hour in a.m./p.m. (1-12)	(Number)	12
H	hour in day (0-23)	(Number)	0
m	minute in hour	(Number)	30
s	second in minute	(Number)	55
S	millisecond	(Number)	978
E	day in week	(Text)	Tuesday
D	day in year	(Number)	189
F	day of week in month	(Number)	2 (2nd Wed in July)
w	week in year	(Number)	27
W	week in month	(Number)	2
a	a.m./p.m. marker	(Text)	PM
k	hour in day (1-24)	(Number)	24
K	hour in a.m./p.m. (0-11)	(Number)	0
z	time zone	(Text)	Pacific Standard Time
’	escape for text	(Delimiter)	
’ ’	single quotation mark	(Literal)	’

The count of pattern letters determines the format in the following way:

- Text: If the presentation is in text and 4 or more pattern letters exist, then use the full form. If fewer than 4 pattern letters exist, then use the short or abbreviated form, if one exists.
- Numeric: If the presentation is numeric, then the field contains the minimum number of digits. Shorter numbers are zero-padded to this amount.

  Note: The year is handled differently. If the count of contents of the y field is 2, then the year is truncated to 2 digits.
- Text and numeric: If the presentation contains 3 or more bytes, use text, otherwise use numerics.

Any characters in the pattern that are not in the ranges of ’a’ - ’z’ and ’A’ - ’Z’] are treated as quoted text. For example, the following characters are displayed in the resulting time text (even if they are not within single quotation marks):

- ’:’
- ’.’
- ’ ’
- ’#’
- ’@’

Patterns containing a pattern letter that is not valid result in a thrown exception during formatting or parsing.

The examples in Table 22 use a US Location.

Table 22. Examples of Time and Date Formatting.

Format Pattern	Result
`"yyyy.MM.dd G 'at' hh:mm:ss z"`	2001.04.01 AD at 15:08:56 PDT

Table 22. Examples of Time and Date Formatting. (continued)

Format Pattern	Result
`"EEE, MMM d, ''yy"`	Sun, April 01, '01
`"h:mm a"`	12:08 PM
`"hh 'o''clock' a, zzzz"`	12 o'clock PM, Pacific Daylight Time
`"K:mm a, z"`	0:00 PM, PST
`"yyyyy.MMMMM.dd GGG hh:mm aaa"`	2001.April.01 AD 12:08 PM

Customizing Data1, Data2, and Data3 Field Labels for RODM Resources

You can customize the labels for the Data1, Data2, and Data3 fields for RODM resources, both real and aggregate. The labels are defined in the defaultschemetext.properties file and the defaults are shown in Table 23.

Table 23. Default Text for Optional Data Fields for RODM Resources

defaultschemetext.properties Attribute	Default Text
Data1.RODM.Resource	Other Data
Data2.RODM.Resource	User Data
Data3.RODM.Resource	IP Address

The updates are made from the topology server or the topology console, depending on the mode in which you are running; to determine where to make the updates, see Table 10 on page 31. To customize the labels, follow these steps:

1. Make a backup copy of the default NetView management console operational scheme.
2. Open the defaultschemetext.properties file in a text editor.
3. For each attribute you want to change (Data1.RODM.Resource, Data2.RODM.Resource, or Data3.RODM.Resource), find the attribute and change the attribute value to the text you want to use.
4. Save your changes.

Verify your changes in the following way:

1. Start the topology console.
2. Open a view.
3. Right-click a RODM resource, and click **Resource Properties**.
4. Verify that the label text you defined is shown.

Customizing Line Thickness

You can adjust the thickness of lines in a topology view. Table 24 shows the values for line thickness; these values are set in the defaultscheme.properties file.

Table 24. Changing Line Thickness (in Pixels)

`defaultscheme.properties` Attribute	Default Value (in Pixels)	Minimum Value (in Pixels)	Maximum Value (in Pixels)
line_thickness	2	1	4

To customize the line thickness, the NetView management console operational scheme must be updated, as shown in Table 24 on page 41. This can be done from the console or the server workstation, depending on the mode in which you are running. See Table 10 on page 31 to determine if the updates need to be performed from the server or console workstation. To customize line thickness, perform the following steps.

1. Make a backup copy of the default NetView management console operational scheme.
2. Open the `defaultscheme.properties` file in a text editor.
3. Search for the string `line_thickness`.
4. Change the value on this line to the line thickness (in pixels). The valid range is from 1 to 4.
5. Save the changes.

To verify this change, perform the following steps:

1. Start an NetView management console topology console.
2. Open a view.
3. Verify the line thickness while in the topology view.

The example in Table 25 shows the changes made to the line thickness from two pixels to one pixel.

Table 25. Changing the Line Thickness in the defaultscheme.properties File

Before	After
**********	**********
* Thickness of connection lines in pixels	* Thickness of connection lines in pixels
**********	**********
line_thickness = 2	line_thickness = 1

For more information about the NetView management console operational scheme, see Table 9 on page 31.

Chapter 6. Creating a Demonstration

You can create your own new NetView management console demonstration, customized to look and feel like your NetView management console environment and display views from your network environment. This section provides the process to create your own demonstration.

The demonstration capability is installed as part of the *NetView Management Console Productivity Kit* (on the Windows platform, only). Be sure to perform a custom installation of the NetView management console topology console and select the *NetView Management Console Productivity Kit*. To access further information about the demonstration function, in the Windows environment, double-click the **NMC Productivity Kit** icon on the desktop.

You can capture views from a live NetView management console system and integrate them into your demonstration. Do this to demonstrate NetView management console in your own environment to provide operator training, for example.

The following sections contain the steps that are required to create a new demonstration.

1. "Capturing Live Views from your NetView Management Console System."
2. "Integrating Captured Views into the Demonstration" on page 50 by:
   a. "Updating the Business Tree" on page 50.
   b. "Renaming Navigation Views" on page 52.
3. "Defining New Resource Types in Saved Views" on page 53.

After you complete these steps, the files shown in Table 26 comprise the new demonstration.

Table 26. Demonstration Files

Demonstration	Directory	Files
View files (basic data files)	<installRoot>\bin\generic_unix\TDS\client\views	*.md
Basic data definitions (See "Using Basic Data Files" on page 45 to define additional resource types).	<installRoot>\bin\generic_unix\TDS\client\settings	basicdata.nmc

Capturing Live Views from your NetView Management Console System

To capture views from a live NetView management console system for subsequent integration into the demonstration, use the `-saveViewsLocally` command-line argument. This argument instructs the `Save View Customization` code to save the view in a local file in the <installRoot>\bin\generic_unix\TDS\client\views directory instead of within the topology server database.

Note: For more information about local files, see "Renaming Navigation Views" on page 52.

To capture a view from the client workstation and to save all the views that you want to be displayed in your demonstration, perform the following steps:

1. From the command prompt, make a backup copy of the demonstration views shipped with NetView management console in the following way:
   a. `cd <installRoot>\bin\generic_unix\TDS\client\views`
   b. `md backup`
   c. `copy * backup`
2. From the command prompt, delete all of the shipped NetView management console demonstration files in the following way:
   a. `cd <installRoot>\bin\generic_unix\TDS\client\views`
   b. `del *`
3. Start the NetView management console topology console by using the `-saveViewsLocally` command-line argument. To do this, change the properties of the NMC Topology Console desktop icon, in the following way:
   a. Right-click on the **NMC Topology Console** desktop icon to display the context menu.
   b. Click **Properties**.
   c. From the notebook that opens, select the **Shortcut** tab.
   d. Select the **Target** field and place the cursor after the last character.
   e. Type a separating blank and then enter **-saveViewsLocally**.
   f. Click **OK**.
4. Start the NetView management console topology console and sign on.
5. Save the business tree view in the following way:
   a. Right click on the background of the business tree.
   b. Select **Save View Customization**.

      This creates the <installRoot>\bin\generic_unix\TDS\client\views\1.ctl.md file. The root node of the business tree is always `resource ID 1`.

      Note: For more information about resource Ids, see "Finding the Resource ID" on page 54.
   c. Save all of the specific views to be included in your demonstration by opening each view individually. To open each view, perform one of the following actions:
      - From the business tree, open each view sequentially.
      - Select **Locate Resource**.
      - Select **More Detail**, **Configuration**, or **Locate Failing Resource** from an open view.
   d. To save each view:
      1) Right click on the view background.
      2) Select **Save View Customization**.
   e. Correlate the type of view saved with the saved name by performing the following steps:
      1) Write down information such as the view type, view name, Configuration Parents, More Detail, and Locate Resource.
      2) Look in the <installRoot> \bin\generic_unix\TDS\client\views\ directory for the view that you just saved. It has the format

<resourceid>.ctl.md. Write down this file name next to the view type. You need this information when you rename the view to conform to the demonstration naming conventions later.

The view you just created is the newest view in the directory. If you kept track of the demonstration views as you created them, you should be able to distinguish which view was just created to associate it with the view type.

Note: Multiple navigation views can be created from the same resource.

A fixed file name is created for each view saved based on the ID of the resource, regardless of how the view is opened. If you save a More Detail and a Configuration view initiated from the same resource, the same view file name is used. The last view saved is the available view. If you want to have multiple navigation views available from a single resource, you *must* use these steps:

a) Select the Nth navigation view.

b) Save the view.

c) Rename the saved view.

d) Repeat these steps for the next navigation view.

6. Integrate the captured views into the demonstration in the following way.
   a. Update the business tree. For detailed information about performing this step, see "Updating the Business Tree" on page 50.
   b. Rename the navigation views to conform to the required demonstration naming conventions. For detailed information about performing this step, see "Renaming Navigation Views" on page 52.
7. Define any resource types in the saved views that were not previously defined.

Using Basic Data Files

Because Resource Object Data Manager (RODM), the topology server, and the CPE command definitions are not available when the NetView management console topology console is run in demonstration mode, the basic data properties files are used to define:

- Resource types
- Individual context menu items
- Sets of context menu items
- A set of view menu items (resource independent)
- A default set of real (resource dependent) resource menu items
- A default set of aggregate (resource dependent) resource menu items

When the console starts, the following files are loaded from the <installRoot>\bin\generic_unix\TDS\client\settings directory:

- basicdata[<multicultural_support_language>] (base definitions of resources and common menu items)
- basicdata[<multicultural_support_language>].nmc (additional NetView management console resource types and menu items)

Before making any changes to these files, make backup copies of these files with one of the following methods:

- Use a packaging tool such as PKZip, WinZip, or tar.
- Create a backup directory and copy the installed files into it.

The following Japanese files can also be customized, using double byte character set (DBCS) characters, with conversion commands:

- basicdata_ja
- basicdata_ja.nmc
- defaultschemetext_ja.properties

To customize these files, first convert them with the following command prior to editing:

```
native2ascii -reverse <input_dir>\filename <outpug_dir>\filename
```

After editing, perform a second conversion with the following command:

```
native2ascii -encoding SJIS <input_dir>\filename <outpug_dir>\filename
```

Defining a Menu Item

Table 27 gives a description of the attributes available for menu item definition where the `N` in the attribute is a consecutive sequence number (1, 2, ..., n) within the group of menu items. Several types of menu items are supported. Each type has a separate definition list in the basic data file and different definition requirements, as shown in Table 28.

Table 27. Description of menu item attributes

Attribute	Purpose
mi.N.<attr>	Java applications launched using `com.tivoli.ihs.client.action.IhsJavaAppAction`. **Note:** Use this type of attribute for testing Java applications.
sc.N.<attr>	Predefined scenarios.
nop.N.<attr>	NOP menus which are displayed in the context menu but do not do anything when selected.

Table 28. Attributes that can be specified for a menu item.

<attr>	Description	mi.N.<attr>	Sc.N.<attr>	nop.N.<attr>
tag	*Tag* name that is used to reference this menu item	Required	Required	Required
label	Menu item *text*	Required	Required	Required
class	Package-qualified Java class that provides the Java implementation of this menu item	Optional Def: IhsJavaAppAction[1]	Ignored Def: IhsScenario[1]	Ignored Def: IhsCPEAction[1]
data	Class specific data	Optional Def: blank	Required scenario file name and arguments	Ignored
html	**Menu Help** HTML panel name for the menu item	Optional Def: ihs_nohelp_XXX.html	Optional Def: ihs_nohelp_XXX.html	Optional Def: ihs_nohelp_XXX.html

Table 28. Attributes that can be specified for a menu item. (continued)

<attr>	Description	mi.N.<attr>	Sc.N.<attr>	nop.N.<attr>
max	Maximum number of selected resources supported by a particular resource type	Optional Def: 1	Optional Def: 1	Optional Def: 1
debug	Determines if this is a debug menu item; enabled when the -debug command-line argument is used	Optional Def: false	Optional Def: false	Optional Def: false
vfy	Determines if a verification prompt is needed before the item is launched[2]	Optional Def: false	Optional Def: false	Optional Def: false

Notes:

1. These classes are part of the com.tivoli.ihs.client.action package.
2. This attribute, <attr>, is not currently implemented.

Defining a Set of Menus

A set of menus is a named group of one of the following items:

- Menu items
- Separators
- Other sets

A set can be used to define either of the following items:

- A set of menu items that can be referenced as many times as needed
- A cascaded menu item

Conditional debug sets are enabled when the -debug command-line argument is used.

A set is defined using the following syntax:

```
set.N.<attr>=value
set.N.X     =tag reference
```

where:

- The set stem uniquely identifies this as a set definition.
- N is a consecutive sequence number (1, 2, ..., n) within the set definition area.
- X is a consecutive sequence number (1, 2, ..., n) within a set.

The attributes in Table 29 can be specified for a set.

Table 29. Attributes that can be specified for a set.

<attr>	Description	Default
tag	Name used to reference this set	Required

Table 29. Attributes that can be specified for a set. (continued)

<attr>	Description	Default
label	Name of cascaded menu item	Optional • If omitted, the set is used as a container; items are copied into the current menu. • If specified, a cascaded menu item is created with the menu items for this set.
debug	Determines if this is a debug set; enabled when the -debug command-line argument is used	Optional Def: false

Each item of a set references one of the items in Table 30.

Table 30. Set reference items.

Tag Reference	Description
- (dash)	Menu separator
`tag` name	Reference to a: • menu item • set

A Sample from basicdata.nmc: The following example is from the basicdata.nmc file.

```
################################################
# Context menu items for an AGGREGATE resource
################################################
set.3.tag   =@nmcAgg
set.3.1     =@baseAgg
set.3.2     =-
set.3.3     =nmcFailing
################################################
# "Configuration" cascaded menu
################################################
set.4.tag   =@nmcConfig
set.4.label =Configuration
set.4.html  =ihs_mi_config_XXX.html
set.4.1     =nmcCfgPar
set.4.2     =nmcCfgChild
set.4.3     =nmcCfgPeer
set.4.4     =nmcCfgLP
set.4.5     =nmcCfgL
set.4.6     =nmcCfgP
set.4.7     =nmcCfgBBone
################################################
# SET for all example Java applications
################################################
set.5.tag   =@exJavaApps
set.5.debug =true
set.5.1     =-
set.5.2     =exGUI
set.5.3     =exCmdRsp
set.5.4     =exIPL
```

Defining a Resource Type

A resource type is defined using syntax as shown in Table 31.

Table 31. Attributes that can be specified for a resource type.

<attr>	Description	Default
rtX_name	Name; specified in node/link ″resource type″ field	
rtX_desc	Description text	Optional resource type name
rtX_image	Image file name	Optional node.gif
rtX_geometric	Geometric shape name	Optional com.tivoli.ihs.reuse.gui.IhsRectangle
rtX_help	Help URL	Optional none
rtX_flags	Flags	Optional 1

The following example is from the basicdata.nmc file:

```
rt34_name=268828673
rt34_desc=LU
rt34_image=duiu5n00.gif
rt34_geometric=com.tivoli.ihs.reuse.gui.IhsTrapezoid
```

You can also specify additional context menu items for specific resource types. In the following example, each `rtN_mi.X` item is a tag reference.

```
rt6_name=Lotus Notes Client
rt6_image=notesc.gif
rt6_geometric=com.tivoli.ihs.reuse.gui.IhsPentagon
rt6_mi.1=-
rt6_mi.2=start
rt6_mi.3=stop
rt6_mi.4=busSysHelp
rt6_mi.5=instSmtpGW
```

The current syntax for defining a resource type is available by using the following command when `CLASSPATH` includes the ihseuc.jar file:

```
java com.tivoli.ihs.client.view.IhsResourceType
```

Defining Defaults

Each basic data file defines defaults for the following items:

- View context menu items
- Common context menu items for each real resource
- Common context menu items for each aggregate resource

These defaults are shown in the following example:

```
#########################################################
# Default VIEW items
#
# - each SET reference is a "tag"
```

```
# - these values override those in "basicdata"
#########################################################
view.addDefault=true            <-- controls if defaultSet is added
view.defaultSet=@nmcView        <-- set tag reference

#########################################################
# Default Resource Type items for this "basic data" file
#
# - each SET reference is a "tag"
# - these values override those in "basicdata"
#########################################################
agg.addDefault =true            <-- controls if defaultSet is added
agg.defaultSet =@nmcAgg         <-- set tag reference

real.addDefault=true            <-- controls if defaultSet is added
real.defaultSet=@nmcReal        <-- set tag reference
```

Creating a New Resource Type

To create a new resource type, perform the following steps:

1. Add the required `rtN_xxx` entries at the end of the appropriate basic data file.

   Note: The sequence numbers, the `N` part of `rtN`, must be sequential with no intervening gaps.

2. To define the geometric shape displayed on a topology view, specify the appropriate package qualified class names as shown in the following list:
   - com.tivoli.ihs.reuse.gui.IhsCircle
   - com.tivoli.ihs.reuse.gui.IhsCircleStar
   - com.tivoli.ihs.reuse.gui.IhsDiamond
   - com.tivoli.ihs.reuse.gui.IhsHexagon
   - com.tivoli.ihs.reuse.gui.IhsOctagon
   - com.tivoli.ihs.reuse.gui.IhsOval
   - com.tivoli.ihs.reuse.gui.IhsParallelogram
   - com.tivoli.ihs.reuse.gui.IhsPentagon
   - com.tivoli.ihs.reuse.gui.IhsRectangle
   - com.tivoli.ihs.reuse.gui.IhsRoundRect
   - com.tivoli.ihs.reuse.gui.IhsSolidLine
   - com.tivoli.ihs.reuse.gui.IhsStar
   - com.tivoli.ihs.reuse.gui.IhsTrapezoid
   - com.tivoli.ihs.reuse.gui.IhsTriangle

Example: `rtN_geometric=com.tivoli.ihs.reuse.gui.IhsHexagon`

To use the new resource type in a view, set the `anN_t` attribute of a resource to the `rtN_name` attribute of your new resource type. For example: `an1_t=2147614793`.

Integrating Captured Views into the Demonstration

To integrate the captured views into the demonstration, update the business tree and rename the navigation views.

Updating the Business Tree

After the business tree has been saved, some work is required to make it usable by the NetView management console code in demonstration mode. Note that the original business tree file, shipped with NetView management console is the

avail.control.md.nmc file. It is located in the <installRoot>\bin\generic_unix\TDS\client\views\backup directory you created before saving views. This file illustrates the parent-child relationships used within the business tree to create the tree node, branches, and leaf nodes.

To make the saved business tree available, perform the following steps from the client workstation.

1. From the command prompt, rename the saved business tree view in the following way:
   a. `cd <installRoot>\bin\generic_unix\TDS\client\views`
   b. `copy 1.ctl.md  avail.control.md.nmc`

   Notes:
   a. The root node of the business tree is always `resource ID 1`. For more detailed information about the resource ID, see "Finding the Resource ID" on page 54.
   b. If you copy the file, instead of renaming it, you can keep the original if you want to restart the process during this procedure.
2. Edit the avail.control.md.nmc file.
3. Replace the complete view model at the top of the file with the line in the **After** column in Table 32. For more detailed information about the view model, see "Defining View Information" on page 56.

Table 32. Replacing the view model text example (before and after).

Before	After
view_id=xxxx	view_id=availAvail
view_prev_id=-1	view_width=500
view_useimage=-1	view_height=900
view_openview=0	
view_width=xxxx	
view_height=xxxx	
view_cust=true	
view_customizable=true	

4. Remove all parent references to the root node of the tree by following these steps:
   a. Locate every parent reference by searching for the string `_p=1`.
   b. Delete each of these lines containing `_p=1`.
   c. Verify that the value is `1` (and not `1x` or `1xx`, for example).
5. Update the resource type for every node by inserting the `RT` prefix as shown in Table 33.

Table 33. Updating the resource type example (before and after).

Before	After
anx_t=536871171	anx_t=**RT**536871171

6. Verify that the special business tree nodes shown in Table 34 on page 52 are defined to use these specific resource type values. If a different value is present as the `anX_t` attribute, replace it with the required value shown in Table 34 on page 52.

By doing this, you can avoid error messages when NetView management console is started and have the correct icon displayed in the business tree.

Table 34. Verifying the resource type values for business tree nodes.

Special Business Tree Node (found in the anX_l attribute)	Required Resource Type (defined in basicdata.nmc)
Business Systems	RT536871171
Systems Management Business System	RT536871228
Networking	RT536871168
Network Views	RT536871169
Exception Views	RT536871170

7. Remove the following attributes from every node in the business tree:
   - `anx_da=true`
   - `anx_def=true`
8. Save your changes.

   To verify these changes, perform the following steps:
   a. Start the NetView management console demonstration.
   b. Verify that the business tree is properly displayed.
   c. Verify that the views you saved, which were initiated from the business tree, can open.
9. Integrate the captured views into the demonstration by renaming the navigation views to conform to the naming conventions for your demonstration.
10. Define any resource types in the saved views that were not previously defined.

Renaming Navigation Views

To rename the navigation view, first locate the correct navigation file. All of the target navigation files are located in the <installRoot\bin\generic_unix\TDS\client\views> directory. Table 35 shows the naming conventions.

Table 35. Naming conventions for navigation views.

Navigation Context Menu	Target File Naming Convention
Configuration > Parents	<resource ID>.par.ctl.md
Configuration > Children	<resource ID>.chd.ctl.md
Configuration > Peers	<resource ID>.peer.ctl.md
Configuration > Logical and Physical	<resource ID>.lp.ctl.md
Configuration > Logical	<resource ID>.log.ctl.md
Configuration > Physical	<resource ID>.phy.ctl.md
Configuration > Backbone	<resource ID>.bak.ctl.md
Locate Failing Resources	<resource ID>.fp.ctl.md
Locate Resource	<locate name>.locate.md
More Detail	<resource ID>.ctl.md

Defining New Resource Types in Saved Views

The views you save might contain resources that are not currently defined for the demonstration. This section provides the steps necessary to define missing RODM resource types to the demonstration. When a resource type is referenced in a view that is not currently defined, messages similar to the following messages are written to `stderr` when the view is opened.

```
IhsViewModel:verifyView(save) Thread-8 p1=64 p2=ResourceType (2416050177) specified
   on 2 was not found
IhsViewModel:verifyView(save) Thread-8 p1=64 p2=Setting 2's resource type to
   IhsNodeRT
IhsViewModel:verifyView(save) Thread-8 p1=1085 p2=ResourceType (2147549291)
   specified on 3 was not found
IhsViewModel:verifyView(save) Thread-8 p1=1085 p2=Setting 3's resource type to
   IhsLinkRT
```

where:

- `2416050177` is the resource type specified in the `anX_t` attribute of a node resource.
- `2` is the resource ID specified in the `anX_r` attribute of a node resource.
- `2147549291` is the resource type specified in the `alXt` attribute of a link resource.
- `3` is the resource ID specified in the `alX_r` attribute of a link resource.

In a view, these resources display as icons.

All of the information required to define a RODM resource type is available from the Legend window when you are connected to a live NetView management console system. To define missing resource types referenced by saved views from the client workstation, perform the following steps.

1. Select **Help→Legend...** to open the Legend window.
2. To locate the resource type, search the ID column of the Legend window for a decimal value (2416050177, for example). This is the resource type that must be added.
3. Edit the basicdata.nmc file.

   For more information about this file, see "Using Basic Data Files" on page 45.
4. Using the information in Table 36, create a new resource type at the end of the currently defined set.

Table 36. Creating a new resource type from the management console – Legend window.

Resource Type Attribute	Description	Legend Column
rtX_name	Name	use the decimal ID value (2nd value)
rtX_desc	Description text	Resource Type
rtX_image	Image file name	Icon File Name
rtX_geometric	Geometric shape	map shape to the corresponding geometric
rtX_help	Help	Help File Name
rtX_flags	Flags	Flags

For more information about creating new resource types, see "Creating a New Resource Type" on page 50.

The following example shows how an Advanced Peer-to-Peer Networking end node can be defined:

```
rtX_name=327883
rtX_desc=APPN end node (EN)
rtX_image=duiu5n00.gif
rtX_geometric=com.tivoli.ihs.reuse.gui.IhsOctagon
```

5. Save your changes.
6. Start the NetView management console demonstration and verify that the proper icon is now displayed.

Finding the Resource ID

The topology server assigns each resource an internal resource ID value. This value is important for demonstration mode because it is used to generate the name of navigation view files. To determine the ID of a resource on a live NetView management console system, perform the following steps.

1. Sign on.
2. Select the **Options** menu item and then **Console Properties...** to open the **Console Properties** notebook.
3. Select the **Service** tab, which is the far right tab in the notebook. If it is not visible, scroll to it.
4. Check **Action** in the **Components** section.
5. Check **Debug** in the **Trace types** section.
6. Click **OK**.
7. Select the view that contains the resource ID you want to find.
8. Click the resource to open the context menu for that resource and select **Resource Properties**.
9. Select the **Debug** tab. The Resource Properties window displays debugging data about the resource.
10. Look for `ibmId` in the `property` field. The corresponding `Value` is the ID for the resource. The ID value is displayed in decimal and hexadecimal. Usually, you need to use the decimal value.

Defining a Node Resource in a View

The attributes used to define a node resource in a view are shown in Table 37.

Table 37. Attributes to define a node resource in a view.

Attribute	Definition
anX_i	Display ID (unique per view).
anX_r	Server ID (unique per server); double clicking the resource drills to <serverID>.md.
anX_p	Parent ID (optional, none).
anX_t	Resource type (optional, defaulted).
anX_s	Status (optional, normal).
anX_u	Flags (optional, zero).
anX_a	Determines if this is an aggregate resource (optional, false).
anX_da	Suppress '+' on aggregate resources. For Topology Display Subsystem view only, others not *live* if suppressed (optional, value of anX_a).
anX_l	Label (optional, blank).

Table 37. Attributes to define a node resource in a view. (continued)

Attribute	Definition
anX_lx	Label X coordinate (optional, auto).
anX_ly	Label Y coordinate (optional, auto).
anX_d1	Data 1: TDS=HB1, NMC=RODM other data (optional, blank).
anX_d2	Data 2: TDS=HB2, NMC=RODM customer data (optional, blank).
anX_d3	Data 3: TDS=HB3, NMC=not used (optional, blank).
anX_d4	Data 4: TDS=HB4, NMC=not used (optional, blank).
anX_per	On the business tree view, determines if a view is permanent (optional, false).
anX_cag	Specifies if customer aggregation is allowed (Not currently in use.) (optional, true).
anX_c1	Monitor Count 1 (optional, none).
anX_c2	Monitor Count 2 (optional, none).
anX_x	X coordinate.
anX_y	Y coordinate.

In the documenting class, `com.tivoli.ihs.client.view.IhsNode`, the following example shows how a node resource can be defined:

```
an3_i=3
an3_r=2100
an3_p=39
an3_t=327882
an3_l=NETA.FVT01EM
an3_s=18
an3_per=false
an3_x=342
an3_y=106
```

Defining a Link Resource in a View

The attributes used to define a link resource in a view are shown in Table 38.

Table 38. Attributes to define a link resource in a view.

Attribute	Definition
alX_i	Display ID (unique per view).
alX_r	Server ID (unique per server); double clicking the resource drills to<serverID>.md.
alX_p	Parent ID (optional, none).
alX_t	Resource type (optional, defaulted).
alX_s	Status (optional, normal).
alX_u	Flags (optional, zero).
alX_a	Determines if this is an aggregate resource (optional, false).
alX_da	Suppress '+' on aggregate resources. For Topology Display Subsystem view only, others not "real life" if suppressed (optional, value of alX_a).
alX_l	Label (optional, blank).
alX_lx	Label X coordinate (optional, auto).
alX_ly	Label Y coordinate (optional, auto).

Table 38. Attributes to define a link resource in a view. (continued)

Attribute	Definition
alX_d1	Data 1: TDS=HB1, NMC=RODM other data (optional, blank).
alX_d2	Data 2: TDS=HB2, NMC=RODM customer data (optional, blank).
alX_d3	Data 3: TDS=HB3, NMC=not used (optional, blank).
alX_d4	Data 4: TDS=HB4, NMC=not used (optional, blank).
alX_per	On the business tree view, determines if a view is permanent (optional, false).
alX_cag	Specifies if customer aggregation is allowed (Not currently in use.) (optional, true).
alX_c1	Monitor Count 1 (optional, none).
alX_c2	Monitor Count 2 (optional, none).
alX_1	First end point able to be displayed.
alX_2	Second end point able to be displayed.
alX_d	Link direction with respect to the first end point • 0=none (default) • 1=origin • 2=destination • 3=bidirectional • 4=replica

In the documenting class, `com.tivoli.ihs.client.view.IhsLink`, the following example shows how a link resource can be defined:

```
al15_r=268
al15_p=216
al15_l=40001A20AC05
al15_d1=Bridge=3F10TOP, Segment=020A, MAC Address=40001A20AC05
al15_per=false
al15_1=28
al15_2=9
```

Defining View Information

Table 39 shows the attributes used to define the appearance of a view.

Table 39. Attributes to Define The Appearance of a View

Attribute	Definition
view_id	ID for the view
view_prev_id	Previous ID for the view
view_width	Width of view (optional, 500)
view_height	Height of view (optional, 300)
view_layer	Layer of view
view_descriptor	Descriptor of view (optional, "")
view_fgcolor	Foreground text color of view; RGB value (optional, black)
view_ftcolor	Free text color of view; RGB value (optional, black)
view_useimage	Specifies whether to use image(1)/color(0)/notSet(-1) for view background (optional, -1)
view_bgcolor	If color, the background color RGB value (optional, gray)

Table 39. Attributes to Define The Appearance of a View (continued)

Attribute	Definition
view_bg	If image, specific image file name
view_bgx	If image, image X location within view (optional, -1)
view_bgy	If image, image Y location within view (optional, -1)
view_bgwidth	If image, width (optional, -1)
view_bgheight	If image, height (optional, -1)
view_swidth	Width of resource symbol area (optional, 40)
view_sheight	Height of resource symbol area (optional, 40)
view_cust	Specifies if the view is customized (optional, false)
view_customizable	Specifies if the view can be customized (optional, true)
view_ffu	Specifies whether or not to force full update of view (optional, false)
view_orp	Specifies if override refresh property (optional, false)
view_activetab	View automatically opened for tab (optional, first tab)
view_openview	Open view option3 (topology or detail view)#

In the documenting class, `com.tivoli.ihs.client.view.IhsViewModel`, the following example shows how the appearance of a view can be defined:

```
al15_r=268
al15_p=216
al15_l=40001A20AC05
al15_d1=Bridge=3F10TOP, Segment=020A, MAC Address=40001A20AC05
al15_per=false
al15_1=28
al15_2=9
```

Defining a Demonstration View

A demonstration view is an ASCII flat file that is typically created as described in "Capturing Live Views from your NetView Management Console System" on page 43. It contains the following collection of object definitions:

1. A view model definition, as described in "Defining View Information" on page 56.
2. Zero or more node definitions, as described in "Defining a Node Resource in a View" on page 54.
3. Zero or more tack point definitions.
4. Zero or more free text definitions.
5. Zero or more link definitions, as described in "Defining a Link Resource in a View" on page 55.

The following example shows how a demonstration view can be defined.

```
################################################################################
# View Definition File
#
# Resource: 89
#
# Created:  Mon Feb 01 10:36:29 PST 1999
#
# Warning:  Be careful if you modify this by hand!
################################################################################
```

```
view_id=89
view_prev_id=-1
view_bgcolor=-1
view_fgcolor=-16777216
view_ftcolor=-16777216
view_useimage=1
view_openview=0
view_width=398
view_height=240
view_cust=true
view_customizable=true
view_descriptor=NETA.4-MDL
################################################################################
# Node objects
################################################################################
an1_i=1
an1_r=473
an1_p=461
an1_t=2147811538
an1_l=D:NETA.NRILOV00
an1_u=134217728
an1_a=true
an1_da=true
an1_per=false
an1_x=200
an1_y=60
an2_i=2
an2_r=474
an2_p=461
an2_t=327776
an2_da=true
an2_per=false
an2_x=200
an2_y=180
################################################################################
# Tackpoint objects
################################################################################
################################################################################
# Free Text objects
################################################################################
################################################################################
# Link objects
################################################################################
al1_i=3
al1_r=475
al1_p=461
al1_l=IC:NETA.NRILOV00.USIBMNT.NTFEMVS
al1_s=20
al1_a=true
al1_da=true
al1_per=false
al1_1=1
al1_2=2
```

Chapter 7. Topology Console Java Applications and Plug-ins

This chapter contains Programming Interface information.

This chapter describes the Java applications and plug-ins provided with the NetView management console. You can write applications or plug-ins to enhance topology console operation.

The example Java applications and plug-ins are installed as part of the *NetView Management Console Productivity Kit* (on the Windows platform, only). Be sure to perform a custom installation of the NetView management console topology console and select the *NetView Management Console Productivity Kit*. Because the technical information about these applications and plug-ins is dynamically created using javadoc, the applications and plug-ins cannot be described in their entirety. References to these examples are made here. The *NetView Management Console Productivity Kit* contains more detailed technical information (such as methods, fields, syntax and Java class hierarchy). To access this information, in the Windows environment, double-click the **NMC Productivity Kit** icon on the desktop.

Note: Though the *NetView Management Console Productivity Kit* can be installed only on the Windows platform, Java application and plug-in functions are available on any supported NetView management console topology console platform.

Supplied Support Files

The files shown in Table 40 provide additional support for the Java application and Java plug-in examples.

Table 40. Files that Support the Java Examples

File	Description
examples\java\ExampleJavaApp.jar	This Java ARchive (JAR) file contains the compiled class files of the examples, so they are immediately usable.
examples\support\ExampleJavaApp.rsp	This server Command Profile Editor (CPE) response file is used to create the context menu command definitions for the example Java applications so that the server can include them in context menus.
examples\support\plugins.properties	This is a plug-in definition file that loads all of the example plug-ins.
examples\javadoc	This is the starting point for the console API documentation.

To access these files, see the Supplied Support Files section of the *NetView Management Console Productivity Kit*.

Installing the Examples

The examples and support files are installed as part of the *NetView Management Console Productivity Kit*. Currently, the *NetView Management Console Productivity Kit* can be installed only on the Windows platform. See the Supplied Support Files section of the *NetView Management Console Productivity Kit* for more information.

Enabling the Examples

Although the examples have been installed, certain steps are required to enable their use. The exact steps vary, depending on how you want to access the examples. You can access the examples either while connected to a server or while running the console in demonstration mode.

To run the examples while signed on to a server, perform these steps:

1. From the server workstation, access the console workstation where you have installed the example files.
2. Change to the `lib` directory: `cd %BINDIR%\TDS\server\db\current\lib` where the `BINDIR` environment variable defines the installation root of the server.
3. Copy the <console_system_install_path>\bin\generic_unix\TDS\client\examples\java\ExampleJavaApp.jar file to this directory. The ExampleJavaApp.jar file is now automatically downloaded to each console that subsequently signs on to this server.

To run the examples in demonstration mode, perform these steps:

1. From the console workstation, change to the <install path>\bin\generic_unix\TDS\client\examples\support directory.
2. Issue the following command:

   ```
   copy plugins.properties ..\..\settings
   ```

Compiling the Examples

It is not necessary to compile the examples before they are used. A provided JAR file, examples\java\ExampleJavaApp.jar contains the compiled examples. See Table 40 on page 59 for a brief description of this file.

If you change an example, you need to recompile the example. Before you recompile the examples, perform the following steps from a console workstation:

1. Install the console code.
2. Install Java™2 SDK, Standard Edition, v4.2 (J2SDK).

To recompile the examples after you have changed them, perform the following steps:

1. Change to the <install path>\bin\generic_unix\TDS\client\examples\java directory.
2. Compile the examples by issuing the following command:

   ```
   javac -classpath %CLASSPATH%;..\..\lib\ihseuc.jar *.java
   ```

   Note: To compile cleanly, the `CLASSPATH` environment variable must include the console code and the J2SDK classes.
3. Create a new JAR file (to contain the compiled classes) using the following command:

   ```
   jar -cfv ExampleJavaApp.jar *.class
   ```

4. Make the new JAR file available. For more information about this process, see "Enabling the Examples" on page 60.

Tracing the Examples

The examples have been instrumented with RAS tracing. You can enable this tracing from the Service page of the Console Properties notebook. To enable tracing, perform the following steps:

1. Select the **Options** menu item and then **Console Properties...** to open the **Console Properties** notebook.
2. Select the **Service** tab, which is the far right tab in the notebook. If it is not visible, scroll to it.
3. Select the **Customer** component.
4. In the Trace types section, click one or more of the following check boxes:
   - **Constructors**
   - **Public methods**
   - **Callback methods**
5. In the Additional Tracing Controls section, check the **Details** box so that all of the traced data is displayed.
6. Click **OK**.

Problem Determination

Message `IHS1011W` is used to report problems encountered while loading a plug-in. Because plug-ins are loaded before the console window has been created, this message cannot be displayed in a pop up window or recorded in the Log window. Therefore, it is written to `stderr`.

The following errors are reported in the `Why` field that is specific to the plug-in:

The specified plug-in class does not exist

```
WARNING: Could not instantiate bean "PlugIn.Does.Not.Exist"
from JAR "d:\Tivoli\bin\w32-ix86\..\generic_unix\TDS\client\lib\ExampleJavaApp.jar"
    We couldn't open the class file "PlugIn/Does/Not/Exist.class" in the JAR
IHS1011W: Unable to start a Java application.
Class: PlugIn.Does.Not.Exist
Why: Plug-in class not found
Phase: 1
```

Note: The first 3 lines of the preceding example are generated by the dynamic Java class loading mechanism of the console.

The specified plug-in class exists but does not implement the IhsIPlugIn interface

```
IHS1011W: Unable to start a Java application.
Class: com.tivoli.ihs.client.IhsClientArgs
Why: Plug-in does not implement IhsIPlugIn interface
Phase: 2
```

Java Applications

A Java application consists of customer-written code that is initiated from a context menu item (as a context menu item). The Application is defined in the command profile editor and runs on the topology console (a Java virtual machine).

The context menu is defined to the topology server using the CPEBATCH batch utility.

Java applications can be either resource dependent or resource independent. Dependent Java applications provide information about each selected resource. The `com.tivoli.ihs.client.action.IhsResInfo` class, for example, provides this information. Java applications provide information about their associated context menu items. The `com.tivoli.ihs.client.action.IhsCmdInfo` class, for example, shows this. A Java application extends the `IhsJavaApplicationAdapter` class.

After it is launched, the application can access any services provided by Java, certain topology console services, or any additional customer or third party services. A command can be generated and issued using any of the command exits provided by the topology server. For more information about command exits, see Chapter 13, "Using NetView Management Console Command Profiles," on page 119.

Note: For more details about any of the following items, see the Java Applications section of the *NetView Management Console Productivity Kit*:

- The `com.tivoli.ihs.client.action.IhsResInfo` class
- The `com.tivoli.ihs.client.action.IhsCmdInfo` class
- The `IhsJavaApplicationAdapter` class
- Topology console services

Java Application Examples

The example Java applications shown and described in Table 41 are provided with NetView management console.

Table 41. Available Java Application Examples

File	Description
examples\java\ExampleGUIJavaApp.java	Displays context in a window. It shows all of the available information about the selected resources. It can be used as either a resource dependent or resource independent command.
examples\java\ExampleCmdWithResponse.java	Illustrates sending a command to the IBM Tivoli NetView for z/OS program and receiving all response lines for subsequent processing.
examples\java\ExampleGUItoIML.java	Illustrates writing a complex graphical interface. It displays the data required to start a 3174 device so that the user can select appropriate options. A command is constructed from the user input and sent to the IBM Tivoli NetView for z/OS program. The results from the command are displayed in the console Log window.

Note: For detailed information about these Java classes, see the Java Application Examples section of the *NetView Management Console Productivity Kit*.

Java Application Development Process

Use the following software development steps when developing a Java application:

1. Edit your application:
   a. Use one of the provided Java application examples as a starting point.
   b. Customize the example to meet your specific requirements.

2. Compile, package, and deploy your application. For more information about this process, see "Compiling the Examples" on page 60.
3. Enable the new Java application. For more information about this process, see "Defining the Example Java Applications."
4. Test the Java application in the following way:
   a. Sign on to the server containing the new code.
   b. Display the Java application in a context menu.
   c. Select the menu item and verify that it is working properly.
   d. If necessary, enable service tracing options to help diagnose problems. For more information about tracing, see "Tracing the Examples" on page 61.

Defining the Example Java Applications

From the Server

To run the example Java applications while signed on to a server, perform the following steps:

1. Go to the command prompt of a workstation on which the NetView management console topology server and console are both installed.
2. Use the server `CPEBATCH` program to define the example Java application so that it is displayed in the context pop-up menu for a resource.
3. Change to the bin directory (where the `BINDIR` environment variable defines the installation root of the server): `cd %BINDIR%\TDS\server\bin`.
4. Issue the following command:

   ```
   cpebatch <console machine install path>\bin\generic_unix\TDS\client\examples
   \support\ExampleJavaApp.rsp -i -g
   ```

In Demonstration Mode

To run the application examples in demonstration mode, from the console workstation, perform the following steps:

1. Change to the `<install path>\bin\generic_unix\TDS\client\settings` directory.
2. Edit the `basicdata.nmc` file, locating the `set.5.debug` key (which is part of the `@exJavaApps` group) and changing the value from `true` to `false`.

Running the Example Java Applications

From the Server

To run the example Java applications from the server, perform the following steps:

1. Start the console using the standard desktop icon.
2. Sign on to the appropriate server.
   - For resource independent examples, perform the following steps:
     a. Position the cursor over any white space in the business tree.
     b. Right-click to display a context menu.
     c. Select an example menu item.

     Note: The context menu items for the example Java applications are named `Example: <application>`.
   - For resource dependent examples, perform the following steps:
     a. Open a view that contains at least one real resource.
     b. Right-click the real resource to display a context menu.

c. Select an example menu item.

Note: The context menu items for the example Java applications are named `Example: <application>`.

In Demonstration Mode

To run the example Java applications in demonstration mode, perform the following steps:

1. Start the console in demonstration mode using the standard desktop icon.
   - For resource independent examples, perform the following steps:
     a. Position the cursor over any white space in the business tree.
     b. Right-click to display a context menu.
     c. Select an example menu item.
   - For resource dependent examples, perform the following steps:
     a. Select **Tasks → Locate Resource** to open the Locate Resource window.
     b. Type `NTFFPU20` in the entry field and then click the **Locate** button.
     c. In the view that opens, position the mouse over one of the nodes.
     d. Right-click to display a context menu.
     e. Select an example menu item.

Java Plug-Ins

Use plug-in code to control various aspects of console operation. Like a Java application, a plug-in is also Java code that runs within the topology console Java virtual machine, but differs from a Java application in that plug-ins have the following characteristics:

- Are not related to any view or resource.
- Are loaded each time you sign on to a topology server (after the plug-ins definition properties file is downloaded from the server). For more information about the definition properties file see "Plug-In Definitions File" on page 66.
- Remain loaded and active as long as the operator is signed on.
- Have call-back methods that are driven as many times as necessary.

Supported Plug-Ins

The topology console currently supports the following plug-ins:

- View Label Formatter Plug-in
- Log Window Filter Plug-in

View Label Formatter Plug-In

Use the view label formatter plug-in to programmatically control the label text displayed on a topology view, a details view, or selected data windows. This plug-in must implement the IhsIPlugInViewLabel interface. To see detailed specifications, see the Java Plug-ins section of the *NetView Management Console Productivity Kit*.

Built-in console function provides some control over the displayed label text (for example, it might be truncated on the left or right), but the number of displayed characters cannot be controlled by this console function. This plug-in can be useful if you are adhering to resource naming conventions that include fixed prefixes or suffixes.

Some possibilities for customization with this plug-in include the following actions:

- Displaying the first *N* characters (where *N* is a number you specify)
- Removing common prefix or suffix text
- Combinations of removing and displaying text

The following windows are examples of data windows that use this plug-in when displaying resource names:

- Command for a Multi-owned Resource
- Event Viewer
- List Suspended Resources
- Session Data
- Status History

Note: The Resource Properties window intentionally does *not* use this plug-in so that the full resource name is always available.

Log Window Filter Plug-In

Commands, their generated responses, and console generated messages are centrally collected and displayed in the console Log window. Use the log window filter plug-in to programmatically control the Log window contents. This plug-in must implement the `IhsIPlugInLog` interface. For detailed information about the `IhsIPlugInLog` interface, see the Java Plug-ins section of the *NetView Management Console Productivity Kit*. A new Log window entry can be handled in the following way:

- It can be added as is.
  - The standard Log window display color is automatically used.
- It can be added with changes:
  - The text can be modified.
  - An override to the standard Log window display color can be specified.
- It can be suppressed.

You can also use this plug-in to initiate customer specific processing that results from a particular command.

Additional Plug-In Support

To determine which plug-ins are loaded, see the Environment Information window. The information shown in Table 42 is displayed for each active plug-in. For more information, see the Java Plug-Ins section of the *NetView Management Console Productivity Kit*.

Table 42. Values for Plug-ins as Shown in the Environment Information Window

Field Column	Value Column
<plug-in class name>:data	Initialization data passed to the `setPlugInData()` method.
<plug-in class name>:desc	Description as provided by the `getPlugInDescription()` method.
<plug-in class name>:version	Version as provided by the `getPlugInVRM()` method.

Table 42. Values for Plug-ins as Shown in the Environment Information Window (continued)

Field Column	Value Column
<plug-in class name>:debug	Debug information as provided by the `toString()` method. **Note:** If the string returned by `toString()` contains dynamic information, the Refresh button can be used to update this value without closing the window. The example plug-ins provide a template for this.

Notes:

1. The `-noPlugin` command-line argument overrides automatic plug-in loading as each console calls the plug-in.
2. When specified, the plug-in properties file is not processed after sign on.

Plug-In Definitions File

The plugins.properties file defines the console plug-ins that are to be loaded. This file is downloaded, after sign on, from the db\current\settings directory of the server.

Syntax for the plugins.properties file is shown in the following example:

```
*****************************************************************************
* Define the plug-ins to be loaded during Console initialization
* for every console that signs on to this server
*****************************************************************************
plugin.1.class = required full package qualified class name
plugin.1.data  = optional data passed to the setPlugInData() method for runtime use
plugin.2.class = <another>
plugin.3.class = <another>
...
```

Plug-In Examples

The example Java plug-ins shown in Table 43 can be used to enhance base console operation.

Table 43. Java Plug-in Examples

File	Description
examples\java\ExampleLogPlugIn.java	Provides control over items (commands, responses, or messages) that are added to the console Log window Also see "Log Window Filter Plug-In" on page 65.
examples\java\ExampleViewLabelPlugIn.java	Provides control over the label that is displayed for a resource Also see "View Label Formatter Plug-In" on page 64.

Notes:

1. To use these example plug-ins to meet your specific requirements, specify the parameters in the data definition of the appropriate plug-in.
2. More information about all of the following items is available in the Java Plug-Ins section of the *NetView Management Console Productivity Kit*:

- Log window filter
- `com.tivoli.ihs.extern.plugin.IhsIPlugInLog` interface
- View label formatter
- `com.tivoli.ihs.extern.plugin.IhsIPlugInViewLabel` interface
- `ExampleLogPlugIn` class
- `ExampleViewLabelPlugIn` class

Plug-In Development Process

To develop a plug-in, use the following steps:

1. Edit an example in the following way:
   a. Use one of the provided plug-in examples as a starting point.
   b. Customize the example to meet your specific requirements.
2. Compile, package, and deploy your plug-in. For more information about this process, see "Compiling the Examples" on page 60.
3. Enable the plug-in. For more information about this process, see "Defining the Example Java Plug-Ins."
4. Test the plug-in using the following steps:
   a. Sign on to the server where the plug-in code resides.
   b. From the Environment Information window, perform the following steps:
      1) Verify that the plug-in has been successfully loaded.
      2) Examine any debugging information externalized by the `toString()` method.
   c. Generate a scenario in which the function for the plug-in code should be performed and verify that it is working properly.
   d. If necessary, enable service tracing to help diagnosis problems. For more information about tracing, see "Tracing the Examples" on page 61.

Defining the Example Java Plug-Ins

From the Server

To run the example plug-ins while signed on to a server, perform the following steps:

1. Go to the command prompt of a workstation on which the NetView management console topology server and console are both installed.
2. Change to the settings directory (where the `BINDIR` environment variable defines the installation root of the server): `cd %BINDIR%\TDS\server\db\current\settings`
3. Copy the following file to the %BINDIR%\TDS\server\db\current\settings directory: <console machine install path>\bin\generic_unix\TDS\client\examples\support\plugins.properties.

In Demonstration Mode

To run the example plug-ins in demonstration mode, from the console workstation, perform the following steps:

1. Change to directory <install path>\bin\generic_unix\TDS\client\examples\support.
2. Issue the following command: `copy plugins.properties ..\..\settings`

Running the Example Java Plug-Ins

Running the Log Window Filter Plug-In: In a Live NetView Management Console System

Start the console by double clicking the standard desktop icon.

- To suppress message IHS2267:
  1. Select **Tasks → Send Message...** from the server or another console to broadcast a message to this console.

     The broadcast message entered is displayed (prefixed by IHS2267) in a message box. The example plug-in suppresses the addition of this message to the Log window.
- To suppress message IHS1107:
  1. Select **Tasks → Locate Resource** to open the Locate Resource window.
  2. Type NEVER in the entry field and then click the **Locate** button.

     Message IHS1107 is displayed in a message box. The example plug-in suppresses the addition of this message to the **Log** window.

Note: Locate Resource generates message IHS1080 in demonstration mode.

Running the View Label Plug-In: In Demonstration Mode

To begin, start the console in demonstration mode by double clicking the **NMC Demo** desktop icon and perform the following steps:

1. Select **Tasks → Locate Resource** to open the Locate Resource window.
2. Type NTFFPU20 in the entry field and click the **Locate** button.
   - A view is displayed.
   - The names of the three nodes in this view are network qualified with USIBMNT.
   - The example plug-in suppresses the display of the network qualifier on the view (in either topology or details mode).
3. Position the cursor over the label of a node to display the fully qualified name in the fly-over section of the status bar.

Chapter 8. Configuring Property Files for Locally Launched Applications

The NetView management console provides the capability to launch local applications on the NetView management console topology console workstation, such as a Web browser, to view a specific Web address in context or to start a telnet session to the host where a managed resource resides. A properties file associated with these locally launched applications defines the specific executable file to launch for each supported platform. The Web browser and telnet commands are already configured. However, you can check the properties file to ensure that the commands specified can be successfully run without a specific dependency on the current drive and directory. For all other local applications, define appropriate entries in the properties file.

Configure the local applications in the following way:

1. Define which application executable file is to be used on each platform where the topology console runs. This can be configured on a user basis or on a server-wide basis. This configuration is done at the topology server.
2. Define the command entries that are to be added to the topology console pop-up menu. This configuration is done at the topology server.

Defining the Pop-up Menu Items

The cpebatch utility, on the NetView management console topology server workstation, can be used to define commands that appear on the topology console pop-up menu. This is done by creating a response file and using it as input to the cpebatch utility. You can generate response files manually, with a standard text editor. To define an application to the NetView management console command menu, define the following variables in the command response file:

- `EXIT_NAME` to be `IHSXTJAM`
- `COMMAND_STRING` to be `com.tivoli.ihs.nmc.cmd.IhsCommandInvoker cmdname var1=value1 var2=value2...`

Although the command name is arbitrary, the properties file needs a matching entry (`cmdname`) for the command name to be resolved. If no match exists, then the command name is run as is.

Response File Input

To enable the application, command information must be added to the commands database through the command profile editor utility.

The following example shows an entry in a command response file:

```
COMMAND = (
   NAME = CISCO_BLUE
   MENU_STRING = "CISCO APPN node detail view"
   RESOURCE_INDEP=NO
   HTML_HELP_FILE =
   HTML_HELP_ANCHOR =
   MIN_RESOURCES = 1
   MAX_RESOURCES = 1
   VERIFY = NO
   PAGE = (
```

```
      COMMAND_STRING = "com.tivoli.ihs.nmc.cmd.IhsCommandInvoker browser
      URL=http://%RODM.ManagementURL%/cgibin/cw-blue/snamaps\?rqx=9&a;=nd&ip;=
      %ipaddress%&rc;=public"
      EXIT_NAME = IHSXTJAM
      MANAGER_NAME = ANY
      HTML_HELP_FILE =
      HTML_HELP_ANCHOR =
      CLIENT_PLATFORM_LIST = GENERIC
      TARGET_PLATFORM_LIST = GENERIC
   )
)
```

Note: This is only an example. Specify the Web address for your own environment in this section:

```
URL=http://%RODM.ManagementURL%/cgibin/cw-blue/snamaps\?rqx=9&a;=nd&ip;=
%ipaddress%&rc;=public
```

The entry for COMMAND_STRING determines how the command is called. For example, COMMAND_STRING for the `browser` command must start with the keyword `com.tivoli.ihs.nmc.cmd.IhsCommandInvoker` followed by `browser`.

COMMAND_STRING contains a definition for the Web address field. A variable can be included in the Web address field that pulls data from Resource Object Data Manager (RODM). The following example shows the syntax of the variable name:

```
%RODM.ManagementURL%
```

This variable is not allowed on resource independent commands. For more detailed information about RODM variables, see Table 46 on page 72.

Creating a Response File for Browser

You can create response files manually, with a standard text editor. In addition, when creating a response file for the browser command, you can use the registration file conversion utility if a Distributed NetView registration file was provided by an equipment vendor. To manually create a response file, see "Response File Input" on page 69.

If you received a Distributed NetView Navigation Bar Registration file, you can use the registration file conversion utility to convert that file to a response file. Change to one of the following directories:

- For Windows: `cd %BINDIR%\..\generic_unix\TDS\client\bin`
- For UNIX: `cd $BINDIR/../generic_unix/TDS/client/bin`

The following syntax is for the registration file `OEMNAVBAR.REG`:

```
tapp<i>xx</i> .. com.tivoli.ihs.nmc.server.IhsHttpParse <path>OEMNAVBAR.REG
```

Where *xx* is the appropriate platform from which the topology console is running. See Appendix B, "Topology Console Commands," on page 157 for more information about the **tapp***xx* command.

Note: The default output is device dependent, but the menu entry is shown for any manager. Any changes to the response file need to be made before running the **cpebatch** command.

The previous syntax generates an `OEMNAVBAR.REG.rsp` file. The generated response file is used as input to the cpebatch utility in the following way:

```
cpebatch OEMNAVBAR.REG.rsp -i -g
```

Changes to the right-click pop-up menu take effect immediately.

Defining the Properties File

The properties file has a heading section that defines generic information followed by sections defining a specific executable file and command string for each platform. Any line starting with a semicolon is considered a comment line. Comments cannot be on the same line as command text.

The following list shows file name types:

- Default File Name: %BINDIR%\TDS\server\db\current\settings\ defaultcmdinv.properties
- User File Name: %BINDIR%\TDS\server\db\current\settings\ <user_name>cmdinv.properties

Note: The file name must be lowercase regardless of the user name.

In the properties file, define the generic and operating system sections. Table 44 displays the information for the generic section of the properties file.

Table 44. Generic Section of Properties File

Field	Description
command.desc	Defines the application entry. You can define as many different application specifications as necessary. This information is not used outside this file. In these examples, replace *command* with the name of the command you are defining.

Table 45 displays the information for the browser section of the properties file.

Table 45. Browser Section of Properties File

Field	Description
browser.usebuiltin	If set to TRUE, the built-in NetView management console browser is used. All operating system entries are ignored. This field is specific to the browser command and is not used for others.

Table 46 on page 72 displays the information for the operating system section of the properties file.

Table 46. Operating System Section of Properties File

Field	Description
command.x.platform	Identifies the operating system for which this entry is valid. The following operating systems are valid: • Windows XP • Windows 2003 • Windows 2008 • Windows Vista • Solaris • Linux You can use wild cards (? or *) to specify multiple versions of an operating system. The file is processed from beginning to end until the first acceptable match is found. You can use the ? wild card to specify a one-character wild card and the * wild card to specify a multiple-character wild card.

command.x.run	Identifies the command used to run the executable file that is to run on this operating system. This executable file must be configured to run from the command line of the system on which it is to operate.
	
	For example, if a user's system has a Windows platform and the user must be in the c:\netscape directory to launch the Web browser, their properties file might define the browser.x.run field as:
	`browser.1.run = c: & cd \\netscape & netscape`
	
	This example uses the (&) to string command-line operations together, changing to the c:\netscape directory before issuing the netscape command. The & string concatenation technique is supported on Windows platforms only. You can use the batch file approach on other platforms which do not support stringing multiple command-line operations together.
	
	Notes:
	1. Notice that you must specify a double back slash (\\) in the command line. In these properties files, the back slash is treated as an escape character similar to the way a C compiler treats a back slash. So when you need a back slash character in your path name, use two back slashes.
	2. Avoid using environment variables, as they might not be resolved in the final command string.
	3. You might want to specify a directory greater than eight characters in length, such as Program Files in the following example:
	`browser.1.run = cmd /c start /Dc:\\Program Files\\netscape\\netscape.exe`
	
	To be sure you run the preferred application, enclose your full path name in quotation marks, prefaced with a back slash, as in the following example:
	`browser.1.run = cmd /c start /D\"c:\\Program Files\\netscape\\netscape.exe\"`
	
	You can also create a script file in a known directory and specify the script file as the executable file. Then the contents of the script file can be written to change to the proper directory and start the browser.

Table 46. Operating System Section of Properties File (continued)

Field	Description
command.x.args	Identifies the argument to be passed to the specified executable file. Anything enclosed in percent (%) signs is considered a substitution variable that can later be resolved by NetView management console or RODM, or passed in from the command response file. For example, the `%url%` substitution variable in the browser.x.arg field defines a specific Web address to be displayed when the browser is started. The specific Web address is defined in the command response file. The substitution variables are gathered from three locations: • in the command string keyword of the CPE response file, where the variable and value are defined as `var=value`. For example, see "Response File Input" on page 69 for the Web address variable. • `%ihs.`*`xxx`*`%` substitution variables. The character strings (*xxx*) following the period are the keys in the `IhsCmdInfo.java` and `IhsResInfo.java` objects. The substitution variable is the value in these objects. See the NetView Management Console Productivity Kit for more information. • `%RODM.`*`xxx`*`%` prefix substitution variables. The character strings (*xxx*) following the period are the field names in RODM for the selected objects. Use these only for resource specific commands. You can optionally specify a colon (:) and a token number to retrieve a single period-delimited token. For example, the `MyName` field of a NetView Resource Manager aggregate object in RODM is in the form: `NRM.CNM01.AUTO1`. If you want to extract just "AUTO1", you can specify: `%RODM.Myname:3%` This extracts the third period-delimited token.

The following example illustrates a properties file:

```
;----------------------------------------------------------------------
; Your comments go here.
;----------------------------------------------------------------------
browser.desc       = Open Web browser and show URL
browser.usebuiltin = false

browser.1.platform = Windows *
browser.1.run      = cmd /c start
browser.1.args     = %url%

browser.2.platform = *
browser.2.run      = netscape
browser.2.args     = %url%;---------------------------------------

ping.1.platform    = Windows *
ping.1.run         = cmd /c start ping -t
ping.1.args        = %d2cmdargs%

ping.2.platform    = Solaris
ping.2.run         = xterm -e ping -s
ping.2.args        = %d2cmdargs%

ping.3.platform    = *
ping.3.run         = xterm -e ping
```

```
ping.3.args        = %d2cmdargs%

;--------------------------------------

telnet.1.platform  = Windows *
telnet.1.run       = cmd /c start telnet
telnet.1.args      = %d2cmdargs%


telnet.2.platform  = Windows *
telnet.2.run       = cmd /c telnet
telnet.2.args      = %d2cmdargs%

telnet.3.platform  = *
telnet.3.run       = xterm -e telnet
telnet.3.args      = %d2cmdargs%
```

Notes:

1. The telnet definitions are shipped as a default in the properties file. You might need to override these definitions to customize the command for your operating system.
2. The `%d2 cmdargs%` variable is needed to support the **RunData2** command. The **RunData2** command uses the Remote Console support in RODM that might define the telnet command to run. See "%REMOTECONSOLE%" on page 134 for more information. The `d2cmdargs` variable is assumed to contain an IP address.
3. The telnet support in NetView management console also defines `%d2cmdargs%` when launching a telnet session. The telnet command appears in the right-click pop-up menu in a view. The `d2cmdargs` variable is assumed to contain an IP address.

You can define as many different application specifications as necessary. Number each one incrementally. For example, the previous example shows three specifications for telnet, numbered incrementally from 1 to 3. The number of specifications you can create is not limited; however, no numbers can be skipped.

Part 3. Using NetView Management Console

Chapter 9. Operating the NetView Management Console

This chapter includes the following information:
- Starting the topology server
- Starting the topology console
- Signing on from the topology console window
- Using the HOSTCMD command
- Stopping the topology server and topology console
- Retrieving inventory data on IP resources
- Writing server information to the NetView management console databases
- NetView functions available through NetView management console

Starting the Topology Server

The following sections describe how to start the topology server either manually or automatically. This task is usually completed by a system administrator for all topology console operators. If you do not need to start the topology server, skip to "Starting the Topology Console" on page 79.

Note: If you receive messages about the topology server, see the topology console help index for information about the messages.

Starting the Topology Server from the Desktop Icon

For Windows operating systems, double-click the **Start NMC Server** icon on the desktop.

Manually Starting the Topology Server

Perform the following steps to start the server and to activate the TCP/IP or LU 6.2 connection to the IBM Tivoli NetView for z/OS environment:

1. Open a workstation command window or shell prompt, depending on your operating system.
2. Change to one of the following directories:
   - For Windows: `%BINDIR%\TDS\server\bin`
   - For UNIX: `$BINDIR/TDS/server/bin`

   Note: BINDIR is an environment variable used by the Tivoli Framework to define path information used by Tivoli applications. For both Windows and UNIX, if the Tivoli Framework is not installed on the platform from which the topology server is run, the installation process installs the `setup_env.cmd` command file or `setup_env.sh` script file that defines `BINDIR`.

   For Windows, AIX, and Linux for System z, you might want to update the PATH environment variable to make running topology server commands from any directory in a command prompt more convenient.

   Depending on the path used during installation, the Framework for the topology server typically has one of the following `BINDIR` values:
   - usr\local\Tivoli\bin*interp*
   - local\Tivoli\bin*interp*
   - Tivoli\bin*interp*

In these BINDIR values, *interp* has one of the following values:
- For Windows, w32-ix86
- For AIX, aix4-r1
- For Linux for System z, linux-s390

3. Enter one of the following commands to start the topology server:
   - For Windows: tserver start
   - For AIX: ./tserver start
   - For Linux for zSeries®: ./tserver start

   Because Linux for System z has no display, perform the following steps before issuing this command to start the server:
   a. Telnet to the host.
   b. From the workstation used to perform the telnet, export the display to the workstation. To do this, use the following command:

      ```
      export DISPLAY=ip_addr:0.0
      ```

      where ***ip_addr*** is the IP address of the workstation.

   Notes:
   a. You must be the root user to start the topology server.
   b. On a UNIX system, if the topology server has been manually stopped and cannot be restarted, see Step 4 on page 90.

Starting the Topology Server as a Windows Service

If the topology server is configured to start manually, select the **Control Panel → Services** control applet:

1. Select **Topology Communication Server** and click **Start**.
2. Select **Topology Server** and click **Start**.

If the topology server is configured to start automatically, it starts when the system is started.

Starting the Topology Server as a Daemon

For UNIX, issue the following command to start the topology server processes at system startup and have them run as daemons:

```
$BINDIR/TDS/server/bin/config -d
```

Note: You must be the root user to issue this command.

Establishing Communication Between the NetView Host and the Topology Server

If communication between the NetView management console topology server and the NetView program is not active, you can issue the NETCONV command to start the communication. Usually this communication is defined to start automatically during NetView initialization. To start communication using TCP/IP, that was defined to start automatically but is not currently active, enter:

```
NETCONV STARTCON=ip_sysdef
```

To start communication using SNA, that was defined to start automatically but is not currently active, enter

```
NETCONV STARTCON=lu_sysdef
```

These NETCONV commands cause the connection to be established under the autotask that is defined by the function.autotask.NetConv statement in the CNMSTYLE member.

If you did not define communication to start automatically, you can use the NETCONV command with the ACTION=START keyword or with the STARTCON=*ip_sysdef* / *lu_sysdef* keyword to start communication under the task issuing the NETCONV command.

Note: If the task under which communication is established ends for any reason, the connection is broken. This is true for both the autotask that is specified in the CNMSTYLE member or the task under which the NETCONV command (with ACTION=START or STARTCON=*ip_sysdef* / *lu_sysdef*) was issued.

For more information about the NETCONV command, see the online help.

Starting the Topology Console

You can start the topology console by selecting a desktop icon in Windows operating systems, or you can start it by issuing the command in line mode from any operating system, or from the Tivoli Desktop.

Note: You can also launch the NetView management console from other applications. For information, see Appendix C, "Launching and Using the NetView Management Console from Other Applications," on page 163.

Selecting the Desktop Icon in Windows

For Windows operating systems, an icon is created on the desktop. To start the topology console using a Windows operating system, double-click the icon.

Using a Line Command

Change to the appropriate directory for your environment:

- For Windows: `\usr\local\Tivoli\bin\generic_unix\TDS\client\bin`
- For UNIX: `/usr/local/Tivoli/bin/generic_unix/TDS/client/bin`

From the directory, issue the appropriate operating system specific command to start the topology console:

- For Windows systems: `tconsoleNT .. -key nmc`
- For UNIX systems: `tconsole.sh .. -key nmc`

See Appendix B, "Topology Console Commands," on page 157 for complete information about all of the `tconsole` command-line arguments and supported environment variables.

Note: To avoid problems when starting the topology console on UNIX systems, add the directory that contains the **xhost** command to the PATH environment variable.

Using the Topology Console Sign On Window

Figure 12 on page 80 shows the topology console Sign On window.

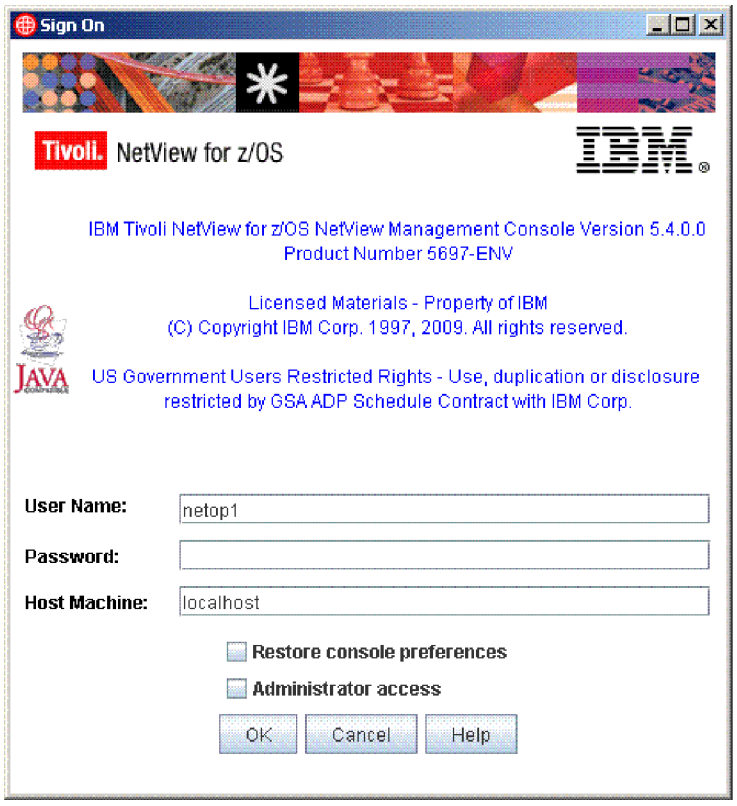


Figure 12. Sign On Window

The following steps describe how to sign on to the topology server.

1. Enter a user name in the User Name entry field of the topology console Sign On window.

   Notes:

   a. The user ID must match your NetView user ID.

   b. To restrict access to selected NetView operators to sign on to the topology console, use the **DUILOGON** command with the `NMC` keyword. This can be protected by command security (NetView command authorization table or SAF), and permitted to particular operators or groups. An example of this is commented out in the sample command authorization table CNMSCAT2 and the sample SAF command authorization table CNMSAF2. For more information, refer to the *IBM Tivoli NetView for z/OS Security Reference*.

2. Enter a password or password phrase in the Password field.

   This password or password phrase must match your NetView for z/OS password or password phrase.

3. Enter the IP host name of the topology server in the Host Machine field. If you reassign the topology server console port, specify the port number in this field the first time you sign on using this new port. However, the topology console retains the *server:port* setting on subsequent sign-ons and uses that setting as the default. See "Establishing Communication Between the NetView Host and the Topology Server" on page 78 for more information.

4. If you have previously signed onto NetView management console and saved the settings for the appearance of the topology console windows, you can select **Restore preferences**.

5. To use administrative functions, such as customizing settings and applying them to all topology consoles of this topology server, select **Administrator access**. More than one person can sign on with administrative access. In this case, one administrator can overwrite changes made by another administrator. The values saved last apply to all topology consoles.

   Note: For more information about authorization, see *IBM Tivoli NetView for z/OS Administration Reference*; specifically, see the information about the `NGMFADMN` keyword.

6. Select **OK** to sign on to NetView management console. The topology console window, as shown in Figure 13 on page 83 is displayed.

   Note: The first time you start the NetView management console, expect a delay as support files are downloaded to the topology console. On subsequent calls of the topology console, only files that have been updated on the topology server are downloaded.

7. If a later level of the NetView management console is available, a message is displayed asking if you want to update the code.
   - If you choose **Yes**, the sign-on window takes a short time while the latest level of code is downloaded. Then, the NetView management console is restarted and you can sign on again with the updated code.
   - If you choose **No**, signon is stopped.

Notes:

1. If you want to run commands, ensure that the `NGMFCMDS` keyword is set to *yes* (`NGMFCMDS=YES`). The default is `YES`. For more information about the `NGMFCMDS` keyword, see *IBM Tivoli NetView for z/OS Administration Reference*.
2. For more information about how files are automatically downloaded from the server at sign-on, see Appendix F, "Automatic File Download at Console Log On," on page 197.

Topology Console Window

Figure 13 on page 83 shows the topology console window. The following list describes the areas on the topology console window:

- In addition to the menu bar, the tool bar contains selectable icons that provide a quick way to perform the most commonly used functions.
- The *business* view shows all views. Each node represents a view. You can click on the plus (+) or minus (-) signs beside a node to expand or collapse the node. Double-click on a node to open that view.

  Note: The icons in the business tree display actual status only in certain cases. The **Business Systems** branch of the business tree displays accurate status. The other branches of the business tree display a satisfactory status, though this is not an accurate representation of the status of the resources they contain. When a node that represents a resource is dynamically added to the business tree as a result of more-detail navigation, the status represents that of the resource. It does not represent the aggregate status of all resources in the view.

- The *animation* icon rotates when the topology console is communicating with the topology server. To cancel a request that is still in progress, click the icon.

- The work space contains the view area and view filter bar. See “The View Area” on page 83 and “The Filter Bar” on page 84 for details about these two views on the work space. You can detach the work space to view several different views at the same time.
- The log contains messages, issued commands, and command responses. Although command responses can be received asynchronously, they are always displayed following the issued command.

  Note: The log is not displayed by default. To display the log, go to the Options menu and click **Show Log**.

  When the visible portion of the log is full, the log begins to automatically scroll. To change this option so that you can manually scroll the log, right-click the log window and deselect **Automatic Scrolling**.

  To save the log, go to the File menu and click **Save Log**. You can save the log to a file on the server or on the console. If you click **Save Log to Console**, a dialog box is displayed so that you can specify the name and location of the file. For additional information, refer to the online help for the menu choices.

  Note: To clear the console log, use the context menu.
- The view bar displays buttons for all open views in the order they were opened. You can click any of the buttons in the bar to re-display a previously opened view.
- The status area displays the following information:
  - The text about the resource or business tree item under the mouse, or the status of any action in progress
  - The name of the topology server connection, your host name, the sign-on time and date, the mode you are currently using, NETCONV status, and the topology console IP address
  - Information about the displayed view

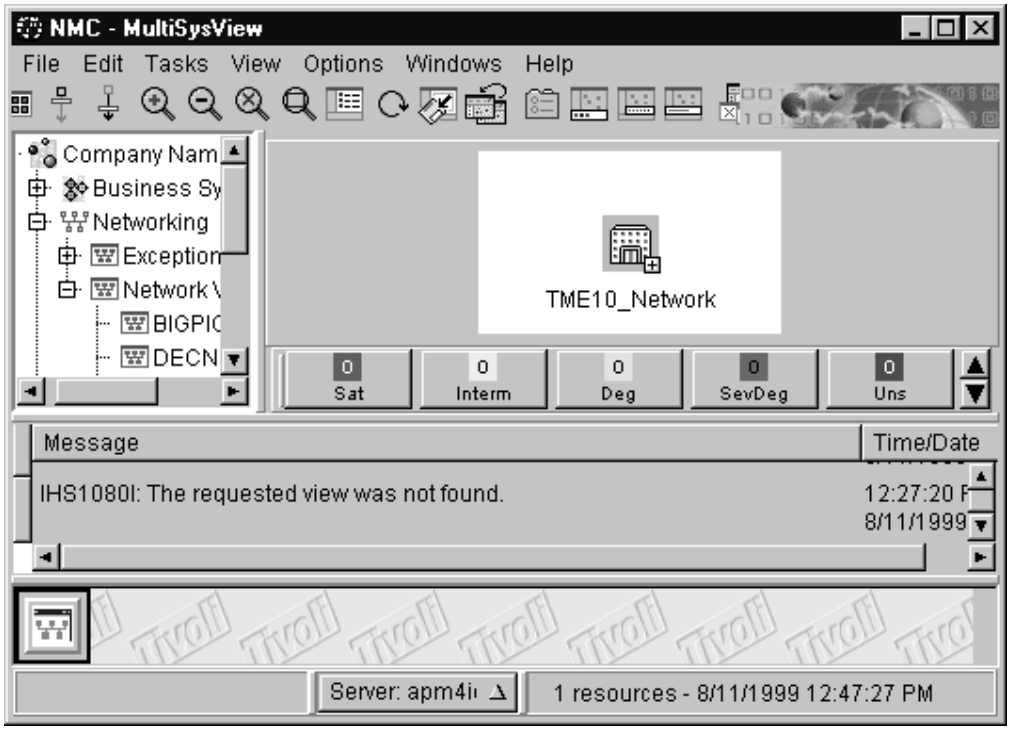


Figure 13. Topology Console Window

The View Area

The *view area* of the topology console window contains a notebook that displays different types of views of your resources. Resources can be displayed in topology or details views.

- *Topology views* display resources and their statuses in a graphical format. Topology views are especially useful in displaying the relationships between resources; for example, displaying connectivity. Resources in a topology view are often placed on a background image, such as a building map or geographical map, which shows their general location and proximity to each other.
- *Details views* display resources and their statuses in a list format. Details views contain additional information about resources, such as type and description. You can also use them to control the order that resources are displayed, such as by time stamp or status. Null nodes and null links are never displayed.

Within a view, context menus contain various commands and functions available for a resource. To access a context menu, right-click a resource. A menu is displayed with a list of the available actions for that resource. To access a context menu for the view, move the mouse pointer to any area of the view that is not over a resource, then right-click. (If your details view is full, move the mouse pointer to any column heading, and then right-click.) A menu that shows the available actions for the view is displayed.

You can print a topology or details view. Select **File** from the menu bar and then select **Print** from the pull-down menu to print a view. Note that you cannot zoom in on a topology view and print it. To print a larger picture of a view, resize your window and then print.

The Filter Bar

The *filter bar* displays the number of resources in the current view at each status or by flag set. This status is indicated in views by the color of the resource.

Note: The filter bar is not displayed by default. To display the filter bar, select **Options → Show View Filter**.

You can use the filter bar to prevent resources with particular statuses from being displayed in the current view. To filter resources with a particular status, click on the corresponding button to that status on the filter bar.

- In topology views, filtered nodes are displayed as empty boxes and filtered links are displayed as dashed lines.
- In details views, filtered resources are not displayed.

The filter bar continues to display the number of resources in each status. To temporarily re-display a filtered resource in a topology view, including its status, icon, and so on, position your cursor on the resource without clicking.

You can customize the statuses that are displayed on the filter bar in the Status page of the Console Properties notebook.

NetView Management Console Online Help

NetView management console provides the following kinds of online help from the topology console:

- Menus
- Dialog boxes and notebook pages
- Windows
- Overview
- Messages for the topology server and topology console
- Commands
- A Legend function that displays a window containing descriptions of all resource that are types defined to the topology server.
- The ability to display a Web site
- Help Index, which you can use to search the help facility. Use a task index to find help on specific tasks. If you receive messages about the topology server, look in the Help Index for information about the messages.

NetView Management Console Functions

This section describes some of the functions of the NetView management console.

Issuing IP Commands

The following IP commands can be issued from the NetView management console interfaces:

- Ping
- Tracerte (trace route)
- Netstat (Network host status)

You can issue the available IP commands from the NetView management console topology console screen in one of two ways:

- Right-click a resource and select **IP Commands** from the context menu. This is available for any resource that contains an IP address.
- From the view background, right-click the background and select **IP Commands** from the pop-up menu.

From the **IP Commands** menu, the following items are available:
- onetstat
- ping
- tracerte

When you select ping, onetstat, or tracerte, a dialog box is displayed. If the menu was accessed from a resource and you select ping or tracerte, the IP address for that resource is inserted into the Host Machine field of the dialog box. The ping command can also be started without opening a dialog box when the menu is accessed from a resource.

The NetView Resource Manager

Use the NetView Resource Manager to manage all NetView programs in an enterprise with the NetView management console. The NetView Resource Manager collects task information and forwards this information to a *manager* NetView for processing. The processed information is stored in RODM.

Use the NetView management console interface and the NetView Resource Manager to build views to monitor your enterprise. After RODM is populated, you can use the NetView management console to monitor the NetView program. The views created by the NetView Resource Manager are network views represented in the view tree as NetView Task views. Selecting a NetView Task view results in a view of NetView domains. This view of NetView domains is the first NetView Resource Manager network object that can be selected from the NetView management console and it is an aggregate. Selecting **More Details** from this view opens a view containing aggregate objects which represent NetView tasks. Each NetView task aggregate can then contain real objects representing the following items:
- CPU
- STG
- MQIN
- MQOUT
- MSGCT
- IO
- status (active/inactive)

Sample views are provided to monitor and manage NetView tasks.

If you want information about...	Refer to...
Sample views	*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*
Using the NetView Resource Manager	*IBM Tivoli NetView for z/OS User's Guide: NetView* and *IBM Tivoli NetView for z/OS Installation: Getting Started*

Using the RODM Collection Manager With NetView Management Console

The RODM Collection Manager enables arbitrary grouping of objects into views or aggregates. Unlike BLDVIEWS which are static, the RODM Collection Manager dynamically manages the views or aggregates. This means that the RODM Collection Manager is continually updating the views or aggregates, enabling you to add, change, or delete collections. To access the RODM Collection Manager from the NetView management console, you must be signed on as an administrator.

Figure 14 shows the RODM Collection Manager main menu.

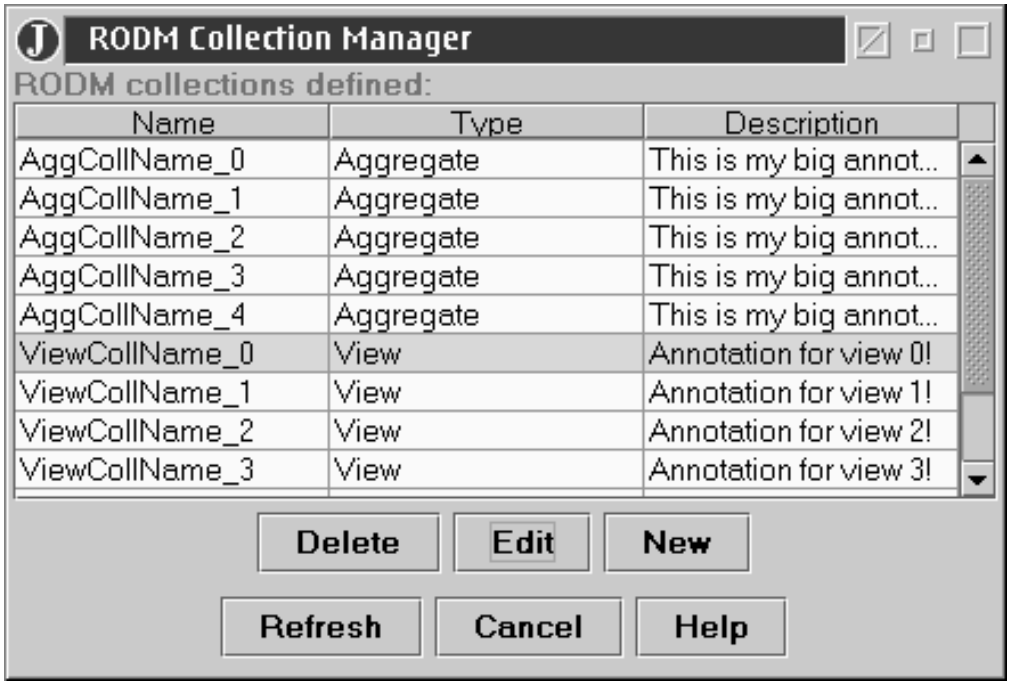


Figure 14. The RODM Collection Manager main menu

NetView Management Console Topology Server Databases

The topology server databases are used to save server information between restarts of the server. The server information contained in these databases includes the following kinds of data:

- Resource data for all resource types
- Operator data for all operators that have logged on
- View data for all saved views
- Command data for all customized commands

When the topology server starts, it loads the data from the first server database directory that contains a database that can be loaded. Table 47 lists the topology server databases, in the order that they are searched for databases that can be loaded.

Table 47. Topology Server Databases

Name	Location
Current	For Windows: `%BINDIR%\TDS\server\db\current\datab` For UNIX: `$BINDIR/TDS/server/db/current/datab`

Table 47. Topology Server Databases (continued)

Name	Location
Backup	For Windows: `%BINDIR%\TDS\server\db\backup\datab` For UNIX: `$BINDIR/TDS/server/db/backup/datab`
Custom Backup	For Windows: `%BINDIR%\TDS\server\db\custom_backup\datab` For UNIX: `$BINDIR/TDS/server/db/custom_backup/datab`
Default	For Windows: `%BINDIR%\TDS\server\db\default\datab` For UNIX: `$BINDIR/TDS/server/db/default/datab`

The default database contains the initial (default) server information. When the server is started for the first time after installation, the default database is loaded. The current and backup databases do not contain server information when the server is loaded for the first time after installation.

The current and backup databases contain the most recent copy of the server information after the server is stopped or checkpointed. Generally, the current and backup databases contain identical server information, but, in some cases, only the current database is updated with server information. For more information, see "Writing Server Information to the Topology Server Databases."

The custom_backup database is used to save a customized version of the server information at a known level of customization. This database is not continuously updated by stopping or checkpointing the server. Use this database to protect your server information from undetected database corruption.

Writing Server Information to the Topology Server Databases

Writing server information to the topology server databases is also known as checkpointing. Server information is written to the topology server databases either manually or automatically.

The default database is never written during a database checkpoint and serves as a starting database if one or all of the other databases become corrupted. The other databases are written based on the type of checkpoint requested. Server information is written to the topology server databases automatically for the following reasons:

- The autoCheckpointInterval or autoCheckpointDaily properties of the `server.properties` file are enabled. These properties are explained in the `server.properties` file. For more information about this file, see "Modifying the Server Properties File" on page 19. All server information is checkpointed to the current database. If the checkpoint completes successfully, the current database is then copied to the backup database.

  Note: The default setting for these properties result in automatic checkpoints at 1 a.m. every day.
- A view is customized and saved to the topology server. Only the files that were changed for the view customization are written, and they are written only to the current database.
- The command profile editor batch utility is run. Only the files that were changed for the command customization are written, and they are written only to the current database.
- The topology server is shut down. All server information is checkpointed to the current database. The backup database is not copied until the server is restarted.

If the current database is successfully loaded during the subsequent server startup, the current database is then copied to the backup database.

To manually write information to the topology server database, use either the `tserver utility -c` command or the `tserver utility -cc` command.

The `tserver utility -c` command manually checkpoints all server information to the current database. If the checkpoint completes successfully, the current database is then copied to the backup database.

The `tserver utility -cc` command manually checkpoints all server information to the custom_backup database. The custom_backup database is used to save a customized version of the server information at a known level of customization. The custom_backup database is meant to override the default database if the current and backup databases become corrupted. For more information about how to use the custom_backup database, see "Creating and Importing Backup Copies of Customized Topology Server Databases" on page 89.

If your installation has a large number of customized views or commands, it is recommended that you use the `tserver utility -cc` command whenever you make significant changes to the customized views or commands. When you have used this command, you can restore the server information to a known good copy of the database that contains your recent customization if the current and backup databases become corrupted.

Handling Corrupted Topology Server Databases

One or more of the topology server databases can become corrupted, causing the server to fail to start or to behave abnormally after it starts. Corrupted databases can cause either or both of the server processes (the topology data server and the topology communications server) to end.

Topology server databases can become corrupted for the following or other reasons:

- The topology server process is incorrectly stopped; for example, stopping a topology server process by closing the topology server window instead of using the `tserver stop` command. Abruptly stopping the topology server in this manner prevents it from properly updating the databases before it stops.
- The file system used by the topology server runs out of space, preventing the topology server from updating its databases.
- The topology server encounters an internal failure, which results in the topology server stopping abnormally.

The default and custom_backup databases are not likely to be corrupted. Most database corruptions are the result of a corruption in the in-storage copy of the database. Because the in-storage database is written only to the current and backup databases, the default and custom_backup databases can be used as your recovery databases.

If you suspect a database corruption problem, perform the following steps:

1. If the server is running, stop the server.
2. Make a backup copy of the current and backup databases. You might want to recover these databases if the problem is not a database corruption problem.
3. Restart the topology server. The topology server attempts to detect database corruption on initialization. If it detects database corruption, the server

attempts to restore the database from the next database that can be loaded. See Table 47 on page 86 for the database recovery order.

4. If the server successfully starts, then your database has been recovered. If the server does not start, continue with the next step.
5. Erase the current database.
6. Restart the topology server. This automatically copies the backup database to the current database. If the server successfully starts, then your database has been recovered. If the server does not start, continue with the next step.
7. Erase both the current and backup databases.
8. Restart the topology server. At this point, the database has been recovered either from the default database or, if you are using a custom backup, from the custom_backup database.

If the topology server successfully starts, then the problem was that the original contents of both the backup and current database directories were corrupted. If you previously saved a copy of the databases, you can optionally use the saved copy to restore the topology server databases, as described in "Creating and Importing Backup Copies of Customized Topology Server Databases."

If the topology server does not start, then the problem is not because of corrupted databases. To continue problem determination, see the *IBM Tivoli NetView for z/OS Troubleshooting Guide* for information about diagnosing NetView management console and Graphic Monitor Facility host subsystem (GMFHS) problems.

Creating and Importing Backup Copies of Customized Topology Server Databases

If you plan to customize many views or commands, you should create a backup copy of the database to insure that your customization is not lost if the server database becomes corrupted.

Customization recommendations:

- Perform customization first on a test, or non-production, server.
- Start (or restart) the server before performing any customization. Having a freshly started server reduces the risk of any corruption to the in-storage database; a server that has been active for many weeks or months can have an in-storage corruption that has not yet been detected.

After you perform the customization, create a backup copy of the server databases that contain the customization using the following steps.

Note: This procedure requires stopping and restarting the server.

1. To manually write (checkpoint) the topology server databases to the custom_backup directory, enter the following command and wait for it to complete successfully before proceeding to the next step:

   ```
   tserver utility -cc
   ```

2. Stop the server.
3. Delete all the files in the current database directory, but do not delete the directory itself.
4. Copy all the files from the custom_backup database directory to the current database directory.
5. Start the server. Verify that the server does not issue an error message indicating that the database is corrupted.

To import the custom_backup database directory to other servers, perform the following steps:

1. Stop the server to which the database is to be imported (the import server).
2. Copy the custom_backup directory from the test server to the custom_backup and current directories on the import server.
3. Restart the import server.

Stopping the Topology Console

To stop the topology console, select **File → Exit** from the topology console window. The topology console and all windows are closed.

Stopping the Topology Server

The procedure for stopping the topology server depends on the platform on which the topology server is running. For Windows, you can stop the topology server by selecting an icon or entering a line-mode command. For AIX, you can stop the topology server only with a line-mode command. The following sections describe the steps for stopping the topology server on each type of platform. For Linux for zSeries, you can stop the topology server one of two ways: with a line-mode command or by clicking the **X** in an Xwindow if the server was started with `tserver start`.

Attention: Do not stop the topology server by clicking the **X** in the top-right corner of the window unless the topology server is running on the Linux for zSeries platform and was started with `tserver start`.

With the Service Version on Windows

To stop the topology server in this environment, select the **Control Panel → Services** control applet, or enter the **ihsxsrv stop** command from the `%BINDIR%\TDS\server\bin` directory on the topology server.

Using a Line-mode Command

To stop the topology server when it is running without problems, complete the following steps:

1. Open a workstation command window.
2. Change to one of the following directories:
   - For UNIX: `$BINDIR/TDS/server/bin`
   - For Windows: `$BINDIR%\TDS\server\bin`
3. Enter the **tserver stop** command to stop the topology server.

   The **tserver stop** command, with no flags, stops the topology server. For UNIX, only root users can stop the topology server.
4. For UNIX only, if the topology server is suspended, or if a daemon has trapped, issue the **tserver stop** command a second time with the `-f flag` in the following way:

   ```
   tserver stop -f
   ```

   The **tserver stop -f** command forces running daemons to stop, and cleans up any remaining inter-process communication resources that were used by the daemons.

Chapter 10. MultiSystem Manager Operation

This chapter explains how to initialize MultiSystem Manager, start and stop MultiSystem Manager processing, and use the views and command support to manage your networks.

Note: For specific information about the Open topology agent, see Chapter 11, "Open Topology Interface Network Operation," on page 105. For specific information about the Tivoli Management Region agent, see Chapter 12, "Tivoli Management Region Operation," on page 111. For specific information about the MultiSystem Manager IP and IBM Tivoli Network Manager agents, see *IBM Tivoli NetView for z/OS IP Management*.

Getting Started

Before using MultiSystem Manager, ensure that:

- You have completed the IBM Tivoli NetView for z/OS installation steps listed in the *IBM Tivoli NetView for z/OS Installation: Configuring Graphical Components*.
- You have created or modified the MultiSystem Manager initialization statements as described in *IBM Tivoli NetView for z/OS Installation: Configuring Graphical Components*.
- RODM is active, and the MultiSystem Manager data model is loaded.
- GMFHS and the NetView management console are active and operational for displaying graphic views.
- The NetView RUNCMD task (DSIGDS) is active and operational to support RUNCMDs if you are using SNA sessions. TCP/IP must be operational if you are using it.
- The NetView SAVE/Restore task (DSISVRT) is active and operational to support the GETTOPO heartbeat function.
- Autotask AUTOMSM is active.
- The NetView hardware monitor is active.
- The NetView automation table is active.
- The event receiver (IHSAEVNT) is active and configured properly. This is used by the MultiSystem Manager IP agent to receive *traps* from the Tivoli NetView program (or a similar product) and *events* from the MultiSystem Manager Tivoli Management Region agent.
- For the MultiSystem Manager IBM Tivoli Network Manager agent, the NetView trap receiver task is active.

Using MultiSystem Manager Views

After you initialize network topology and status in RODM, your NetView management console provides views of your MultiSystem Manager networks.

Using the NetView management console pull-down menus, you can navigate among the views to isolate failures and send commands to resolve problems. The following view navigation information is common to all MultiSystem Manager features.

Note: For navigation information for the Open topology agent, see Chapter 11, "Open Topology Interface Network Operation," on page 105. For navigation information for the Tivoli Management Region agent, see Chapter 12, "Tivoli Management Region Operation," on page 111. For navigation information for the MultiSystem Manager IP and IBM Tivoli Network Manager agents, see *IBM Tivoli NetView for z/OS IP Management*.

Navigating Views

Common view navigation options are available across all MultiSystem Manager features. From an aggregate object, the **More Detail** and **Configuration->Child** navigations display resources that are contained in the aggregate object. **Resource->More Detail** shows just the immediate child resources. **Resource->Configuration Child** shows the immediate child resources and any lower-level (grandchild) resources. The **Configuration->Parent** navigation is also supported. This navigation option shows the selected object connected to all aggregate object parents of that object.

To identify where a real network resource is managed by more than one topology agent, operators can select **Configuration->Parent** (on the context menu of the network resource object). Operators can use this to display a view that contains the parent of the resource or the path from a real resource to the host that owns it. This is useful for navigating between managed networks. It is also useful in identifying cases where your systems administrator has defined two agents to manage the same network resource.

Operators can also select **Configuration->Logical** and **Configuration->Physical** (on the context menu of the network resource object) to view the logical and physical relationship between resources. However, dependent upon the configuration, these navigations might only re-display resources in the current view.

As you navigate through your views, you can select any object and get resource information about that object. Each Resource Information window has the following fields:

Resource name
Name of the selected object.

Other data
Detailed information unique to the selected object.

Type Class of view object representing the resource.

Customer data
Reserved for customer use.

Managed by
Name of the MultiSystem Manager topology manager that manages this object.

System status
Status of the selected object and time status was last updated.

Operator status
Status initiated by operator, for example, if aggregation is suspended.

Aggregation priority (real objects only)
The number of exception status levels to be reported.

Total resources (aggregate objects only)
The number of resources contained in this aggregate.

Unsatisfactory/Exception resources (aggregate objects only)
The number of resources contained in this aggregate that have either unsatisfactory or exception status.

Degraded (aggregate objects only)
The number of resources that must report exception status before the status of this aggregate is changed to *degraded*.

Severely degraded (aggregate objects only)
The number of resources that must report exception status before the status of this aggregate is changed to *severely degraded*.

Unsatisfactory (aggregate objects only)
The number of resources that must report exception status before the status of this aggregate is changed to *unsatisfactory*.

Topology Correlation Across Different Types of Networks

The IBM Tivoli NetView for z/OS program uses topology correlation to automatically tie together resources managed by different types of topology features such as IP and Tivoli Management Region. Topology correlation is provided for all MultiSystem Manager topology features, the NetView SNA Topology Manager, and customer or vendor applications that use the GMFHS data model.

Most managed resources participate in more than one network or system. For example, a workstation can serve as a station in an SNA network and a Windows platform, and be present in the IP environment. Most management agents recognize only one type of network or system, as opposed to every resource running in the system. The IP agent, for example, only recognizes resources with an IP addresses. Therefore, an IP view of the example workstation contains information related to IP, but not the token ring adapter and Windows platform.

Topology correlation overcomes the agent-specific perspective of management agents and can be used by a NetView management console operator to view all components of a network device. This single view includes the current status of all components.

Topology correlation is accomplished using a network address or a customer-defined free-form string value (for example, `Accounting` or `Australia`).

Network topology correlation automatically creates or updates correlated aggregate objects that can contain the following resource objects, if the corresponding topology feature is running for that type of resource:

- IP Interfaces
- Vendor or customer created Open Node and Node aggregate resources
- Tivoli Management Region Monitor resources
- SNATM PU resources
- GMFHS managed real resources
- TN3270 server and client resources

You can include any resource in topology correlation by setting a free-form correlator value. This is useful when a resource is not automatically correlated because no IP address is discovered for it. It is also useful when you want to include managed objects in views that you define based upon organizational structure or geography. For example, if a customer or partner application specifies Australia for the field value on different managed resource objects in RODM, all of

those objects are contained in one aggregate object with a display name of Australia. This object has a display type of Open System Aggregate.

You can also use support for multiple values, in the free-form Correlator field, to create or locate a hierarchy of correlated aggregate objects that match your organizational structure. Figure 15 is an example.

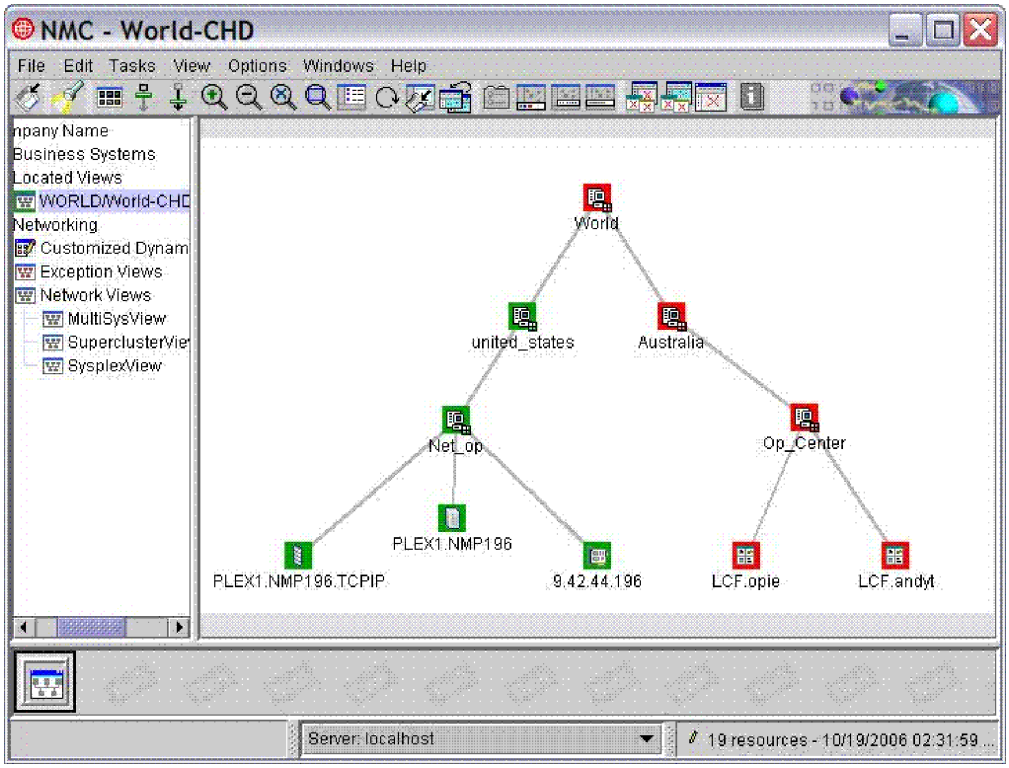


Figure 15. Correlated Aggregate Objects Example

If a topology feature has both a free-form correlator string and a TCP/IP address for a managed resource, topology correlation typically creates just one correlated aggregate object and connects the resource to it.

You can use topology correlation to isolate a failure that one managed system caused in a separately managed system. For example, operators can see that a TCP/IP failure caused a workstation managed by a Tivoli Management Region to fail; they can then solve the problem in the IP network without further investigation of workstation software.

Topology correlation is an optional function that discovers correlations dynamically through RODM methods. To enable this function, your system administrator must include the FLCSDM8 file in the RODM structure load. Including SNA resources in correlation also requires a change to the FLBSYSD file. See the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* or the FLBSYSD file.

You can override correlation default display characteristics and you can include additional RODM objects in network or free-form topology correlation. All customization can be done in the FLCSDM8 RODM load file. See the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for customization information.

Viewing Correlated Resources

To view correlated resources, select a real resource object, then select **Configuration -> Parent**. The resulting NetView management console view, Figure 16, shows both the navigation hierarchy for the topology feature of that object and a branch to the correlated aggregate in which the object is contained.

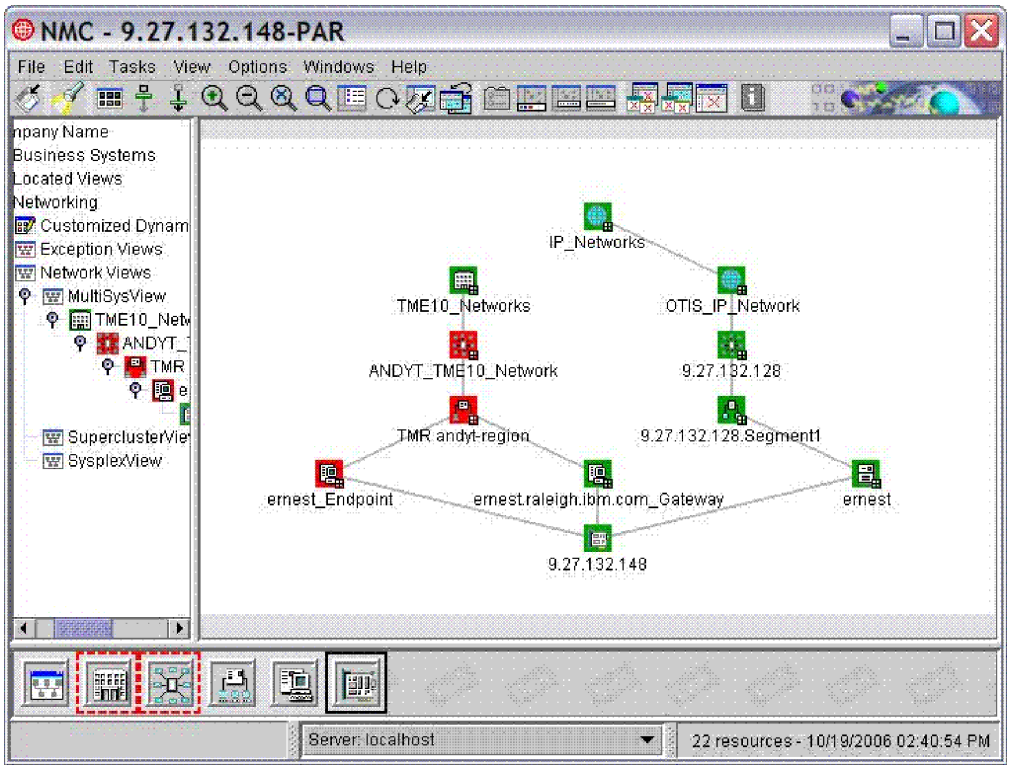


Figure 16. Configuration->Parent Example

Figure 16 shows how an IP interface is part of `IP_Networks`, but it is also contained in a workstation aggregate that is part of the Tivoli Management Region. In this example, the workstation has both an endpoint and a gateway presence.

In your views, correlated aggregate objects created or located by network address have an object type of *LAN Workstation aggregate* or *IP System aggregate*. If a correlated aggregate object is created or located by free-form correlation, the object is given an object type of Open System Aggregate. To see the resources contained in a correlated aggregate object, select the aggregate object and then select **More Detail**. The status of the correlated aggregate object reflects the combined status of the resources contained within.

For example, the NetView management console view in Figure 17 on page 96 shows all the resources contained in workstation `andyt`.

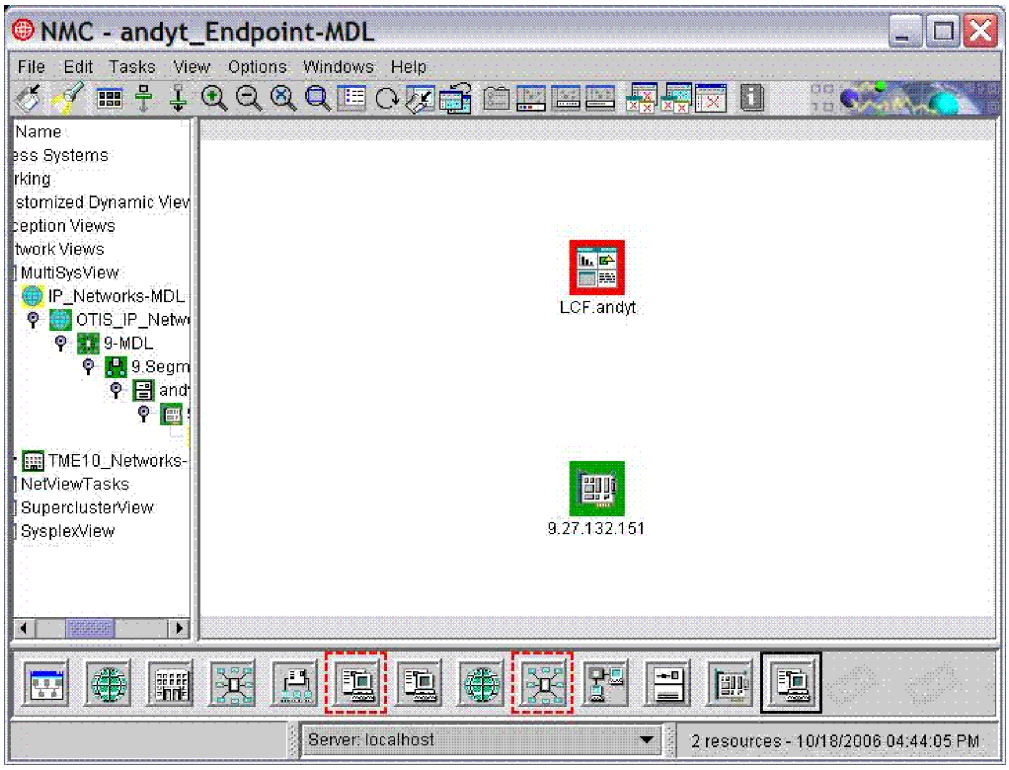


Figure 17. Correlated Resources Example - More Detail

The workstation (andyt) contains resources discovered by the IP Networks and the Tivoli Management Region agents.

If you select a correlated aggregate object, you can display the contained resources and the aggregate by selecting **Configuration Child**. The NetView management console view shown in Figure 18 on page 97 is the result.

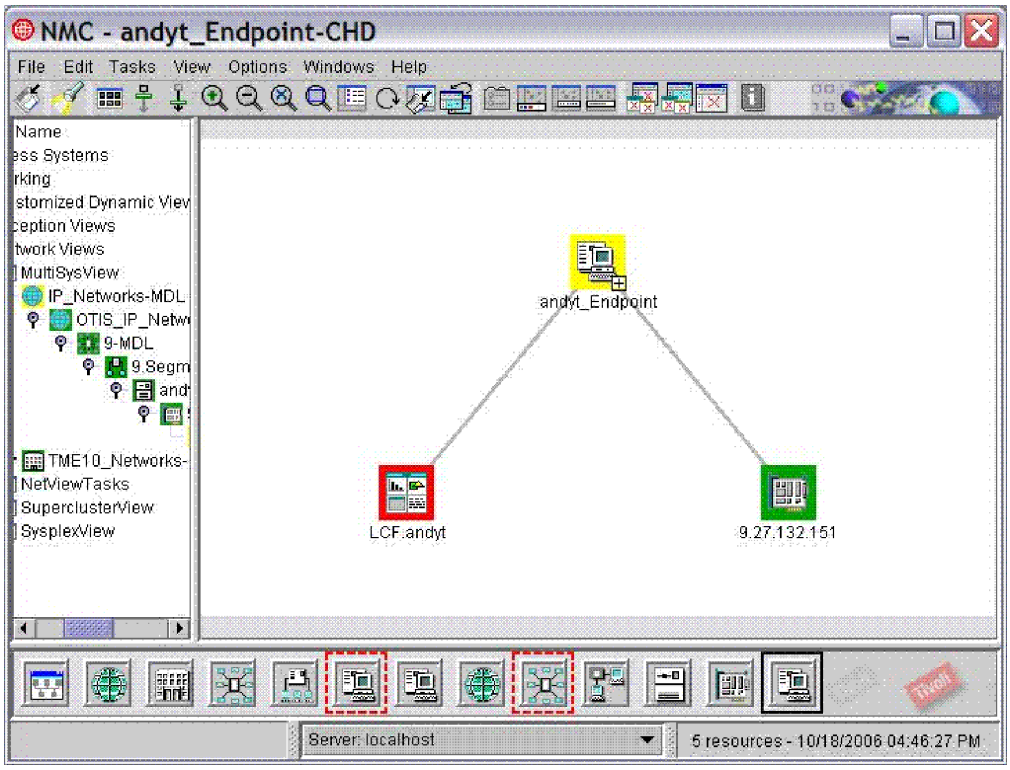


Figure 18. Correlated Aggregate and Resources Example

Workstation Aggregate Objects Created by MultiSystem Manager Agents

The MultiSystem Manager Tivoli Management Region feature can manage workstations and selected resources contained in those workstations. Other MultiSystem Manager features manage network resources, some of which might be contained in a workstation. Because of these differing objectives, the Tivoli Management Region feature creates correlated aggregates and displays them in the More-Detail NetView management console navigation hierarchy (inline); the other features do not. For the other features, if topology correlation has been loaded, you can access correlated aggregate by selecting **Configuration->Parent** from the context menu of a real resource.

Customization Options

Figure 15 on page 94 illustrates the result of free-form correlation of a multi-valued string. This string can be used to locate or create a hierarchy of correlated aggregate objects that match an organizational structure. When you set the Correlator field value on real RODM objects that are created and linked to the correlated aggregate objects in Figure 15 on page 94, the Correlator field values set on the real RODM objects are shown in the following example:

```
Set PLEX1.NMP196.TCPIP  Correlator = 'Net_op united_states World'
Set PLEX1.NMP196        Correlator = 'Net_op united_states World'
Set 9.42.22.196         Correlator = 'Net_op united_states World'
Set LCF.opie            Correlator = 'Op_Center Australia World'
Set LCF.andyt           Correlator = 'Op_Center Australia World'
```

Topology correlation, by default, displays each correlated aggregate by its workstation name, if available. This is the same name that the MultiSystem Manager Tivoli Management Region feature uses to name workstations. If a

workstation name is not available, the next available name type on the Display Name Priority list is selected. However, you can select a different name type to display for all correlated aggregates by customizing the RODM Loader file FLCSDM8. Refer to the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for the available customization options. Refer to the customization instructions in the FLCSDM8 file for customization instructions. Table 48 shows the default display name priority.

Note: If you alter the display name priority in file FLCSDM8, you might cause a discrepancy between the name types that are displayed in the NetView management console and the name types that are displayed in example views in this document for Tivoli Management Region.

Table 48. Correlated Aggregate Object Default Display Name Priority

Priority	Name Type	Discovered by
1	Computer Name	Tivoli Management Region features
2	IP Host Name	IBM Tivoli Network Manager, IP, and Tivoli Management Region features
3	TCP/IP Address	IBM Tivoli Network Manager, IP, and Tivoli Management Region features
4	SNA Node Name	SNATM

If a resource is correlated to an aggregate based upon a free-form value, the display name for the aggregate is the free-form string value.

The topology correlation method also enhances your ability to locate a correlated aggregate using the NetView management console *locate* function. Operators can locate a correlated aggregate using a value for any of the following network addresses, dependent upon what is discovered in your network:

- LAN MAC Address
- IP Address
- SNA PU or LU
- IP Host name
- Free-form correlator string

Setting Free-Form Correlation Values

You can set free-form Correlator field values with point-and-click menus using Visual BLDVIEWS running on a workstation. You can also set Correlator field values, using RODMView, by creating a command list (CLIST) or by creating a BLDVIEWS script. For more information, see the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

Using Visual BLDVIEWS has several advantages. You can use it to point-and-click to select resources to be included in correlation. To do this, you do not need to understand RODM or BLDVIEWS syntax. Visual BLDVIEWS creates a valid BLDVIEWS script. BLDVIEWS typically includes objects in views if those objects have a consistent naming scheme (for example, `CPNRTR2` and `CPNHST14`). Views are built, hierarchically, from the top down.

Multiple free-form correlation does not require objects to have a similar object naming. It builds views from the bottom up. Using BLDVIEWS and Topology correlation together, it is easier to build custom views to match your enterprise. To

set a Correlator field value on an object in Visual BLDVIEWS, select your target object in the left side of the Visual BLDVIEWS menu and drag it to the Build/Set Resources area on the right side of the menu. This is illustrated in Figure 19.

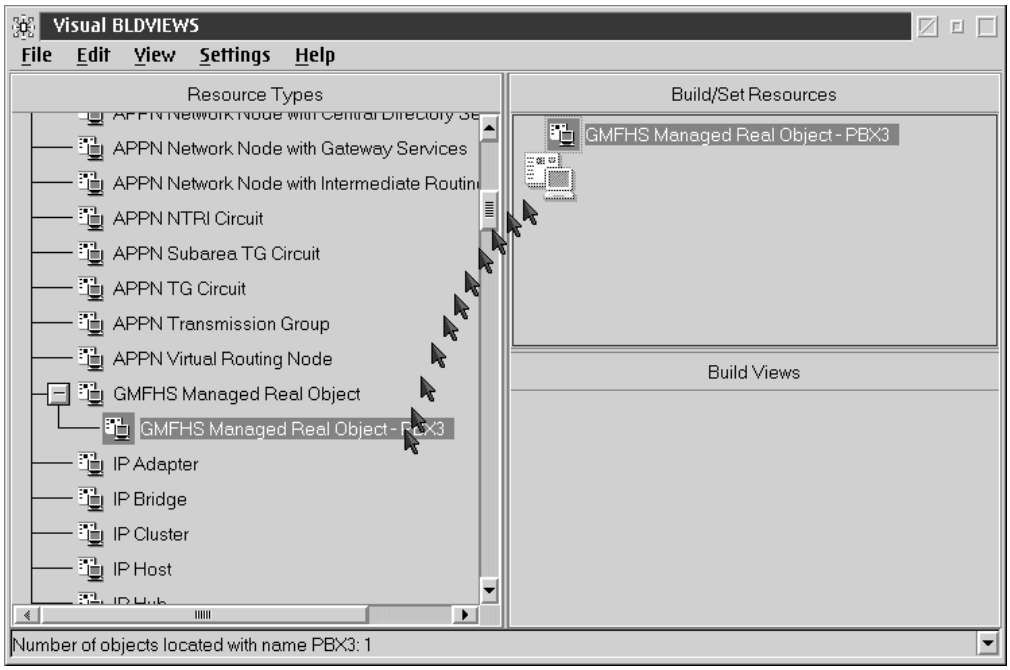


Figure 19. Setting a Correlator Field Value

After you drop the object into the Build/Set Resources area, double-click the object to bring up the settings dialog box for it.

Fill in the Correlator field. You can set a single string value or a multi-valued string, delimited by blank spaces or commas. After you have set correlator values on all the managed objects that you want to include in free-form correlation, save your requests to the host and run them, as shown in Figure 19. Specify a file name and location to save the BLDVIEWS script that are generated. If RODM is recycled, you can restore your custom correlations by issuing this script from Visual BLDVIEWS or from BLDVIEWS on z/OS. If you want to modify the script, you can edit it from Visual BLDVIEWS or z/OS/TSO.

Issuing Commands

MultiSystem Manager provides an easy-to-use command interface based on the NetView command support that you can use to select objects from your NetView management console views and issue commands to these objects without requiring you to enter the name or address of the object.

Generic commands (such as Current Status, Activate, Inactivate and Recycle) are not implemented by MultiSystem Manager. They can be implemented using the BLDVIEWS utility. For more information about BLDVIEWS, see the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

For example, to issue a command to an IP system:

1. Right-click on the system and select **IP resource specific commands**.
2. Select the command you want to issue.

Note: Responses to commands you issued using the command support are returned to the Command Responses window available from your NetView management console view under the **Options** pull-down menu. To display the window, select **Show Log** from the **Options** pull-down menu. To display the entire log, select **Tear Away Log** from the **Options** pull-down menu.

For more information about messages that are displayed in the NetView management console Command Responses window, see the online help.

Resolving Network Problems

To resolve network problems, first locate the object in your view that is having the problem, as described in "Finding a Failing Object." Then understand the meaning of the status of the object as described in "Understanding View Object Status" to diagnose the problem and take the appropriate action.

Finding a Failing Object

Suppose you are monitoring a MultiSystem Manager networks view, and you notice the network aggregate turns yellow, indicating a problem. To find the problem, you can do one of the following actions:

- Select the yellow aggregate object and then select **More Detail** and repeat this procedure until you find the failing resource.
- Select an object and then select **Locate Failing Resource**. This function automatically builds a detailed view showing all real objects under the aggregate object that have unknown or unsatisfactory status.

For information about locating failing resources in your views, see "Locate Failing Resources" on page 12.

Finding the Status of an Object

To view alert information about the NetView management console, right-click on the object and select **Event Viewer**. The event viewer window shows the alerts received for that object.

Understanding View Object Status

The status for both aggregate and real objects is displayed. The status of a real object reflects the operational capability in the network for that object. Table 49 shows a general description of real object status.

Table 49. Real Object Status Summary

Status	Meaning
Satisfactory	Resource is fully operational.
Intermediate	Performance of resource is degraded.
Unsatisfactory	Resource is not operational.
Unknown	MultiSystem Manager cannot communicate with the resource, and therefore, does not know the current status of the resource.

Setting Status Aggregation Thresholds

The status for an aggregate object is derived or aggregated from the real resource objects below it. The aggregation algorithm is based on the degraded, severely degraded, and unsatisfactory thresholds for each of the aggregate objects.

Setting aggregation is important. If the thresholds are set too high, failures are not seen on the high-level views that you are monitoring. If the thresholds are set too low, minor failures cause the status of the aggregate objects to have unsatisfactory status, which can hide major failures.

MultiSystem Manager sets default thresholds for each object type. You can use the defaults, or you can modify these thresholds by using NetView management console workstation menus or by changing the value of the threshold in RODM. See the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for information about the GMFHS aggregation algorithm for aggregate objects.

Note: You can also set aggregation thresholds by using the view customization utility, BLDVIEWS. For information about BLDVIEWS, see the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

Setting Aggregation Priorities

Aggregation priorities reflect the relative importance of a real object in your network. MultiSystem Manager sets a default aggregation priority for each type of real object in your view. You can use the defaults, or you can modify these aggregation priorities. To modify the aggregation priorities, use NetView management console workstation menus or change the value of the aggregation priority value in RODM. See the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide* for information about the GMFHS aggregation algorithm for real objects.

Note: You can also set aggregation priorities by using the view customization utility, BLDVIEWS. For information about BLDVIEWS, see the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

Updating Topology and Status

After the initial topology and status are stored in RODM, you can extend the set of managed resources by gathering topology and status for resources that were not previously managed. Use the GETTOPO command to get topology and status for specific resources or groups of resources. For example, if you are monitoring only the MultiSystem Manager topology agent, you can issue a GETTOPO IPRES (IP specifies the network) command to gather information about the resources managed by that agent.

Removing MultiSystem Manager Objects from Views

Under usual network operations, you might have objects in your views that you do not need to monitor. Perhaps you want to remove objects from a view because their status has been UNKNOWN or UNSATISFACTORY for an extended period of time. Any object that has had UNKNOWN or UNSATISFACTORY status for an extended period of time probably is no longer connected to your network, and can be removed from your views. You can remove objects from your views by one of the following methods:

- Issuing a command from the NetView operator command line
- Scheduling a user-written command procedure to run as part of your automated network operations

Removing Objects Meeting Criteria

MultiSystem Manager provides a command, REMVOBJS, to remove objects from your views if the objects meet certain criteria. This command can best be thought of as a cleanup routine. You can issue the REMVOBJS command against an aggregate object or a real object.

When objects, both real and aggregate, are defined in RODM a Purge attribute is created for each object. This attribute defines whether the object can be removed from your views. The Purge attribute can be set as shown in Table 50.

Table 50. Purge Attributes and Meanings

Purge Attribute Value	Meaning
0	Object can be removed
1	Object cannot be removed but certain links can be removed
2	Neither the object nor any of its links can be removed

An object is defined as being able to be purged if the Purge attribute is zero. When MultiSystem Manager creates a new object in RODM, the value of the Purge attribute for the object is set to zero. This enables the object to be removed. To prevent an object from being removed from your views, use the SETREMV command to change the value of the Purge attribute. For more information about the Purge attribute and using the SETREMV command, see the *IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)* or online help.

Note: The display of special connectivity relationships, such as in an IBM token-ring, might be affected by the removal of objects from the view. If the NetView management console does not display the view after the REMVOBJS command runs, you must rebuild the view by issuing the appropriate GETTOPO command.

Removing Real Objects

Each real object must satisfy the following two criteria before it can be removed from your views:

- The object must be outdated, that is, it must have had UNKNOWN or UNSATISFACTORY status for a user-specified period of time.
  - When you issue the REMVOBJS command, specify whether you want to remove objects in UNKNOWN or in UNSATISFACTORY status.
  - When the command is issued, specify how long the objects to be removed can be in an UNKNOWN or UNSATISFACTORY status. If you do not specify a time period, the default is to remove objects after 1 day (24 hours).
- The object must be able to be purged; its Purge attribute is zero.

If the real object is both outdated *and* able to be purged, REMVOBJS removes the real object from your views. If neither or only one of the criteria are met, the real object is not removed.

Note: The REMVOBJS command has completed successfully when all selected objects that are outdated and are able to be purged are removed. If no selected objects meet this criteria, the command still completes successfully, though no objects were removed.

Removing Aggregate Objects

Each aggregate object must satisfy the following two criteria before it can be removed from your views:

- The aggregate object must be defined in RODM as being able to be purged; its Purge attribute is zero.
- All the real objects included in the aggregate object must also meet the criteria to be removed (outdated and able to be purged).

  If one or more of the real objects included in the aggregate object do not meet the criteria, the aggregate object is not removed.

When you issue a REMVOBJS command against an aggregate object, MultiSystem Manager first checks to determine if the aggregate object itself can be purged. If it can be purged, MultiSystem Manager searches through RODM to determine if any of the real objects included in the aggregate object are outdated and can be purged. These real objects are then removed from your view. If all the real objects are removed, then the aggregate object is removed.

Preventing an Object from Being Removed

MultiSystem Manager provides a command, SETREMV, to change the Purge attribute of any MultiSystem Manager-created object. By changing the Purge attribute, you can prevent objects from being removed from your views.

For more information about the SETREMV command, see the online help.

Using Online Help

MultiSystem Manager provides online help for commands and messages. This help is available on your NetView management console and your NetView operator station task (OST). If you cannot access help, verify that the installation procedures were followed correctly.

Help for the topology commands and all messages is available from the NetView command line. See the online help for more information about the NetView HELP command.

- For information about a specific command, enter `HELP command_name`, where *command_name* is the name of the command such as GETTOPO IPRES.
- For information about a specific message, enter `HELP message_id`, where *message_id* is the identifier of the message such as FLC000.

Chapter 11. Open Topology Interface Network Operation

You can use the MultiSystem Manager Open topology feature to create your own topology agents and to manage diverse network resources, those resources not managed by other MultiSystem Manager topology features, for example.

You can also obtain applications from Tivoli Partner organizations that use the Open Topology Interface. Partner applications, that extend IBM Tivoli NetView for z/OS management to network and systems resources that are not managed by Tivoli features, are available.

This chapter explains using the views and command support to manage open networks. See *IBM Tivoli NetView for z/OS Installation: Configuring Graphical Components* for more information about creating Open topology agents. After you initialize network topology and status in RODM, the NetView management console provides views of your Open networks. Using the NetView management console pull-down menus, you can navigate among the views to isolate failures and send commands to resolve problems.

Open Topology Interface View Objects

The Open Topology Interface can be used to display views containing any object in the *IBM Tivoli NetView for z/OS Data Model Reference*. The following Open objects are displayed in various Open views:

- A group of Open networks
- A single Open network
- A collection of resources grouped by location, business organization, or network systems topology
- Aggregate nodes
- Real nodes
- Aggregate links
- Real links

Figure 20 on page 106 shows the various types of objects that are displayed when using the Open Data Model.

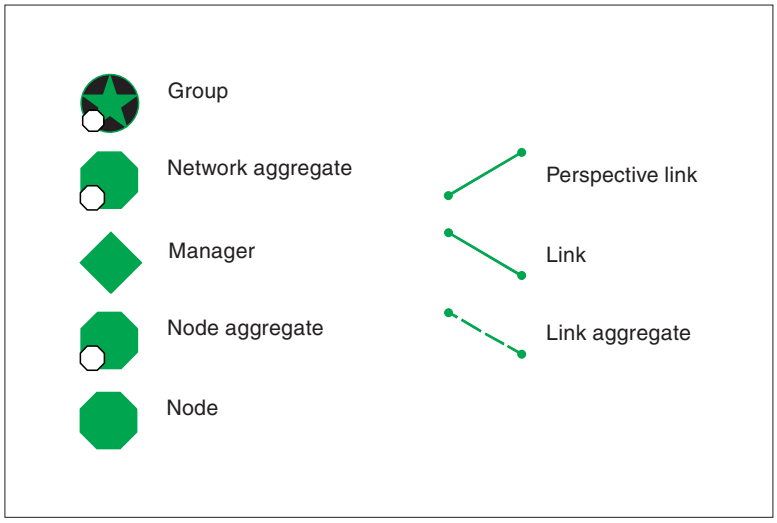


Figure 20. Types of Open View Objects

Finding Resources

To find resources, access the Locate Resource window. From the NetView management console **Tasks** menu, select **Locate Resource** to open the Locate Resource window. From the Locate Resource window, you can locate a resource by its display name.

Navigating Network Views

How you navigate through an Open network depends upon the views provided by your application. Open views can consist of different levels, as defined by the application used. By default, MultiSystem Manager creates 3 high-level objects for every Open Topology Interface application and displays them in two views. This topic illustrates the default set of views displayed when you navigate an example Open network. In the example shown in this chapter, the application is named 'Open.' The following GETTOPO command, sent to a topology agent with an LU name of NTB6I133, was used to build this set of views:

```
GETTOPO OPENRES, SP=NTB6I133, APPL=OPEN
```

To navigate the views, begin by selecting the default MultiSystem Manager network view, MultiSysView, from the NetView management console business tree as shown in Figure 21 on page 107.

Figure 21 on page 107 shows the NetView management console network view named `MultiSysView`.

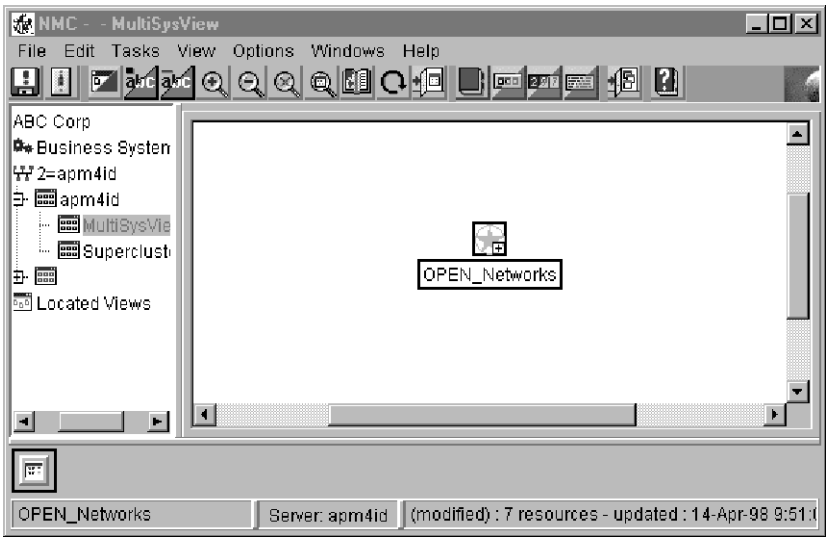


Figure 21. NetView Management Console Default Network View (MultiSysView) Example

MultiSysView consists of a cluster object (a star within a circle) that represents all the Open networks defined in the initialization file used to initialize MultiSystem Manager. This aggregate object is named Open_Networks and its resource type is *Group*. From MultiSysView, you can select more detailed views that display your Open networks.

Viewing Open Networks

The networks object is displayed in its own view, as shown in Figure 21 on page 107. One or more pairs of network and manager objects are created as children of the networks object. A pair of these objects are shown in Figure 22 on page 108.

Open Networks View

MultiSystem Manager displays a separate Open network for each one defined in your initialization file. To view your Open networks on the NetView management console, select the `Open_Networks` object as shown in Figure 21 and select **More Detail**. The resulting NetView management console view, as shown in Figure 22 on page 108 in our example, shows one Open networks object containing one Open network object.

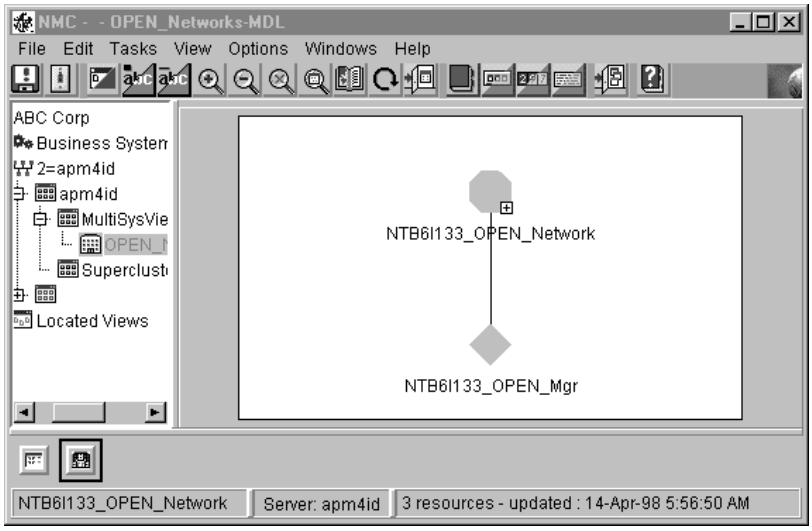


Figure 22. NetView Management Console Open Networks View (Open_Networks) Example

MultiSystem Manager represents each Open network by two connected symbols:

- A node aggregate object called a *network aggregate*, for example, `NTB6I133_OPEN_Network`.
- An application real object called a *manager*, for example, `NTB6I133_OPEN_Mgr`.

The network aggregate represents all the resources monitored by the Open application. The manager, `NTB6I133_OPEN_Mgr`, represents the topology agent residing in the Open application. The name of the agent is composed of the SNA service point address and the type of network the agent is managing (Open network).

To view additional information about the Open network on NetView management console, select the Open manager and select **Resource Properties**. Figure 23 shows the resulting NetView management console Resource Properties window for the selected Open object.

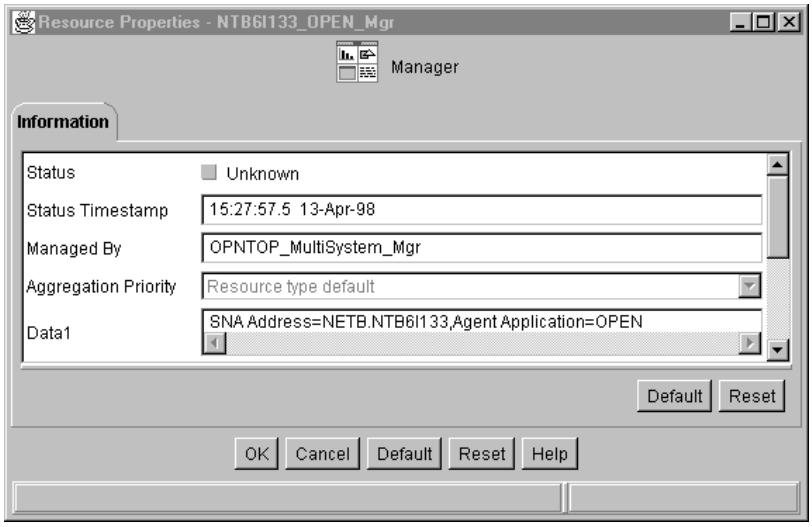


Figure 23. NetView Management Console Open Topology Interface Resource Properties Window Example

An Open Topology Interface application consists of one or more additional levels of views. These views can consist of real objects and aggregate objects with any of these objects shown linked together. Refer to the information with each of your Open Topology Interface applications for a description of navigating the views contained in a network aggregate object.

Chapter 12. Tivoli Management Region Operation

This chapter explains how to use the views and command support to manage your Tivoli Management Region networks. After you initialize network topology and status in RODM, the NetView management console provides views of your Tivoli Management Region networks. Using the NetView management console pull-down menus, you can navigate among the views to isolate failures and send commands to resolve problems.

Tivoli Management Region View Objects

The following Tivoli Management Region objects are displayed in various Tivoli Management Region views:

- Aggregate objects that represent:
  - A group of Tivoli Management Region networks
  - A single Tivoli Management Region network
  - Managed regions
  - User-defined policy regions
  - Managed nodes (IP Systems)
  - Gateways
  - Endpoints
  - Profiles
  - Resource Models
- Real objects that represent:
  - Topology agents
  - Indicators
  - Resource Models

Figure 24 shows the various types of objects that you can see in your MultiSystem Manager views of Tivoli Management Region.

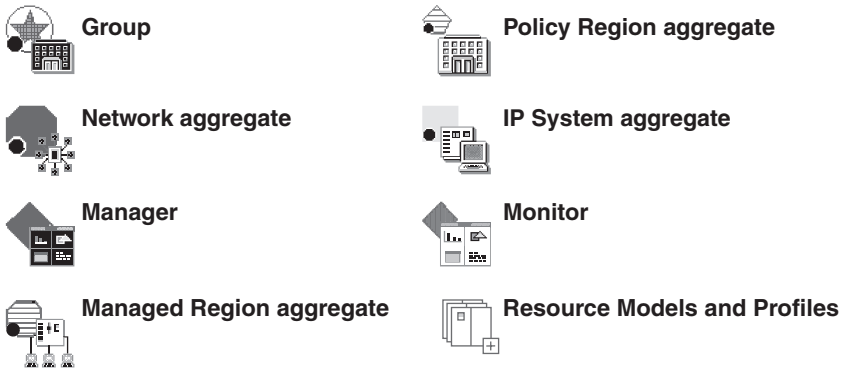


Figure 24. Types of Tivoli Management Region View Objects

Finding Resources

To find resources, access the Locate Resource window. From the NetView management console **Tasks** menu, select **Locate Resource** to open the Locate Resource window. From this window, you can locate the Tivoli Management Region device by its names. You can also locate a resource by its display name.

Navigating Network Views

To navigate the views, begin by selecting the default MultiSystem Manager Manager network view, **MultiSysView**, from the NetView management console business tree as shown in Figure 25 on page 112.

Note: The following GETTOPO command, sent to a topology agent with a host name of MSMTIV2, was used to build this set of NetView management console views:

```
GETTOPO TMERES, SP=MSMTIV2
```

Figure 25 shows the NetView management console network view named MultiSysView.

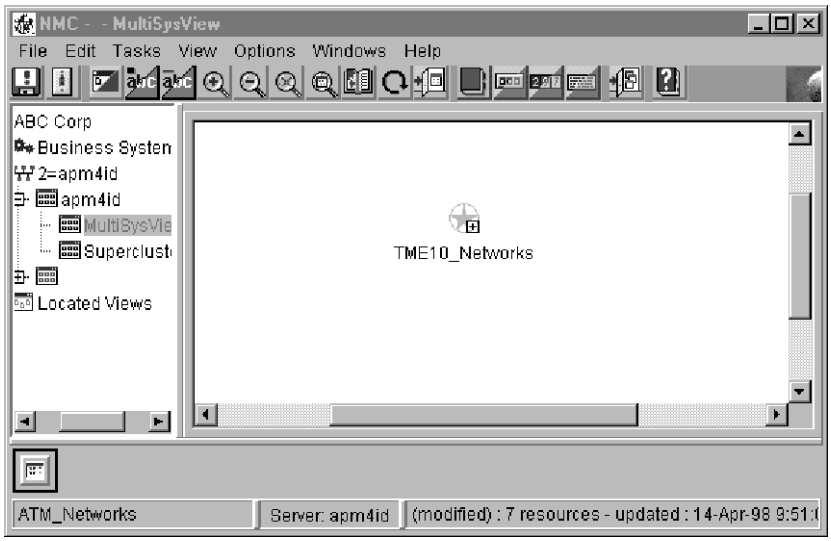


Figure 25. NetView Management Console Default Network View (MultiSysView) Example

MultiSysView consists of an object (a NetView management console cluster aggregate object) that represents all the Tivoli Management Region networks defined in the initialization file used to initialize MultiSystem Manager. This aggregate object is named `TME10_Networks` and its resource type is Group. From MultiSysView, you can get more detailed views that display your Tivoli Management Region networks.

Viewing Tivoli Management Region Networks

Tivoli Management Region views consist of the following levels:

Tivoli Networks View
: A single view that shows all your Tivoli Management Region networks

Tivoli Network Views
: Views showing the Tivoli Management Region and policy regions of a Tivoli Management Region network

TMR View
: A view showing the managed nodes in a Tivoli Management Region in their repeater configuration

TME Policy Region
A view showing the managed nodes and sub-policy regions in a policy region

TME Managed Node View
A view showing the IBM Tivoli Monitoring monitors associated with a managed node or endpoint

Profiles
A view showing the profiles that have been distributed to an endpoint

Resource Models
A view showing the Resource Models that are contained in a profile

Tivoli Networks View

To create the Tivoli management region networks view for NetView management console, select the `TME10_Networks` object displayed in Figure 25 on page 112 and select **More Detail**. The resulting view, Figure 26, shows the Tivoli Management Region networks included in the aggregate object `Tivoli_Networks`.

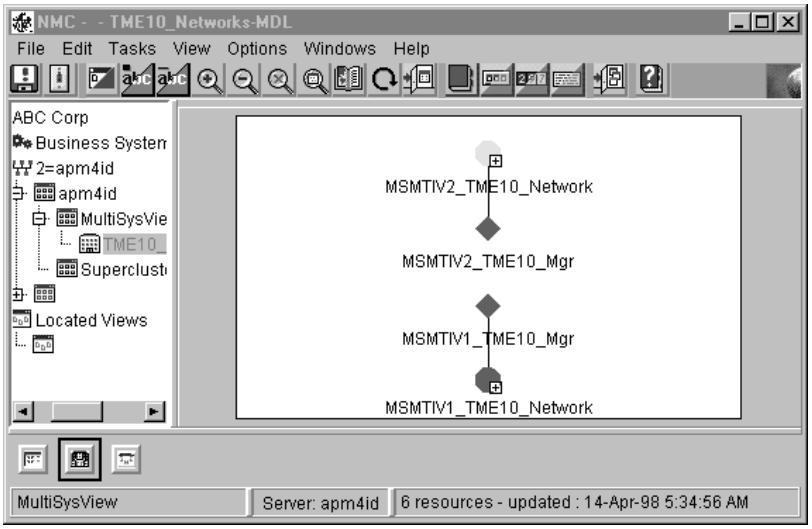


Figure 26. NetView Management Console Tivoli Networks View Example

This view shows one Tivoli Management Region network. MultiSystem Manager displays a separate Tivoli Management Region network for each Tivoli Management Region agent defined in your initialization file. You will see a separate Tivoli Management Region network for each GETTOPO statement you have in your initialization file.

Each Tivoli Management Region network is represented by two connected symbols:
- A node aggregate object (`MSMTIV2_Tivoli_Network`)
- An application real object (`MSMTIV2_Tivoli_Mgr`)

In Figure 26, the aggregate object represents all the managed Tivoli Management Region resources in the network. The real object represents the MultiSystem Manager topology agent. The name of the agent is composed of the TCP/IP host name and the type of network the agent is managing, Tivoli Management Region.

To find additional information about the topology agent for NetView management console, select the Tivoli Management Region manager and select **Resource Properties**. Figure 27 shows the resulting Resource Properties window for the selected Tivoli Management Region manager on the NetView management console.

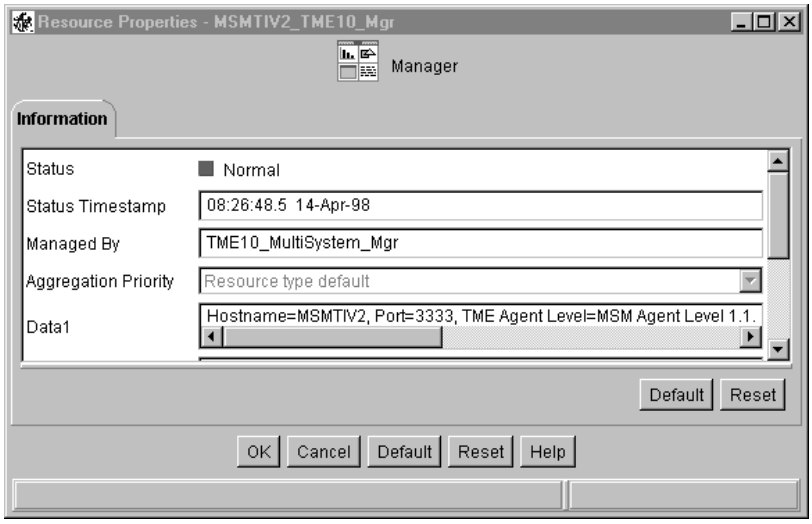


Figure 27. NetView Management Console Resource Properties Window Example

Tivoli Network Views

To view the managed Tivoli Management Region resources in the network, select one of the managers, examples of which are shown in Figure 26 on page 113, then select **More Detail**. Your NetView management console creates a view showing the managed region aggregate and policy region aggregates that are related to the manager in Figure 28 and Figure 29 on page 115.

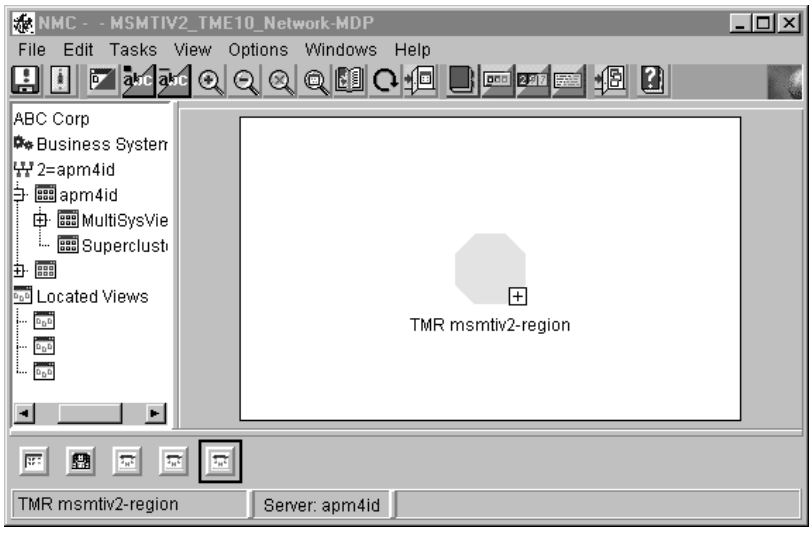


Figure 28. NetView Management Console Tivoli Network Managed Region Example

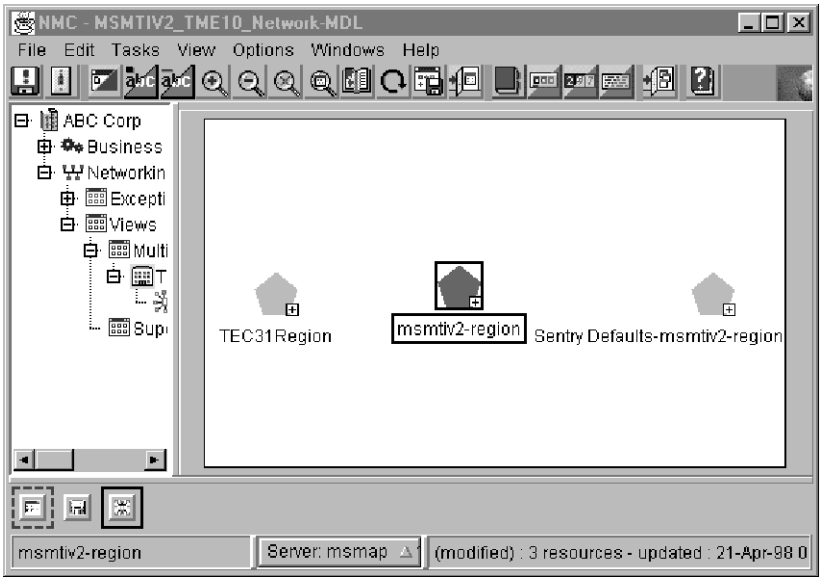


Figure 29. NetView Management Console Tivoli Policy Region Example

Tivoli Management Region Views

Each Tivoli Management Region Managed Region Aggregate consists of managed nodes. To view the IP Systems Aggregates, select the `Managed Region` aggregate object and then select **More Detail**. Figure 30 displays the software distribution configuration for the MSMTIV2 Tivoli Management Region on the NetView management console.

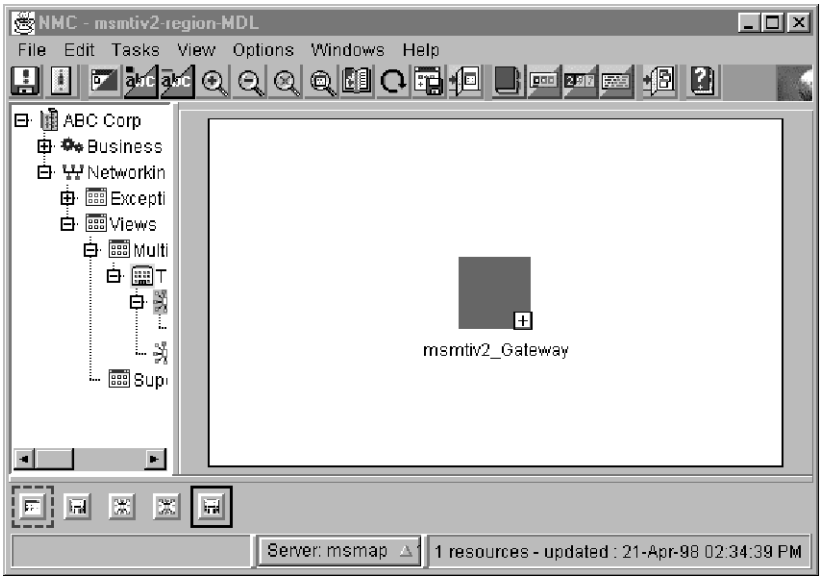


Figure 30. NetView Management Console IP System Aggregate View Example

Tivoli Management Region and End Points

The Tivoli environment extends the number of nodes a Tivoli Management Region can manage with a gateway (Tivoli gateway), which is a role taken by one or more managed nodes in a Tivoli Management Region.

The gateway acts as a surrogate Tivoli Management Region server for the resources and end points that are logged into it. The endpoints do not have an `oserv` running on them, but they can run methods sent to them by the gateway, where they can then cache locally. Endpoints function like managed nodes, and the topology agent gathers the same information from the endpoints that it does from the managed nodes in a Tivoli Management Region:

- Tivoli Management Region number
- IP address
- Host name
- Status
- OS type
- OS release
- OS version

Any managed node that functions as a gateway becomes a repeater. The gateways are automatically displayed in the repeater configuration and physical view of the MultiSystem Manager agent. Figure 31 shows an example view of gateways and endpoints.

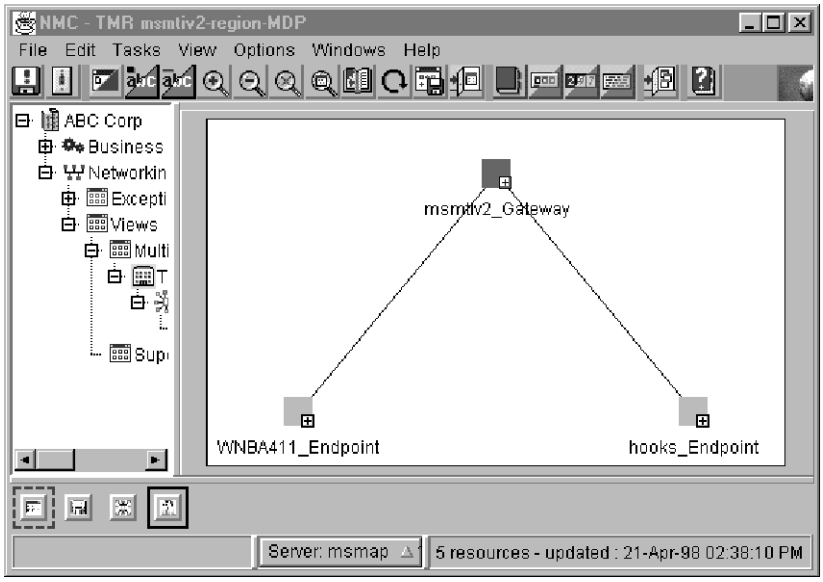


Figure 31. NetView Management Console Gateways and Endpoints View Example

Figure 31 shows two endpoints (`WNBA411_Endpoint` and `hooks_Endpoint`) that are connected to a gateway (`msmtiv2_Gateway`). The gateway also serves as a Tivoli Management Region server in this view.

Tivoli Management Region Managed Node Views

To view the details of a specific Managed Node in the Tivoli management region, select the Managed Node, then select **More Detail**. The Managed Node details shows the oserv process that is running on the managed node and the status of that process. Note that gateways are also Managed Nodes and can be viewed using the same method. Figure 32 on page 117 is an example of a view showing the Managed Node details.

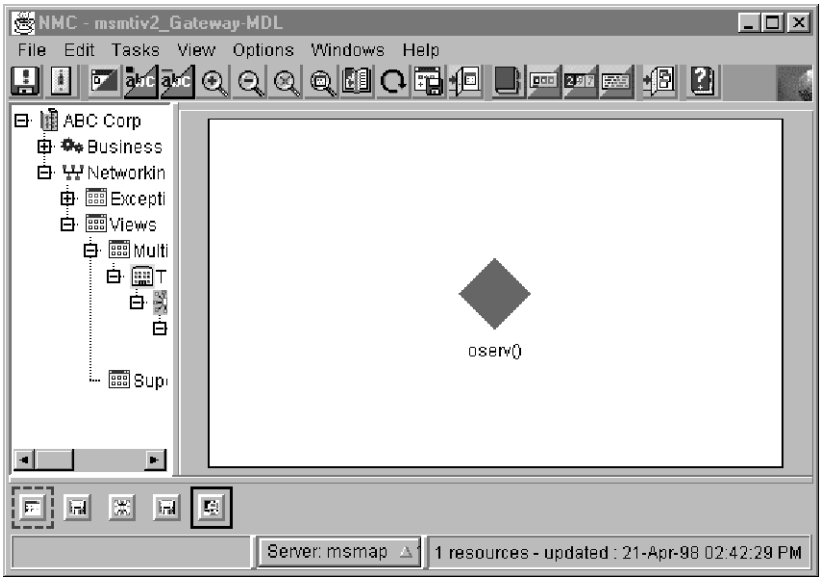


Figure 32. NetView Management Console Monitor View Example

Tivoli Managed Region Endpoint View

To view the details of a specific endpoint in the TMR, select the endpoint and then select **More Detail**. The endpoint details view shows an object representing the LCF process and its status, plus any profiles that are distributed to that endpoint. Figure 33 is an example showing the endpoint details.

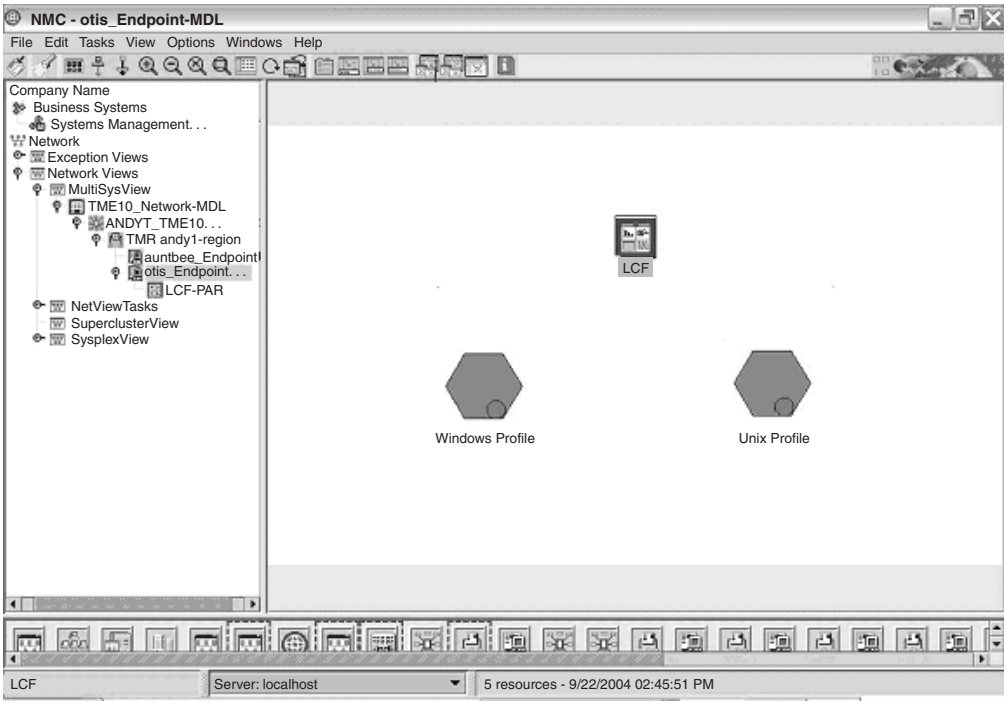


Figure 33. Endpoint Details View

Tivoli Managed Region Profile View

To view the details of a specific profile in an endpoint, select the profile and then select **More Detail**. The profile details view will show an object representing an Resource Models that are distributed in that profile. Figure 34 is an example of a view showing the profile details.

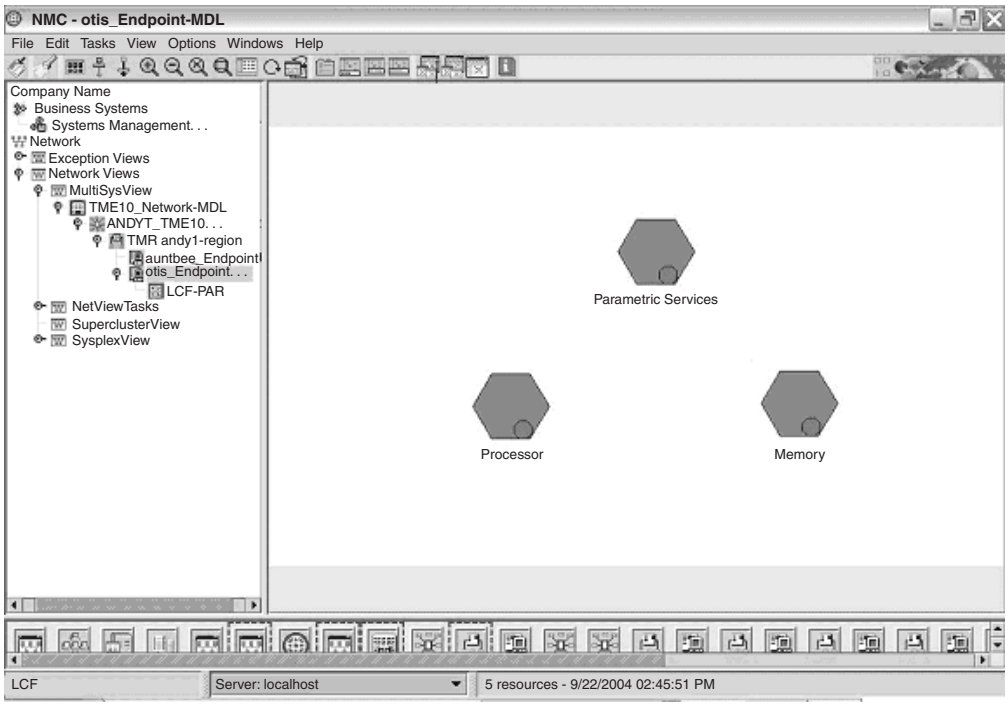


Figure 34. Profile Details View

Tivoli Managed Region Resource Model View

To view the details of a specific Resource Model in a profile, select the Resource Model, and then select **More Detail**. The Resource Model details view will show an object representing the status of the Resource Model, plus any indicators that are distributed in that Resource Model. If the Resource Model has been distributed, but has been disabled the object representing the Resource Model itself will be red; otherwise it will be green. The status of the indicators is kept current through the use of events from the framework environment. If the indicator is configured to send events, but is not configured to send clearing events, objects might turn red, but will not go back to green status without operator intervention. Figure 35 on page 119 is an example of a view showing the profile details.

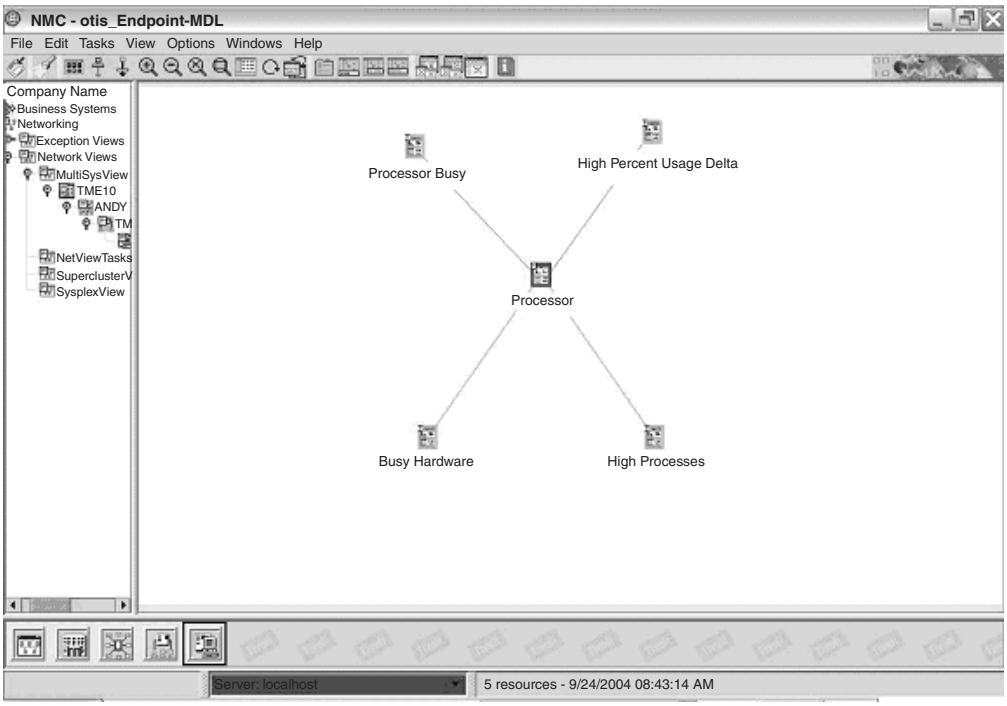


Figure 35. Resource Model Details View

Chapter 13. Using NetView Management Console Command Profiles

This chapter contains Programming Interface information.

When operators use the topology console, available commands are displayed in the context menu for a selected resource. Also, resource independent commands are shown in a context menu if you right-click the view background. Because operators have different areas of responsibility, levels of authority, and preferences, you might want to customize the command menus. Use the command profile editor batch-utility to control the content, order, and capabilities of these menus for an individual operator or group of operators.

The NetView management console provides a command profile editor batch-utility on both Windows and UNIX platforms.

Understanding Command Profiles

Command profiles contain the following objects:

- Resource managers
- Commands and command sets
- Profiles
- Operators

Resource Manager Objects

Resource manager objects represent applications that manage specific resources. The properties of a resource manager define a unique range of command

indicators and specify the values in that range that are enabled for commands. You can use the command profile editor to add, change, or delete resource managers.

A command indicator is a numeric identifier assigned to a resource by its controlling resource manager. Each resource manager is assigned a unique range of values that it can define. Unless you are creating a resource management application or performing actions on behalf of such an application, you probably never need to alter the properties of a resource manager.

Note: The first 32767 command indicators are available for customer use. Command indicators 32768 to 65534 are not defined and they are reserved for current and future Tivoli use. For a detailed description of these command indicators, see the *IBM Tivoli NetView for z/OS Data Model Reference*.

Commands and Command Set Objects

Command objects represent menu items that can be displayed in the context menus for selected resources. Command set objects also represent menu items, but command set items have submenus of commands or other command sets associated with them. The command set object specifies the order and content of the submenu. You can use the command profile editor to add, change, and delete commands and command sets.

Commands consist of the following items:

- A command definition page that defines the purpose and behavior of the command
- One or more implementation pages that describe the command exits that are responsible for carrying out the command

Command sets are composed of one or more commands or command sets.

Many commands have a common meaning for a wide variety of resource types. However, the mechanics of carrying out this command can vary from one resource type to another.

For example, a Lotus® Notes® resource is started differently than a NetView resource, which is activated differently than a Tivoli Enterprise Console® resource. This is because a variety of syntaxes and command exits are used; therefore command objects are composed of multiple implementation pages. When a command is issued, a single page is selected (based on the command indicator of the target resource).

For example, you can create a **Stop Tasks** command that is enabled for Lotus Notes resources, but not for Tivoli Enterprise Console resources. You can do this by defining a command called **Stop Tasks** and defining a page for Lotus Notes resources but not defining a page for Tivoli Enterprise Console resources.

Profile Objects

Profile objects represent a collection of commands. Through the commands, command sets, and separators, the profile objects define the content and order of the items on the context menus. You can use the command profile editor to add, change, and delete command profiles.

A profile can be shared by multiple operators. An operator object is created for each operator who does not want to use the default profile. The operator object is then assigned a profile object. Changes to a profile affect all operators using that profile. This method makes it easier to maintain profiles and ensures that operators are using the same commands.

Note: A profile named <default> is used for all operators who do not have a specific operator definition. The <default> profile is appended by the topology server during initialization when processing component description files or business description files (CDF/BDF) of instrumented applications. This is done for the Topology Display Subsystem view.

Operator Objects

An operator object represents a topology console operator and is associated with that operator by a common name. An operator object specifies the profile object assigned to the operator. You can use the command profile editor to add, change, or delete operators.

When an operator right-clicks a resource, the profile object used to populate the context menus for that operator is identified in the following ways:

- If an operator object exists with the same name, its specified profile object is used.
- If an operator object does not exist with the same name, but a profile object with the name <default> exists, the profile object is used.
- Otherwise, no commands from the command profile editor are displayed on the context menu.

Note: When you delete or rename the <default> profile, only operators with operator objects assigned to them have access to commands from the NetView management console. You can use this to provide security for a topology server.

Using the Command Profile Editor Batch Utility

You can use the command profile editor batch utility program to add, change, or delete commands in the command profile editor database. The batch utility runs on the Windows and UNIX platforms that are supported by the topology server. The utility uses a text file (the response file), which contains responses to all the parameters used in creating commands.

The command profile editor batch utility can do the following things:

- Create a response file from the information currently in the command profile editor database of the topology server. Note that the response file can be used as a backup for the command profile editor database or for synchronizing your topology servers.
- Update the command profile editor database of the topology server from a response file.

Any program can provide a response file and run the batch utility to add commands to the command profile editor database.

Two sample response files are provided with the command profile editor. These are available from the databases of the NetView management console topology server. The following samples are provided for customizing the commands:

ihsscpe
Base commands that are shipped with the NetView management console topology server

flccpe Commands for MultiSystem Manager (MSM) resources

These files are located in the following directories:

- ihsscpe, for Windows: %BINDIR%\TDS\server\sample\ihsscpe.*xxx*.rsp
- ihsscpe, for UNIX: $BINDIR/TDS/server/sample/ihsscpe.*xxx*.rsp
- flccpe, for Windows: %BINDIR%\TDS\server\sample\flccpe.*xxx*.rsp
- flccpe, for UNIX: $BINDIR/TDS/server/sample/flccpe.*xxx*.rsp

Note: *xxx* is a country code indicator, such as en_US.

Depending on the parameters specified on the command line, the utility adds, modifies, or deletes commands. The command profile editor utility does not save information to the topology server until the entire response file is processed and verified. If the entire file cannot be processed successfully, no updates are made in the topology server database.

Starting the Command Profile Batch Utility

To run the command profile editor batch utility, issue the **cpebatch** command from one of the following directories:

- For Windows systems: %BINDIR%\TDS\server\bin
- For UNIX systems: $BINDIR/TDS/server/bin

For more information about the **cpebatch** command, see "cpebatch" on page 139.

Input and Output Files of the Response File

The format for the input and output of the response file is identical. The response file is a standard ASCII file containing assignment statements that are generally in the form of *keyword = value*. The *keyword* is on the left side of the statement and identifies the parameter. The *value* is on the right side of the statement and either assigns a value to the keyword, or contains a block of keyword and value assignments.

Lines in which the first nonblank character is an asterisk (*) are comments and are ignored by the utility. Comments can be either inside a block or outside a block, for example:

```
MANAGER = (
* This is a comment inside of the manager block
     NAME = SNA
     INDICATOR_HIGH = 10
     INDICATOR_LOW = 5
     INDICATOR.INDICATOR_LOW+0 = PU 2.1
     INDICATOR.INDICATOR_LOW+1 = PU 2.0
     INDICATOR.INDICATOR_LOW+2 = 3274
Communications Controller
     COMMENT = Defines manager SNA
)
* This is a comment outside of the manager block
```

The preceding example creates a manager named SNA and assigns it a range of command indicators from 5 to 10. It also creates three indicators for that manager, the lowest indicator defined is 5. For an example input file, see:

- For Windows: `%BINDIR%\TDS\server\Sample\ihsscpe.xxx.rsp`
- For UNIX: `$BINDIR/TDS/server/Sample/ihsscpe.xxx.rsp`

Note: *XXX* is a country code indicator, such as en_US.

The response file is processed in a single pass; therefore, the order of the main blocks of keywords is important. Use the following order:
1. Manager
2. Command
3. Command_Set
4. Profile
5. Operator

You cannot use items before they are defined in the response file. For example, if command set A embeds command set B, command set B must be defined first.

You can omit any block as long as all the required keywords are already defined in the topology server. For example, if you add a command and the resource manager is already defined in the topology server, it is not necessary to define the manager in the response file.

You can delete a block from the current database by placing the DELETE keyword inside the block.

Note: Before deleting a command or command set, it must first be removed from any profiles or command sets that contain it. To do this, use the `REMOVE_COMMAND` or `REMOVE_COMMAND_SET` option of the `MENU` keyword for each profile or command set that contains the command or command set.

You can delete the following types of blocks:
- Manager
- Command
- Command_Set
- Profile
- Operator

Manager Keywords

The manager block defines a manager and the command indicators that this manager controls. The keywords to define managers in command blocks are shown in Table 51.

Table 51. Manager command block keywords

Keyword	Explanation
Name	Uniquely defines the manager to the command profile editor.
INDICATOR_HIGH	High end of the range of indicators controlled by this manager.
INDICATOR_LOW	Low end of the range of indicators controlled by the manager.
INDICATOR_*n*	*n* must be the indicator value followed by the description string. The indicator value can be either a fixed number or can be relative to the INDICATOR_LOW+*x*, when *x* is the offset of this indicator from the low range.
COMMENT	Comment string. Limit of 256 characters.

Command Keywords

The command block defines a complete command, including general information and each of the command pages. The keywords used in command blocks are shown in Table 52.

Table 52. Command block keywords

Keyword	Explanation
NAME	Uniquely identifies the command to the command profile editor.
MENU_STRING	The string that is displayed on the context menu.
COMMENT	Comment string. Limit of 256 characters.
HTML_HELP_FILE	The name of the HTML file that contains the help for this command.
HTML_HELP_ANCHOR	The anchor, if applicable, to a particular location in the HTML file that contains the help for this command. The pound sign (#) in the first position is optional; when HTML is displayed by the NetView management console topology console, # is inserted, if necessary.
MIN_RESOURCES	Minimum number of resources that must be selected in a view before this command is enabled. The range is 1 to 10.
MAX_RESOURCES	Maximum number of resources that can be selected in a view before this command is disabled. The range for this is *value of the minimum resources*–100, or an infinite number. An infinite number is denoted by the keyword INFINITY. **Note:** Specifying too many resources can cause system resources to become overburdened and, therefore cause the system to hang.
PAGE	Defines the characteristics of a page for an individual command. See "Page Keywords in the Command Block" for the items you can specify here.
RESOURCE_INDEP	YES or NO. Indicates whether this command is enabled regardless of whether resources are selected. If the value of RESOURCE_INDEP is YES, then MIN_RESOURCES and MAX_RESOURCES values are ignored. Resource dependent commands are displayed when you right-click a resource. Resource independent commands are displayed when you right-click the view background.
VERIFY	YES or NO. Indicates whether to issue a confirmation message before the command is sent to the manager.

Page Keywords in the Command Block

The page keywords in the command block define the characteristics of a page for an individual command. The keywords to define pages in command blocks are shown in Table 53 on page 125.

Table 53. Page command block keywords

Page Characteristic	Explanation
CLIENT_PLATFORM_LIST	The topology console platform or platforms associated with this command page.Use the following syntax for all platforms: `CLIENT_PLATFORM_LIST = GENERIC`
TARGET_PLATFORM_LIST	The target platform or platforms where the selected resource is located. Use the following syntax for all platforms: `TARGET_PLATFORM_LIST = GENERIC`
MANAGER_NAME	The name of the manager to which the page applies. This must be previously defined in the response file. If this command is not associated with particular resource types, specify the string `ANY`. An `ANY` manager command can have only one command page, which can not specify an INDICATOR_LIST or PAGE_ID.
PAGE_ID	Uniquely identifies this command page by specifying one indicator from the indicator list. When updating a command page, the PAGE_ID must be in the indicator list of the existing page in CPE.

INDICATOR_LIST	Defines the indicators that call this page. Specify single indicators by separating each with a comma. Specify a range by connecting numbers with a dash (for example, 1–100). You can replace real numbers with relative numbers using INDICATOR_LOW+*x*. INDICATOR_LOW denotes the lowest defined indicator for this manager.
	
	If the indicator list spans multiple lines, you can use the following format:
	`INDICATOR_LIST = (`
	`  VALUE.0 = INDICATOR_LOW + 0`
	`  VALUE.1 = INDICATOR_LOW + 1`
	`)`

COMMAND_LIST	The command string to be sent to the command exit specified by EXIT_NAME.
EXIT_NAME	The name of the command exit to call for this page. See “Using Topology Server Command Exits” on page 130 for information.
LU_NAME	Currently, this field is ignored by NetView management console. All exits are run on the topology server.
HTML_HELP_FILE	The name of the HTML file that contains the help for this command.
HTML_HELP_ANCHOR	The anchor, if applicable, to a particular location in the HTML file that contains the help for this command. The pound sign (#) in the first position is optional; when HTML is displayed by the topology console, # is inserted, if necessary.
IP_RESOURCE	IP Resource commands are displayed In the NetView management console when a resource with an IP address is right-clicked. Valid values are `YES` or `NO`. The default value is `NO`.

Command Set Keywords

The command set block defines what a command set looks like. Order is important in the menu block. The keywords used in command set blocks are shown in Table 54.

Table 54. Command set block keywords

Keyword	Explanation
NAME	Uniquely identifies the command set to the command profile editor.
COMMENT	Comment string. Limit of 256 characters.
MENU_STRING	The string that is displayed on the Context menu.
HTML_HELP_FILE	The name of the HTML file that contains the help for this command set. This file must be installed on the topology server.
HTML_HELP_ANCHOR	The anchor, if applicable, to a particular location in the HTML file that contains the help for this command. The pound sign (#) in the first position is optional; when HTML is displayed by the topology console, # is inserted, if not already there.
MENU	Defines the commands that this command set contains and their order. You can specify one or more of the following options: • **COMMAND_NAME** - Specifies the name of a command to add. • **COMMAND_SET_NAME** - Specifies the name of a command set to add. • **SEPARATOR** - Specifies that a separator is to be placed on the menu. • **REMOVE_COMMAND** - Specifies the name of a command to remove. • **REMOVE_COMMAND_SET** - Specifies the name of a command set to remove.

To add a new command to a command set or profile, first identify the command set or the profile to which you want to add the command, then specify the command you want to add inside the MENU block. The new command must already be defined in the database or must have been defined earlier in the response file.

The following example shows how to add `My command` to the Network command set:

```
COMMAND_SET = (
  NAME = Network               (identifies the
command set)
  MENU = (
    COMMAND_NAME = My command  (identifies
the command to add)
  )
)
```

This example places `My command` at the end of the Network command set. Make sure you use the `-G` option on the **cpebatch** command when adding the command.

To insert My command after an existing command in the command set, specify the following lines:

```
COMMAND_SET = (
  NAME = Network
  MENU = (
    COMMAND_NAME = NetView command line   (existing
command)
    COMMAND_NAME = My command
  )
)
```

To remove My command from the Network command set, specify the following lines:

```
COMMAND_SET = (
  NAME = Network               (identifies the
command set)
  MENU = (
    REMOVE_COMMAND = My command  (identifies
the command to remove)
  )
)
```

Profile Keywords

The profile block defines individual profiles. Order is important in the menu block. The keywords used in the profile command block are shown in Table 55.

Table 55. Profile command block keywords

Keyword	Explanation
NAME	Uniquely identifies the profile to the command profile editor.
COMMENT	Comment string. Limit of 256 characters.
MENU	Defines the commands that this profile contains and their order. You can specify one or more of the following options: • **COMMAND_NAME** - Specifies the name of a command to add. • **COMMAND_SET_NAME** - Specifies the name of a command set to add. • **SEPARATOR** - Specifies that a separator is to be placed on the menu. • **REMOVE_COMMAND** - Specifies the name of a command to remove. • **REMOVE_COMMAND_SET** - Specifies the name of a command set to remove.

Operator Keywords

The operator block defines operators to the command database. The keywords used in the operator command block are shown in Table 56.

Table 56. Operator command block keywords

Keyword	Explanation
NAME	Uniquely defines the operator to the command profile editor
COMMENT	Comment string. Limit of 256 characters
PROFILE_NAME	The name of the profile that this operator is to use

Chapter 14. Using the Topology Server Command Exits

This chapter contains Programming Interface information.

This chapter contains information about topology server command exits. The command exit facility enables commands defined in the context menus for selected resources to call specific functions when those commands are selected. These specific functions are known as command exits and are processed as remote procedure calls. When a command is selected, its corresponding command exit procedure is driven by the topology server.

Command Profiles

Command profiles define the commands available from the context menus for a particular operator. When an operator right-clicks a resource or the view background, the command profile for that operator is used if it exists; otherwise, the default command profile is used.

A default command profile is shipped with NetView management console. You can add command definitions to the default command profile or modify existing definitions. You can create new command profiles for individual operators or groups of operators. Commands are also automatically defined by component instrumentation.

Understanding Topology Server Command Exits

The topology server provides a set of command exits to send commands from the workstation to the NetView host as described in "Using Topology Server Command Exits" on page 130. The command exit facility supports exit-to-exit communication, which enables a command exit to modify a command and to pass the data to another command exit for processing.

When a command exit returns, control is given to its caller. Eventually, the first exit called by the command selection returns.

Note: Some events take place when a command exit is driven. If an exit procedure is not yet registered, then an executable file with the same file name as the exit name is sought using the defined path. If found, this executable file is started in a separate session.

For example, if the exit is called TESTEXIT, the topology server searches for and starts the TESTEXIT.EXE (for Windows platforms) or TESTEXIT (for UNIX platforms). This program registers a procedure within itself as a command exit procedure.

When a command is issued from the topology console, the command exit indicated in the command profile is driven by the topology server. For a resource dependent command, the command exit that is driven is based on the command indicator of the resource. A parameter block is passed to the command exit procedure containing information about the command in the command profile editor and information about a resource (if selected) when the command was called. The command exit procedure can pass a return code to its caller when processing of the command is complete.

Using Topology Server Command Exits

The command exits that are supplied with the topology server for general use are shown in Table 57.

Table 57. Command exits supplied by the topology server

Command exit	Use to...	For information about...
IHSDGENE	Send a fixed set of generic commands to the NetView Graphic Monitor Facility host subsystem (GMFHS) for processing.	Using with the command profile editor, see "IHSDGENE Command Exit."
IHSDNATV	Send commands to GMFHS, which forwards the command to the service point for the specified resource.	Using with the command profile editor, see "IHSDNATV Command Exit."
IHSXTHCE	Send a command to the NetView host.	Using with the command profile editor, see "IHSXTHCE Command Exit" on page 131.
IHSXTJAM	Launch a single Java class on the topology console for multiple resources.	Using with the command profile editor, see "IHSXTJAM Command Exit" on page 132.
IHSXTJAV	Start a Java class on the topology console.	Using with the command profile editor, see "IHSXTJAV Command Exit" on page 132.

IHSDGENE Command Exit

For resources managed by GMFHS and MultiSystem Manager (MSM), use the IHSDGENE exit to send a fixed set of generic commands to NetView GMFHS for processing. Examples of the generic commands you can send include **Activate**, **Inactivate**, and **Recycle**. This exit supports only the commands listed in the default command profile and cannot be extended. To determine the commands supported by this exit and their syntax, look at the default command profile shipped with the topology server as defined in the ihsscpe.*xxx*.rsp and flccpe.*xxx*.rsp response files, where *xxx* is a country code indicator, such as *en_US*.

Note that you do not have to define the generic commands to the command profile editor as they are already defined in the response file.

For more information, see the *IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*.

IHSDNATV Command Exit

IHSDNATV is used to send a command to GMFHS, which forwards the command to the service point for the specified resource. A resource must be selected before IHSDNATV is called.

When defining a command that uses this exit, specify the following information in the Commands notebook.

1. In the Command string field, specify the command to send to a network management gateway that manages the selected resource. GMFHS performs substitution for the following symbols in the command string:

%appl%
Substitutes the value of the TransactionProgram field of the Non_SNA_Domain_Class instance.

%domain%
Substitutes the value of the EMDomain field of the Non_SNA_Domain_Class instance.

%resource%
Substitutes the resource name portion of the MyName field of the GMFHS_Managed_Real_Objects_Class or a subclass of the GMFHS_Managed_Real_Objects_Class instance.

For example, (`EMDomain.Resource = SPI6E69.MINI69A`) causes `MINI69A` to be substituted.

%spname%
Substitutes the value of the MyName field of the `NMG_Class` instance.

%type%
Substitutes the value of the TypeName field of the Display_Resource_Type_Class instance associated with a resource.

2. In the Exit name field, enter `IHSDNATV`.

IHSXTHCE Command Exit

IHSXTHCE sends a command to the NetView host. If a resource is not selected when the command is called, IHSXTHCE can send resource-independent commands to the NetView host. If a resource is selected, the IHSXTHCE command exit can substitute resource specific information.

When defining a command that uses this exit, specify the following information in the Commands notebook:

1. In the Command string field, specify the command to run on the NetView for z/OS program. If you selected **Resource dependent**, you can optionally specify substitution variables in the Command string field. See "Substitution Variables" on page 132 for a list of valid substitution variables.
2. In the Exit name field, enter `IHSXTHCE`.

The following substitution variables are unique to the IHSXTHCE command exit:

%network%
Valid only for SNA topology manager Resource Object Data Manager (RODM) resources with command indicator values of 32769 and 32770. Substitutes the data before the first period in the display name, if it exists.

The display name is the resource name assigned by the manager of the resource and is the name that is displayed in the view. For RODM resources, this data is from the DisplayResourceName field.

%noresponse%
Directs the command response back to the Tivoli NetView for z/OS session.

%resource%
Valid only for SNA topology manager RODM resources with command indicator values of 32769 and 32770. Substitutes the data after the last period in the display name.

The display name is the resource name assigned by the manager of the resource and is the name that is displayed in the view. For RODM resources, this data is from the DisplayResourceName field.

%response%
Directs the command response back to the topology console.

IHSXTJAM Command Exit

The IHSXTJAM command exit starts a Java class on the topology console. The Java class name to be started must be the first blank delimited token in the command string field. Unlike the IHSXTJAV command exit, the IHSXTJAM command exit launches only one instance of the Java class when several resources are selected on the topology console. Thus, one instance of the Java class has access to information about every selected resource. The IHSXTJAM command exit can be specified in a resource independent or a resource dependent command. See Chapter 7, "Topology Console Java Applications and Plug-ins," on page 59 for information about installing this Java class.

IHSXTJAV Command Exit

The IHSXTJAV command exit starts a Java class on the topology console. The Java class name to be started must be the first blank delimited token in the command string field. Unlike the IHSXTJAM command exit, the IHSXTJAV command exit launches multiple instances of the Java class, when several resources are selected on the topology console. For example, when two resources are selected, and a command is selected that calls the IHSXTJAV command exit, two instances of the class specified in the command string launched are on the topology console, with each instance of the Java class having information about one of the selected resources. The IHSXTJAV command exit can be specified in a resource independent or a resource dependent command. See Chapter 7, "Topology Console Java Applications and Plug-ins," on page 59 for information about installing this Java class.

Substitution Variables

Table 58 lists the common variables that are substituted by the topology server in the command string for the command exits. Note that the substitution variables are not case-sensitive. These variables are common across all command exits.

Table 58. Common substitution variables

Substitution Variable	Description
%data1%	Resource specific data1. For RODM resources, this is data from the DisplayResourceOtherData field. For RODM predefined views in the tree view, this data is from the Annotation field.
%data2%	Resource specific data2. For RODM resources, this data is from the DisplayResourceUserData field.
%data3%	Resource specific data3. For RODM resources, this data is from the iPAddress field.
%data4%	Resource specific data4.

Table 58. Common substitution variables (continued)

Substitution Variable	Description
%hb_hostname%	In the distributed environment, the value must be the TCP/IP host name of the system. In the z/OS environment, it is the NetView domain name (non-network qualified). Each of these is used to direct the processing of commands to the correct system: distributed using Tivoli tasks or host using NetView command support. Does not apply to RODM resources. Valid only for instrumented resources.
%hb_origin%	An enum value pair list used to identify the system on which the component resides. The enumerations are listed in the AMS document in relation to the Discover Connections task in the Connection Type group (Primary). Does not apply to RODM resources. Valid only for instrumented resources.
%hb_primary%	First key value pair of the value of the *%hb_origin%* variable. Valid only for instrumented resources.
%hb_secondary%	First key value pair of the value of the *%hb_sub_origin%* variable. Valid only for instrumented resources.
%hb_source%	Contains a semicolon-delimited triplet with information from the ComponentID group in the applications global description file (GDF) (manufacturer;product;version). Does not apply to RODM resources. Valid only for instrumented resources.
%hb_sub_origin%	An enum value pair list used to differentiate between components on the same system that support the same relationship. Also described with discover connections (secondary). Does not apply to RODM resources. Valid only for instrumented resources.
%hb_sub_source%	Same format and attribute use as source, but from the component description file (CDF). The topology server uses this value to locate the correct CDF. Does not apply to RODM resources. Valid only for instrumented resources.
%ipaddress%	The value of the *%data3%* variable. Valid only for RODM resources.
%label%	The display name, which is the resource name assigned by the manager of the resource. This is the name that is displayed in the view. For RODM resources, this data is from the DisplayResourceName field.
%monitor%	The monitor name from the topology server database. Valid only for instrumented resources.
%objectid%	The NetView RODM object ID, if applicable.
%remoteconsole%	See "%REMOTECONSOLE%" on page 134 for more information.
%RODM.xxx%	The variable to be pulled from RODM. The character string following the period (*xxx*) is the field name to be pulled from RODM for the selected object. Use these only for resource specific commands. To extract just a period-delimited token from a field, append a colon (:) and a token number in the following way: `%RODM.xxx:3%` This extracts the third period-delimited token from the field.

Table 58. Common substitution variables (continued)

Substitution Variable	Description
%tme_oid%	The TME object ID, if available. Does not apply to RODM resources. Valid only for instrumented resources.

%REMOTECONSOLE%

The %REMOTECONSOLE% command-line substitution variable applies only to RODM resources. The value for %REMOTECONSOLE% comes from the resource specific data2 value, which comes from the DisplayResourceUserData field in RODM. This substitution variable calls a command or application on the topology console workstation.

For an example of how this substitution variable works with the Command Profile Editor and the topology console, see the command definition for Run Data2 Command in the %BINDIR%\TDS\server\sample\ihsscpe.*xxx*.rsp, where *xxx* is the country code, such as en_US. If syntax RemoteConsole=/(can be anything)/ exists in the the resource specific data2 value, then %REMOTECONSOLE% is substituted using the following rules:

- Syntax: "RemoteConsole = /the_command the_args(0-n)/"
- Fixed portions of this syntax are not case sensitive (such as RemoteConsole).
- Spaces around the equals sign are optional.
- Spaces between the first delimiter and the the_command are optional.
- The delimiter / can be any character. The first nonblank character after the equals sign becomes the delimiter.
- A second occurrence of the delimiter character must exist after the first occurrence.
- There must be a nonblank character between the delimiters.
- The command is assumed to be a valid command on any topology console workstation that runs this menu item. To map a command to an appropriate command for the platform of the topology console workstation, update the usercmdinv.properties file on the console. See Chapter 8, "Configuring Property Files for Locally Launched Applications," on page 69 for more information.
- This syntax can occur anywhere inside the resource specific data2 value; that is, other characters can come before or after this syntax.

The %REMOTECONSOLE% substitution variable is converted to the following string:

```
the_command d2cmdargs="the_arg1 the_arg2 the_arg3"
```

The fixed characters are d2cmdargs=" and the second double quotation mark after the last arg. All other values are obtained from between the two delimiters.

Part 4. Appendixes

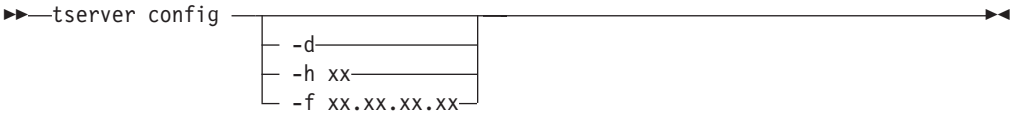
Appendix A. Topology Server Commands

The topology server commands provided in this appendix are intended to be used as reference material. Syntax diagrams are provided for each command.

config

Format

config



Purpose

User the **config** command to specify that the topology server processes are to be started as daemons. Also use this command to specify the heartbeat interval for the topology server.

Parameters

-d Specifies that the topology server processes be started as daemons. Updates are made to the appropriate system files so processes start automatically when the system is started.

This operand can be used only on the UNIX platform.

-h *xx*
Specifies the heartbeat interval for the topology server, where *xx* is the interval, in minutes, for generating heartbeats, such that the topology server resource in the topology display subsystem view remains in satisfactory status. The minimum is 5 minutes; the default value is 60 minutes unless it is overridden in the server.properties file.

-f *xx.xx.xx.xx*
Specifies an additional address by which your system is known. This can be used if you are using network address translation (NAT) and the topology server is being accessed by this method.

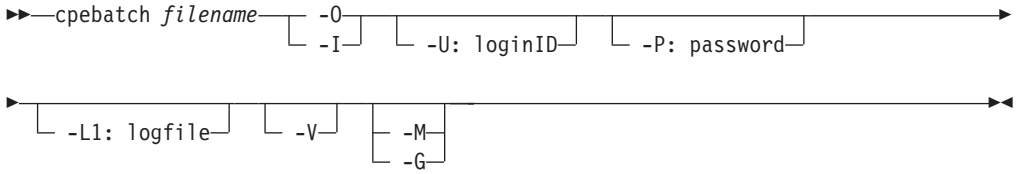
The variable, *xx.xx.xx.xx*, is the TCP/IP address in dot notation.

Note: The server must be rebooted after issuing this command in order for it to take effect.

cpebatch

Format

cpebatch



Purpose

Starts the command profile editor batch utility. For additional information about the command profile editor, see Chapter 13, "Using NetView Management Console Command Profiles," on page 119.

Parameters

filename
If the -O option is specified, this is the name of the output file that is created by the utility. If the -I option is specified, this is the name of the input file that is read by the utility. This parameter is required with the –I option.

-O
Generates a response file from the current commands database. The *filename* specifies the name of the file generated by the utility. If this is not a fully-qualified name, the file is generated in the current directory. If no name is specified, the default name of `IHSECPED.RSP` is used. The -O or -I parameter must be specified.

-I Specifies a response file to be used to update the current commands database. Unless the filename is fully-qualified, it is assumed to reside in the current directory. The -O or -I parameter must be specified.

-U *loginID*

If the user ID is not present in the command line, you are prompted to provide this information. The user ID must match your NetView user ID.

The login user ID must have administrative authority.

-P *password*
The password or password phrase for the login user ID that is specified by the -U parameter. There is no default value. If a password or password phrase is not in the command line, you are prompted to provide this information. This password or password phrase must match your NetView password or password phrase. If a value is not specified, the command profile editor utility tries to sign on to the topology server with a null password. This is an optional parameter.

-L1 *log file*
The name of the error log file to which you want to log informational and error messages. The default is IHSECPED.LOG. If the file is not fully-qualified, it is put in the following directories:

- For Windows: `%BINDIR%\TDS\server\log`

- For UNIX: `$BINDIR/TDS/server/log`

The error log file is continually appended, so that multiple runs of the utility are logged in the same file.

This is an optional parameter.

-V

Forces the utility into verify mode. The response file is processed and compared with the data in the topology server, but no changes are made to the actual topology server database. This is an optional parameter.

-M

Overrides the default add mode of the utility and forces the utility into modify mode. This enables information to be replaced in or deleted from the existing database. This is an optional parameter.

-G

Overrides the default add mode of the utility and forces the utility into modify mode. This enables updates to be made to command sets and profiles by adding to them without replacing existing information. This is an optional parameter.

Return codes

The command profile editor utility program generates the following return codes:

RC	Explanation
0	Successful. The utility completed successfully. Unless -V was specified in the **cpebatch** command, the database was updated.
4	Warning. The database was updated (unless -V specified) and warning messages were logged.
8	Error. The database was not updated and error messages were logged.
12	Severe Error. The database was not updated and a severe error ended the program immediately.

dbtransfer

Format

dbtransfer

Purpose

The **dbtransfer** command copies the topology server databases from the default installation directories to the location defined by the *TSERVER_DB* variable. After the *TSERVER_DB* variable is defined on the topology server workstation, this utility must be run before the topology server is started. This command does not change the contents of the topology server databases in the default installation directories.

getpd

Format

getpd

Purpose

Gathers information about your system environment, error logs, and trace files and stores them in the following file. You can send this file to IBM Software Support to help with problem determination.

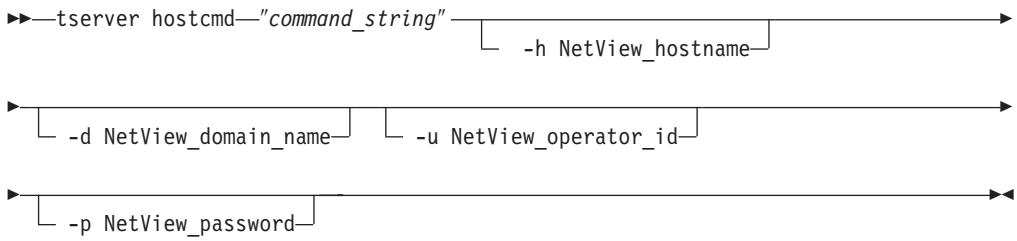
- For UNIX: $BINDIR/TDS/server/tmp/toposerv.xx.tar.Z
- For Linux for zSeries: $BINDIR/TDS/server/tmp/toposerv.xx.tar.gz

This command can be used only on the UNIX platform.

hostcmd

Format

hostcmd



Purpose

Issues commands to the NetView host from a command prompt on the topology server.

Parameters

"command string"
The command to be sent to the NetView host.

-h *NetView_hostname*
The IP address or host name of the NetView host where you want to issue the command.

-d *NetView_domain_name*
NetView domain name where you want to issue the command.

-u Specifies the NetView operator ID where you want to issue the command. This ID overrides any preset NetView operator ID (such as the ID that is set in the ihsshstc.cfg file or the **hostcmdoper** command).

If the **-u** operand is specified without the **-p** operand, you are prompted to enter the NetView password or password phrase.

-p Specifies the NetView password or password phrase where you want to issue the command. This password or password phrase overrides any preset NetView password or password phrase (such as the password or password phrase set in the ihsshstc.cfg file or **hostcmdoper** command).

If the **-p** operand is specified without the **-u** operand, you are prompted to enter the NetView operator ID.

Usage

The default is to run the command on the NetView host where the NETCONV session was initiated. See "Establishing Communication Between the NetView Host and the Topology Server" on page 78 for more information about setting up a NETCONV session.

The **hostcmd** command is issued from the command line or a script file. The response to the **hostcmd** command is displayed in the same command window you use to issue the command.

The **hostcmd** command is located in one of the following directories:

- For Windows systems: `%BINDIR%\TDS\server\bin`
- For UNIX systems: `$BINDIR/TDS/server/bin`

For UNIX or Windows systems, you can either change to this directory before running the **hostcmd** command, or add the directory path to your PATH environment variable.

For more information about presetting the NetView operator ID and password or password phrase, see "Defining the NetView for z/OS User ID and Password on the Topology Server" on page 17.

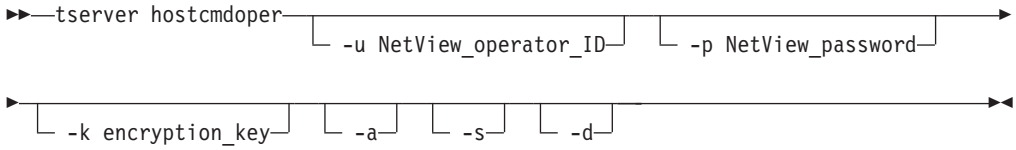
If you want to route command responses to a file or log, or to automate actions based on the command responses, you can customize the command exit source file, `ihsshstc.c`, located in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\sample`
- For UNIX: `$BINDIR/TDS/server/sample`

hostcmdoper

Format

hostcmdoper



Purpose

Issue this command before issuing the **hostcmd** command to set the NetView operator ID and password or password phrase. The **hostcmdoper** command can also be used to display or delete the NetView operator ID and password or password phrase in the ihsshiccc.cfg file.

Parameters

-u Specifies the NetView operator ID where you want to issue a **hostcmd** command.

-p Specifies the NetView password or password phrase where you want to issue the **hostcmd** command.

-k Specifies the new key for encrypting the NetView password or password phrase. This encryption key must be 32 characters or 64 hexadecimal characters in length.

Note: This operand cannot be specified with any other operand.

-a Specifies to migrate the password file to the new encryption level.

Note: This operand cannot be specified with any other operand.

-s Displays the NetView operator ID that is stored in the ihsshiccc.cfg file. This operand cannot be specified with any other operand.

-d Deletes the ihsshiccc.cfg file, which stores the NetView operator ID and password or password phrase. This operand cannot be specified with any other operand.

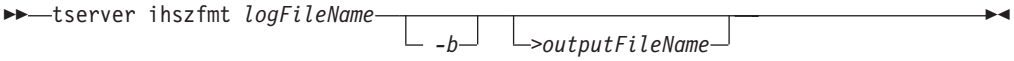
Usage

If the **hostcmdoper** command is never issued or if the **-d** option is run, and then the **hostcmd** command is run, the **hostcmd** command either uses the operator ID and password or password phrase that is stored in the ihsshstc.cfg file or prompts the user for the operator ID and password or password phrase. The password or password phrase is not encrypted in the ihsshstc.cfg file.

ihszfmt

Format

ihszfmt



Purpose

Formats the topology server error and trace logs. The output from the command is directed to `stdout`. The error logs and trace files are located in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\log`
- For UNIX: `$BINDIR/TDS/server/log`

Parameters

logFileName
Specifies the topology server error log or trace file to format. To format the error log, specify either `ihserror.log` or `ihserror.bak`. To format the trace log, specify either `ihstrace.log` or `ihstrace.bak`.

-b Specifies to suppress the formatting of the log in EBCDIC. This parameter does not affect the formatting of the log in ASCII. Formatting of the log in EBCDIC is important because the log contains data being sent between the topology server and the NetView host; thus, do not suppress this formatting.

outputFileName
Specifies the file name for the formatted error log or trace file.

ihszset

Format

ihszset

Purpose

Starts the command-line interface so that you can set the trace options for the topology server. If you do not specify an option and if the system is not a Linux system, **ihszset** starts the GUI interface so that you can set the trace options for the topology server.

Parameters

-help | -? | -h
Specifies a help menu to be displayed describing all the options you can specify with the **ihszset** command.

ihszsett

Format

ihszsett

Purpose

Starts the graphical user interface so that you can set the trace options for the topology server.

Note: This command is not available on the Linux for zSeries platform.

service

Format

service

Purpose

Sets up the topology server to run as a Windows service.

Parameters

account_name

If the service type is SERVICE_WIN32_OWN_PROCESS, this name is the account name in the form of 'DomanName\Username', which the service process logs on as when it runs. If the account belongs to the built-in domain, '.Username' can be specified. Services of type SERVICE_WIN32_SHARE_PROCESS are not a valid specification of an account other than LocalSystem. If NULL is specified, the service logs on as the 'LocalSystem' account, in which case the password parameter must be null.

If the service type is SERVICE_KERNAL_DRIVER or SERVICE_FILE_SYSTEM_DRIVER, this name is the driver object name (that is, '\FileSystem\Rdr' or '\Driver\Xns'), which the input and output (I/O) system uses to load the device driver. If NULL is specified, the driver is run with a default object name created by the I/O system, based on the service name.

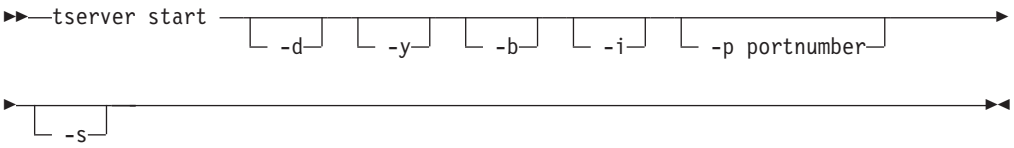
password

Contains the password to the account name specified by the IpServiceStartName parameter, if the service type is SERVICE_WIN32_OWN_PROCESS or SERVICE_WIN32_SHARE_PROCESS. If the pointer is NULL or if it points to an empty string, the service has no password. If the service type is SERVICE_KERNAL_DRIVER or SERVICE_FILE_SYSTEM_DRIVER, this parameter is ignored.

start

Format

start



Purpose

Starts the topology server and communications server, then displays a topology server window and a communications server window. If the topology server is already running either in the background as a service or daemon or in a Windows system, an additional instance of the topology server is not started.

Parameters

-d Specifies that the topology server be run in debug mode. Additional information is written to both the topology server window and to the topology server message log.

-y Generates the default topology server databases if no databases exist. Do not use unless requested by IBM Software Support.

-b Writes additional information to the message log when event data is received. This is helpful if instrumentation is being developed, since it displays the data received along with the information as it is processed.

-i Runs the topology server in IP mode only. Communications through LU 6.2 is not attempted.

Note: The server on a Linux for zSeries systems runs only in IP mode.

-p *portnumber*
Specifies which port is to be used for communications with the topology consoles. This does not override what is specified in the services file.

-s Starts the server processes in the background for a UNIX system. On a Windows system, this starts the server processes as a Windows service, assuming the server has been set up to run as a service.

Note: This parameter must be specified first.

Usage

The topology server and communications server windows only display messages. Use the **utility** command to manually write the topology server databases to disk, dump the server databases, dump the server semaphores, and so on.

stop

Format

stop

Purpose

Stops the topology server processes.

Parameters

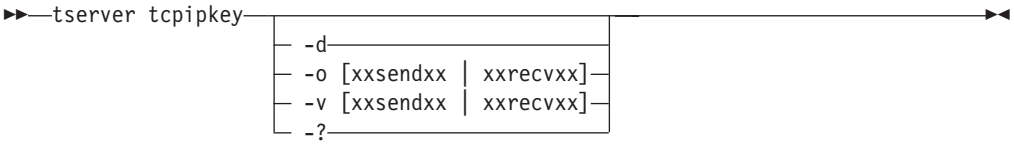
-f Use the force flag if one of the topology server processes ended abnormally or if the topology server is hung. The force flag stops any remaining topology server processes and then cleans up any remaining interprocess communications (IPC) resources.

Note: This is applicable only to the UNIX platform.

tcpipkey

Format

tcpipkey



Purpose

Specifies the send and receive keys used for encrypting and decrypting data sent or received by the workstation on the IP connection with the NetView host.

Parameters

-d Resets the keys to their default values.

-o Sets the keys used for the NETCONV connection with the NetView host. Enter the keys in the same format as they were entered in `DSITCPRF`. The first key is used to encrypt data sent from the workstation to the NetView host. The second key is used to decrypt data received by the workstation from the NetView host.

xxsendxx
Used to encrypt data sent from the topology server to the NetView host.

xxrecxx
Used to decrypt data received by the topology server.

-v Compares the two keys provided against the two NetView keys that are stored. If the keys match, `Yes` is returned. If the keys do not match, `No` is returned. If only one key is provided on the command line, both keys are prompted. If no keys are provided, the user is prompted for both keys.

xxsendxx
Used to encrypt data sent from the topology server to the NetView host.

xxrecxx
Used to decrypt data received by the topology server.

-? Displays the command syntax.

Usage

This command is used only for non-SSL (Secure Sockets Layer) connections.

The keys for this command must be either eight or sixteen characters long. If a sixteen-character key is entered, it is assumed to be a hexadecimal representation of the key and it is compressed down to eight bytes. If only one key is provided on the command line, then prompts are issued for both keys. If no keys are provided, then the user is prompted for both keys. The keys are stored in an encrypted format in the following file:

- For Windows: `%BINDIR%\TDS\server\config\ihssckey.cfg`
- For UNIX: `$BINDIR/TDS/server/config/ihssckey.cfg`

tserver

Format

tserver

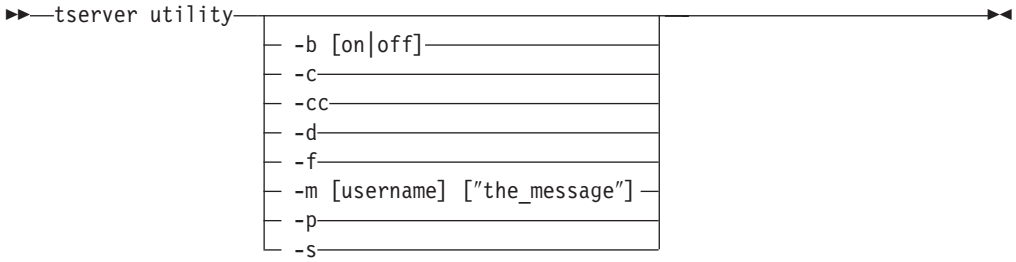
Purpose

Use **tserver** before some of the topology server commands (for example, **tserver stop**). The commands that require the **tserver** prefix are documented in this appendix. On the Windows platform, the topology server commands are a combination of command files and shell scripts. The **tserver** prefix ensures that the appropriate environment is set up and then starts the requested command.

utility

Format

utility



Purpose

Use this command to manually write the topology server databases to disk, dump the server databases, dump the server semaphores, or send a message to topology consoles signed on to the same topology server.

Parameters

-b [on | off]
Turns the instrumentation-related topology server message logging on or off.

-c Manually writes (checkpoints) the topology server databases to disk.

Note: After the command has completed, the current and backup copies of the topology server databases are identical and both contain the latest database updates.

-cc Manually writes (checkpoints) the topology server databases to the custom_backup directory.

-d Dumps the topology server databases to the following directory:

- For Windows: %BINDIR%\TDS\server\log
- For UNIX: $BINDIR/TDS/server/log

-f Dumps the topology server databases, without semaphore access protection, to the following directory:

- For Windows: %BINDIR%\TDS\server\log
- For UNIX: $BINDIR/TDS/server/log

-m Sends messages to topology consoles connected to the topology server.

user_name
The user name of the topology console to whom you want to send the message, or use **all** to broadcast the message to all topology consoles signed on to the same server.

"the_message"
The message to send to the topology console.

Note: The double quotations are required unless the message is a single token.

-p Displays the current settings of the server properties file to the topology server window and to the message log.

-s Dumps the server semaphores to the topology server window and to the message log.

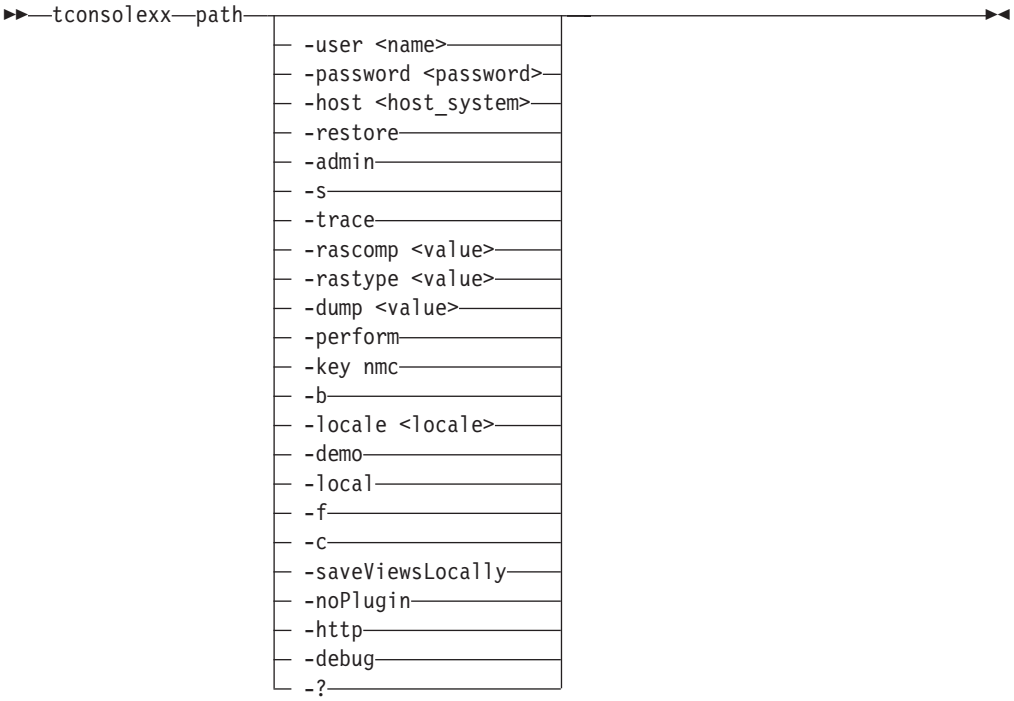
Appendix B. Topology Console Commands

The topology console commands provided in this appendix are intended to be used as reference material. The format in the following commands is in the form of syntax diagrams.

tconsolexx

Format

tconsolexx



Purpose

Starts the topology console from the command line. Start this directly from the topology console bin directory. Following are the possible values for xx:

NT Windows systems

.sh UNIX systems

Parameters

path
Specifies the top level of the topology console installation path:

- For Windows: `%BINDIR%\..\generic_unix\TDS\client`
- For UNIX: `$BINDIR/../generic_unix/TDS/client`

-user <user>
Specifies the topology console sign on user ID. See "Using the Topology Console Sign On Window" on page 79 for more information.

-password <password>
Specifies the topology console sign on password or password phrase. See "Using the Topology Console Sign On Window" on page 79 for more information.

-host <host_system>
Specifies the topology console sign on host name and possible port number. See "Using the Topology Console Sign On Window" on page 79 for more information.

-restore
Specifies the topology console sign on restore console attribute. See "Using the Topology Console Sign On Window" on page 79 for more information.

-admin
Specifies the topology console sign on administrator attribute. See "Using the Topology Console Sign On Window" on page 79 for more information.

-s Automatically signs on using the specified options, previously saved values, or both. See "Using the Topology Console Sign On Window" on page 79 for more information.

-trace
This option is for IBM Software Support only. It enables default tracing.

-rascomp <value>
This option is for IBM Software Support only. The <value> variable can be obtained from the Service page of the Console Properties notebook.

-rastype <value>
This option is for IBM Software Support only. The <value> variable can be obtained from the Service page of the Console Properties notebook.

-dump <value>
This option is for IBM Software Support only.

-perform
This option is for IBM Software Support only. Enables performance tracing.

-key nmc
Optional keyword.

-b Use buffered tracing.

-locale
Override the default locale. The format for *locale* follows:

```
langCode [countryCode]
```

-demo
Starts the topology console disconnected from the topology server.

-local
Starts the topology console disconnected from the topology server.

-f Suppresses automatic synchronization of support files from the topology server.

-c Suppresses automatic synchronization of topology console code from the topology server. This is not recommended.

-saveViewsLocally
When Save View Customization is used while a connection to a topology server exists, the view is saved in a standalone file on the topology console workstation. This option is used only for capturing live views for use in a demonstration.

-noPlugin
Suppresses loading of any plug-ins.

-http
Forces the starting of the Web server regardless of the current console property settings. This option enables another application to set up the NetView management console environment before sending locate-resource requests.

-debug
Enables built-in debugging code. This option is for IBM Software Support only.

-? Display the command-line usage.

Usage

All scripts support the following optional environment variables:

TCONSOLE_BACKDOOR
Java code library (or libraries) placed at front of CLASSPATH. This option is for IBM Software Support only.

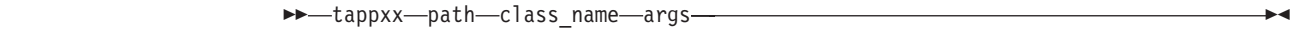
TCONSOLE_CLASSPATH
Java code library (or libraries) placed at end of CLASSPATH. Typically, this is used to define other Java classes to be accessed by the NetView management console.

TCONSOLE_JAVAOPTS
Command-line arguments for Java program. This option is for IBM Software Support only.

tappxx

Format

tappxx



Purpose

Starts the topology console utility functions. Start this directly from the topology console bin directory. Following are the possible values for xx:

NT Windows systems

.sh UNIX systems

Parameters

path
Specifies the top level of the topology console installation path:

- For Windows: `%BINDIR%\..\generic_unix\TDS\client`
- For UNIX: `$BINDIR/../generic_unix/TDS/client`

class_name
Specifies the class name for the topology console provided utility.

args
Specifies any arguments that are required for the utility.

Usage

All scripts support the following optional environment variables:

TCONSOLE_BACKDOOR
Java code library (or libraries) placed at front of CLASSPATH. This option is for IBM Software Support only.

TCONSOLE_CLASSPATH
Java code library (or libraries) placed at end of CLASSPATH. Typically, this is used to define other Java classes to be accessed by NetView management console.

TCONSOLE_JAVAOPTS
Command-line arguments for Java program. This option is for IBM Software Support only.

Appendix C. Launching and Using the NetView Management Console from Other Applications

This appendix describes how to launch and use the NetView management console from other applications using servlets, command-line programs, and scripts provided with the NetView management console. It includes information about the servlets, setting up the NetView management console to use the servlets, and the command-line programs that call the servlets.

Servlets

Two servlets provided with the NetView management console can be used by other applications:

- The check-Web-server servlet, **IhsRunning**, enables other applications to check whether the NetView management console Web server is running.
- The locate-resource servlet, **IhsLocRes**, enables other applications to have the NetView management console locate a specific resource. If the resource is found, a view containing the specified resource is opened on the NetView management console topology console to which the request is sent.

The default security access for each servlet can be overridden with a *servlet_name*.access attribute in the defaultscheme.properties file. The defaultscheme.properties file is located on the NetView management console topology server in the following directory: <installRoot>\bin\<interp>\TDS\Server\db\current\settings.

The values for servlet security access are shown in Table 59, where:

local Indicates that the servlet can be run only from the local host.

any Indicates that the servlet can be run from any host.

disabled
Indicates that the servlet cannot be run.

Table 59. Changing servlet security access

defaultscheme.properties Attribute	**Default Value**	**Other Valid Value**
com.tivoli.ihs.servlet.IhsLocRes.access	local	any, disabled
com.tivoli.ihs.servlet.IhsRunning.access	local	any, disabled

Setup

For other applications to use the servlets to launch the NetView management console:

1. The NetView management console must be started.
2. The Web server must be started in one of these ways:
   - Checking the **Automatically start web server** check box on the Web Server page in the Console Properties notebook.
   - Using the **http** parameter with **tconsolexx** to force the starting of the Web server; for more information, see "tconsolexx" on page 158.
   - Clicking on the **Start web server now** button on the Web Server page in the Console Properties notebook.

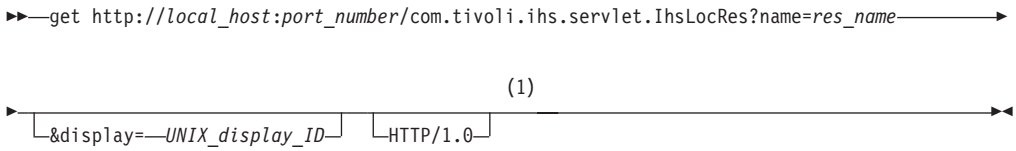
Using the Servlets

This section describes the syntax for using the servlets directly, and the programs that provide a front end to using the servlets.

IhsLocRes Servlet

Format

IhsLocRes



Notes:

1 The **name** and **display** parameters can be in any order.

Purpose

This is an HTTP get request that uses the **IhsLocRes** servlet to locate a resource.

Parameters

local_host:*port_number*
Specifies the host name and port number of the NetView management console Web server.

com.tivoli.ihs.servlet.IhsLocRes
Specifies the **IhsLocRes** servlet.

name=*res_name*
Specifies the name of the resource to be located.

display=*UNIX_display_ID*
Specifies the display ID of the NetView management console. This parameter is required for UNIX, and must match the *DISPLAY* environment variable on the UNIX system.

HTTP/1.0
If specified, indicates that response codes are to be returned.

Return codes

If the HTTP/1.0 parameter is used, these response codes can be returned:

200 The request has been accepted.

400 The required servlet name was not specified or an incorrect value was specified.

403 The request was not made from the NetView management console host system or the NetView management console is currently disabled.

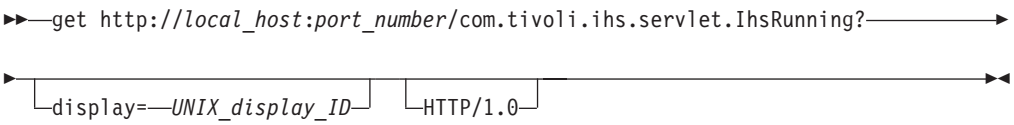
409 The display ID specified does not match the display ID of the NetView management console.

500 An internal error occurred.

IhsRunning Servlet

Format

IhsRunning



Purpose

This is an HTTP get request that uses the **IhsRunning** servlet to check whether or not the NetView management console Web server is running.

Parameters

local_host:*port_number*
Specifies the host name and port number of the NetView management console Web server.

com.tivoli.ihs.servlet.IhsRunning
Specifies the IhsRunning servlet.

display=*UNIX_display_ID*
Specifies the display ID of the NetView management console. This parameter is required for UNIX, and must match the *DISPLAY* environment variable on the UNIX system.

HTTP/1.0
If specified, indicates that response codes are to be returned.

Return codes

If the HTTP/1.0 parameter is used, these response codes can be returned:

200 The NetView management console Web server is running.

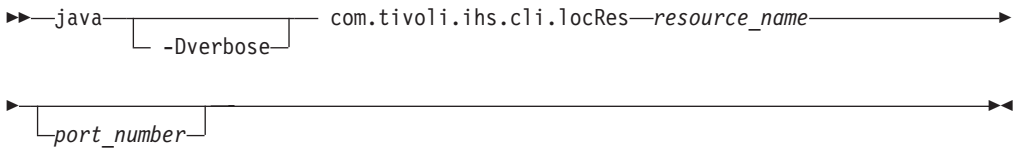
403 The request was not made from the NetView management console host system or the NetView management console web server is currently disabled.

409 The display ID specified does not match the display ID of the NetView management console.

locRes Java Class

Format

LocRes



Purpose

The **locRes** Java class uses the **IhsLocRes** servlet to locate a resource.

Parameters

–Dverbose
Specifies that debugging information, including the return code, is to be printed.

com.tivoli.ihs.cli.locRes
Specifies the **locRes** Java class, which uses the **IhsLocRes** servlet.

resource_name
Specifies the name of the resource to be located.

port_number
Specifies the port number of the NetView management console Web server. The default is port 80.

Return codes

0 The request was accepted.

1 The request was rejected because the Web server is not running, access is denied, or the resource name is not valid.

2 A command-line argument is in error.

Usage

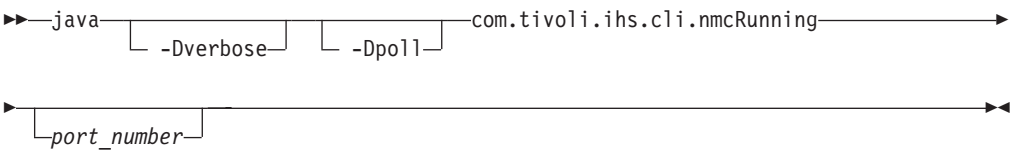
This program is started from the command line or can be run using the **tappxx** scripts; for information about **tappxx**, see "tappxx" on page 161. To use the Java command, you must install your own Java environment.

If you use the **tappxx** script, the **–Dverbose** parameter must be specified with the **TCONSOLE_JAVAOPTS** environment variable, rather than on the command line.

nmcRunning Java Class

Format

nmcRunning



Purpose

The **nmcRunning** Java class uses the **IhsRunning** servlet to check whether the NetView management console Web server is running.

Parameters

–Dverbose
Specifies that debugging information, including the return code, is to be printed.

–Dpoll
Specifies that the program is to poll 10 times before stopping. A ten-second wait occurs between each poll. If this parameter is not specified, the program polls only once.

com.tivoli.ihs.cli.nmcRunning
Specifies the **nmcRunning** program, which uses the **IhsRunning** servlet.

port_number
Specifies the port number of the NetView management console Web server. The default is port 80.

Return codes

0 The request was accepted.

1 The request was rejected because the Web server is not running or access is denied.

2 A command-line argument is in error.

Usage

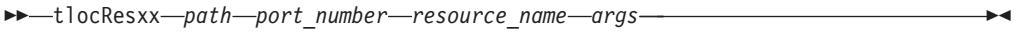
This program is started from the command line or can be run using the **tappxx** scripts; for information about **tappxx**, see “tappxx” on page 161. To use the Java command, you must install your own Java environment.

If you use the **tappxx** script, the **–D** parameters must be specified with the **TCONSOLE_JAVAOPTS** environment variable, rather than on the command line.

tlocResxx Script

Format

tlocResxx



Purpose

This script submits a locate-resource request to the NetView management console Web server, starting the NetView management console topology console, if necessary, with the appropriate **tconsolexx** script. The possible values for *xx* are:

NT Windows systems

.sh UNIX systems

Parameters

path
Specifies the top level of the topology console installation path:

- For Windows: `%BINDIR%\..\generic_unix\TDS\client`
- For UNIX: `$BINDIR/../generic_unix/TDS/client`

port_number
Specifies the port number of the NetView management console Web server.

resource_name
Specifies the name of the resource to be located.

Note: If the % (percent sign) character is included in the string for the resource name, you need to include two percent sign (%%) characters as the first is removed on the Windows platform.

args
Specifies optional **tconsolexx** start-up arguments. These arguments are used if the NetView management console topology console has to be started.

Usage

This script supports the following optional environment variables:

TCONSOLE_BACKDOOR
Java code libraries placed at the front of CLASSPATH. This option is for IBM Software Support only.

TCONSOLE_CLASSPATH
Java code libraries placed at the end of CLASSPATH. Typically, this is used to define other Java classes to be accessed by the NetView management console topology console.

TCONSOLE_JAVAOPTS
Command-line arguments for the Java program. This option is for IBM Software Support only.

Appendix D. Sending Commands to Multiple NetView Domains

When sending certain commands from the topology console, such as **Activate**, **Inactivate**, or **Recycle**, to the NetView host, there might be more than one NetView domain to run the command against. This occurs only when you right click a Systems Network Architecture topology manager resource with a command indicator value of 32769 and the command is to be run at NetView for z/OS with the IHSXTHCE command exit. For example, when you select an SNA topology manager resource against which to issue a command, more than one NetView domain might be monitoring that resource. In this case, a multiple domain dialog box is displayed, so you can select one or more NetView domains. The following example shows the multiple domain dialog box.

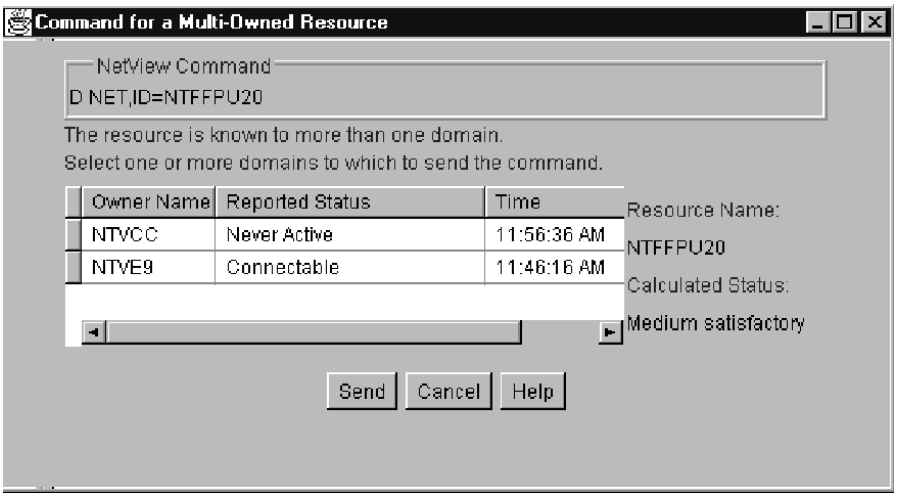


Figure 36. Multiple Domain Dialog Box Example

You can also specify one or more NetView domains as a default. In this case, when a command is issued against a resource, only the NetView domains specified as the default receive the command.

To specify a NetView domain as the default, follow these steps:

1. Create one or more files named:
   - **userMultiDom.properties** where user is the user name of an operator. Use this file if you want to specify one or more NetView domains as a default for a specific operator.
   - **DefaultMultiDom.properties** to specify one or more NetView domains as the default for all operators.

   The following example shows these properties:

   ```
   autoSelect.1 = NTA09
   autoSelect.2 = NTM12
   autoSelect.3 = NTM13
   autoSend = true
   ```

   When `autoSend = true`, commands are run on the NetView domains specified in the properties file without any operator interaction (the multiple domain

dialog box is not displayed). All other NetView domains are ignored. A value of `autoSend = false` indicates that the multiple domain dialog box should be displayed for the operator with the domains specified in the properties file automatically selected.

2. Store these files on the topology server workstation in one of the following directories:
   - For Windows: `%BINDIR%\TDS\server\db\current\settings`
   - For UNIX: `$BINDIR/TDS/server/db/current/settings`

Appendix E. Auditing and the ihsaudit.xml File

The NetView management console includes auditing capability and enables a log of various actions taken on the NetView management console topology console and NetView management console topology server which can be used as an audit trail. This auditing function is based on and operated by the topology server. An XML (Extensible Markup Language) file, `ihsaudit.xml`, that is located in the `%BINDIR%/TDS/server/log` directory contains the auditing information. The `ihsaudit.xml` file can be edited by an ASCII text editor. This log can be displayed in a Web browser, or other application, which supports XML.

Note: For display from a Web browser, the browser must support either CSS (Cascading Style Sheet) or XSL (Extensible Style sheet Language) to enable the display of the log data from the `ihsaudit.xml` file.

The following NetView management console functions can be audited:

- Starting and stopping of the NetView management console topology server
- Using the NetView NETCONV command to start and stop communication between the host and server
- Signing on and off of the NetView management console topology console
- Signing on and off of the command profile editor (CPE) batch utility
- Starting commands
- Command responses
- Open view requests, including locate resource requests
- Open view responses
- Closing views
- View Customization
- Flag change requests, including note changes and list suspended resources requests
- Send message requests
- Aggregation priority and threshold change requests

To specify which functions are to be audited, edit the `server.properties` file located in the `%BINDIR%\TDS\server\config` directory. For more details on editing this file, see “Auditing Functions and the Server.Properties File” on page 19. The size of the `ihsaudit.xml` file can also be set. For more information about this function in the `server.properties` file, see “Customizing the Size of the Audit Log” on page 22.

Elements in the ihsaudit.xml file

The audit entry elements in the `ihsaudit.xml file` are defined in a data type definition (DTD) file, `ihsaudit.dtd`, which is shipped with the NetView management console topology server as a sample. The `ihsaudit.dtd` file is located in one of the following directories:

- For Windows: `%BINDIR%\TDS\server\sample`
- For UNIX: `$BINDIR/TDS/server/sample`

Note: Editing this file is not recommended.

The ihsaudit.dtd file lists the data elements contained in the ihsaudit.xml file. A description of each of these elements follows:

action ELEMENT

Format

```
<!ELEMENT action
  (#CDATA) >
```

Description

General purpose, further qualifies the *action* that occurred for this audit entry. See "Audit Entry Actions" on page 188 for the possible actions for each audit entry.

Parents

Aggregation, command, cpe, flag, netconv, nmcConsole, nmcServer, sendMsg, viewCust, viewNav

Children

None

admin ELEMENT

Format

```
<!ELEMENT admin
   (#CDATA) >
```

Description

Determines, during sign on and sign off, if administrator access was requested. Will be either yes or no.

Parents

nmcConsole

Children

None

aggPri ELEMENT

Format

```
<!ELEMENT aggPri
   (#CDATA) >
```

Description

Aggregation priority was changed for a real resource.

Parents

Aggregation

Children

None

aggregation ELEMENT

Format

```
<!ELEMENT aggregation
  (action,id,ipAddr,rodmId,((aggPri)|(thresholdDeg,thresholdSevDeg,
   thresholdUnsat))) >
```

Description

Qualifies an auditEntry to represent a change to the aggregation priority of a real resource or a change to the aggregation thresholds of an aggregate resource.

Parents

auditEntry

Children

One each of (action, id, ipAddr, rodmId). One each of (aggPri) for a real resource and one each of (thresholdDeg, thresholdSevDeg, thresholdUnsat) for an aggregate resource.

auditEntry ELEMENT

Format

```
<!ELEMENT auditEntry
  (date,
  (aggregation|command|cpe|flag|netconv|nmcConsole|nmcServer|sendmsg|
   viewCust|viewNav)) >
```

Description

Wrappers one audit log entry.

Parents

auditLog

Children

One date and one of the following elements: aggregation, command, cpe, flag, netconv, nmcConsole, nmcServer, sendMsg, viewCust, viewNav

auditLog ELEMENT

Format

```
<!ELEMENT auditLog
  (auditEntry+)  >
```

Description

Wrappers the entire audit log.

Parents

None

Children

One (or more) auditEntry (or entries).

cmd ELEMENT

Format

```
<!ELEMENT cmd
   (#CDATA) >
```

Description

Contains the command string to be processed. Before a command string is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters found in the command response string that cannot be browsed is translated to a character format that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

command

Children

None

cmdResp ELEMENT

Format

```
<!ELEMENT cmdResp
   (#CDATA) >
```

Description

Contains the command responses. Before a command response string is written to the audit log, the following `server.properties` attributes are checked in the following order:

- **`auditCarriageReturnReplacement`**

  If the `auditCarriageReturnReplacement` attribute is set to 1, then any carriage return (0x0D), line feed (0x0A), or carriage return immediately followed by a line feed (0x0D0A) found in the command response string is replaced with the characters specified by `server.properties` attribute `auditCarriageReturnReplacementValue`.

  For more information, see "Customizing the Replacement Character for Carriage Returns and Line Feeds" on page 23.

- **`auditTranslateToBrowseableChars`**

  If the `auditTranslateToBrowseableChars` attribute is set to 1, then a subset of characters found in the command response string that cannot be browsed is translated to a format which can be browsed. The following characters are examples:

  – `&` is changed to `&`

  – `<` is changed to `<`

  – `>` is changed to `>`

  For more information, see "Customizing Special Characters to be Browsed" on page 23.

- **`auditCommandResponseMaxSize`**

  If the `auditCommandResponseMaxSize` attribute is not equal to 0 (zero), then the command response string is truncated to the maximum length specified by the attribute.

  For more information, see "Customizing the Length of Command Responses within the Audit Log" on page 22.

Parents

command

Children

None

command ELEMENT

Format

```
<!ELEMENT command
  (action,id?,ipAddr?,corrId,res?,rodmId?,menuText?,(cmd|cmdResp))   >
```

Description

Qualifies an auditEntry to represent either a command to be started or the command responses.

Parents

auditEntry

Children

One each of the following elements: action, corrId, (cmd|cmdResp). Optionally, one each of the following elements: id, ipAddr, res, rodmId, menuText.

corrId ELEMENT

Format

```
<!ELEMENT corrId
   (#CDATA) >
```

Description

Contains a numeric correlation ID used to correlate a command that is started with the corresponding command responses.

Parents

command

Children

None

cpe ELEMENT

Format

```
<!ELEMENT cpe
    (action,id) >
```

Description

Qualifies an auditEntry to represent a command profile editor (CPE) batch utility sign on.

Parents

auditEntry

Children

One each of action and id.

date ELEMENT

Format

```
<!ELEMENT date
    (#CDATA) >
```

Description

Timestamp of the audit log entry in the following format: `YYYY-MM-DD HH:MM:SS`

Parents

auditEntry

Children

None

extSearch ELEMENT

Format

```
<!ELEMENT extSearch
    (#CDATA) >
```

Description

When a locate resource request is performed, indicates if the extended search option was selected. Will be either `yes` or `no`

Parents

viewNav

Children

None

flag ELEMENT

Format

```
<!ELEMENT flag
    (action,id,ipAddr,((flagMask,flagValue)|(noteMask,note))?)
  >
```

Description

Qualifies an auditEntry to represent one of the following items:

- Change to a flag, or flags, of a resource
- Change to a note of a resource
- List suspended resources request

If several notes are changed at one time, a separate auditEntry exists for each changed note.

Parents

auditEntry

Children

One each of the following elements: action, id, ipAddr. Optionally, one each of the following elements: flagMask, flagValue, noteMask, note.

flagMask ELEMENT

Format

```
<!ELEMENT flagMask
    (#CDATA) >
```

Description

Contains a hexadecimal 4-byte field, with one or more bits turned on. It represents the flag or flags that changed (not the value to which the flag changed). These bit values are defined by the UserStatus field in RODM. An example is `0x40000000`, which indicates that the suspended flag is being changed.

Parents

flag

Children

None

flagValue ELEMENT

Format

```
<!ELEMENT flagValue
    (#CDATA) >
```

Description

Contains a hexadecimal 4-byte field, with one or more bits turned on or off. This represents the value to which the flag or flags are being changed. The value in the flagMask field represents the flag or flags being set. These

values are defined by the UserStatus field in RODM. An example is `0x40000000`, which turns on the suspended flag.

Parents

flag

Children

None

fromHostname ELEMENT

Format

```
<!ELEMENT fromHostname
    (#CDATA) >
```

Description

Contains a TCP host name from which a message is sent. This is in the context of the send message function. If the message is sent from a topology console, fromHostname contains the TCP host name of the topology console that sent the message. If the message is sent from the topology server, fromHostname contains the TCP host name of the Tivoli NetView for z/OS program connected with the topology server.

Parents

sendMsg

Children

None

fromId ELEMENT

Format

```
<!ELEMENT fromId
    (#CDATA) >
```

Description

Contains the user name (a Tivoli NetView for z/OS logon ID) from which a message is being sent. This is in the context of the send message function.

Parents

sendMsg

Children

None

fromIpAddr ELEMENT

Format

```
<!ELEMENT fromIpAddr
    (#CDATA) >
```

Description

Contains the IP address of the TCP/IP host name from which a message is sent. This is in the context of the send message function. If the message is sent from a topology console, fromIpAddr contains the IP address of the TCP host name of the topology console that sent the message. If the message is sent from the topology server, fromIpAddr contains the IP address of the TCP host name of the Tivoli NetView for z/OS program connected with the topology server.

Parents

sendMsg

Children

None

hostname ELEMENT

Format

```
<!ELEMENT hostname
    (#CDATA) >
```

Description

Contains a TCP host name.

Parents

netconv, nmcConsole

Children

None

id ELEMENT

Format

```
<!ELEMENT id
    (#CDATA) >
```

Description

Contains a user name (a Tivoli NetView for z/OS logon ID).

Parents

aggregation, command, cpe, flag, nmcConsole, viewCust, viewNav

Children

None

initRes ELEMENT

Format

```
<!ELEMENT initRes
    (#CDATA) >
```

Description

Contains the initiating resource name; for example, the RODM object ID of the resource that was selected, to navigate to another view.

Parents

viewNav

Children

None

ipAddr ELEMENT

Format

```
<!ELEMENT ipAddr
    (#CDATA) >
```

Description

Contains an IP address.

Parents

aggregation, command, flag, netconv, nmcConsole, viewCust, viewNav

Children

None

locateName ELEMENT

Format

```
<!ELEMENT locateName
    (#CDATA) >
```

Description

Contains the name of the resource that was specified in the Locate Resource dialog box. Before a locate name is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters found in the name that cannot be browsed is translated to a format that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

viewNav

Children

None

lu62name ELEMENT

Format

```
<!ELEMENT lu62name
    (#CDATA) >
```

Description

Contains the LU 6.2 address.

Parents

netconv

Children

None

menuText ELEMENT

Format

```
<!ELEMENT menuText
    (#CDATA) >
```

Description

Contains the menu text of the command being started. Before a menu text is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters that cannot be browsed is found in the menu text and translated into characters that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

command

Children

None

msg ELEMENT

Format

```
<!ELEMENT msg
    (#CDATA) >
```

Description

Contains the message that was sent, using the send message function. Before a message is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters in the message that cannot be browsed is found and translated into characters that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

sendMsg

Children

None

netconv ELEMENT

Format

```
<!ELEMENT netconv
    (action,nvDomain,((ipAddr,hostname)|(lu62name))) >
```

Description

Qualifies an auditEntry to represent a change in the NETCONV connection between the Tivoli NetView for z/OS program and the NetView management console topology server (from the perspective of the NetView management console topology server).

Parents

auditEntry

Children

One each of the following elements: action, nvDomain. Optionally, if the NETCONV connection is with IP, then one each of the ipAddr and hostname elements, and, if the NETCONV connection is with LU 6.2., then one lu62name element.

nmcConsole ELEMENT

Format

```
<!ELEMENT nmcConsole
    (action,id,ipAddr,hostname,admin) >
```

Description

Qualifies an auditEntry to represent the sign on or sign off of a NetView management console.

Parents

auditEntry

Children

One each of the following elements: action, id, ipAddr, hostname, admin.

nmcServer ELEMENT

Format

```
<!ELEMENT nmcServer
    (action) >
```

Description

Qualifies an auditEntry to represent the start or stop of the NetView management console topology server.

Parents

auditEntry

Children

One of action.

note ELEMENT

Format

<!ELEMENT note (#CDATA) >

Description

Contains the free-form note text that was changed for the flag of a resource. If the note was made null, this element exists with no data in it. The value in the noteMask field represents the flag (bit) to which this note pertains. Before a note is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters in the note that cannot be browsed is found and translated into characters that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

flag

Children

None

noteMask ELEMENT

Format

```
<!ELEMENT noteMask
    (#CDATA) >
```

Description

Contains a hexadecimal 4-byte field, with one bit turned on, that represents

the note that was changed. An example is `0x40000000`. This indicates that the note for the suspended flag is changed.

Parents
flag

Children
None

nvDomain Element

Format
```
<!ELEMENT nvDomain
    (#CDATA) >
```

Description
Contains a Tivoli NetView for z/OS domain ID.

Parents
netconv

Children
None

res ELEMENT

Format
```
<!ELEMENT res
    (#CDATA) >
```

Description
Contains the display name of a resource at the server.

Parents
command

Children
None

rodmId ELEMENT

Format
```
<!ELEMENT rodmId
    (#CDATA) >
```

Description
Contains the hexadecimal RODM object id, for example, `0000000E00000466`.

Parents
aggregation, command, flag

Children
None

sendMsg ELEMENT

Format
```
<!ELEMENT sendMsg
    (action,fromId,fromIpAddr,fromHostname,(toId,toIpAddr,toHostname)?,
    msg) >
```

Description
Qualifies an auditEntry to represent a message that was sent, using the send message function.

Parents

auditEntry

Children

One each of the following elements: action, fromId, fromIpAddr, fromHostname, msg. Optionally, when a specific user is targeted (not a broadcast message), one each of the following elements: toId, toIpAddr, toHostname.

thresholdDeg ELEMENT

Format

```
<!ELEMENT thresholdDeg
    (#CDATA) >
```

Description

The degraded threshold was changed for an aggregate resource.

Parents

aggregation

Children

None

thresholdSevDeg ELEMENT

Format

```
<!ELEMENT thresholdSevDeg
    (#CDATA) >
```

Description

The severely degraded threshold was changed for an aggregate resource.

Parents

aggregation

Children

None

thresholdUnsat ELEMENT

Format

```
<!ELEMENT thresholdUnsat    (#CDATA) >
```

Description

The unsatisfactory threshold was changed for an aggregate resource.

Parents

aggregation

Children

None

toHostname ELEMENT

Format

```
<!ELEMENT toHostname
    (#CDATA) >
```

Description

Contains a TCP host name to which a message is sent. This is in the context of the send message function. If the message is sent to a topology console, toHostname contains the TCP host name of the topology console

that received the message. If the message is sent to the topology server, toHostname contains the TCP host name of the Tivoli NetView for z/OS program connected with the topology server.

Parents

sendMsg

Children

None

toId ELEMENT

Format

```
<!ELEMENT toId
    (#CDATA) >
```

Description

Contains the user name (a Tivoli NetView for z/OS logon ID) to which a message is sent. This is within the context of the send message function.

Parents

sendMsg

Children

None

toIpAddr ELEMENT

Format

```
<!ELEMENT toIpAddr
    (#CDATA) >
```

Description

Contains the IP address of the TCP host name to which a message is sent. This is in the context of the send message function. If the message is sent to a topology console, toIpAddr contains the IP address of the TCP host name of the topology console that received the message. If the message is sent to the topology server, toIpAddr contains the IP address of the TCP host name of the Tivoli NetView for z/OS program connected with the topology server.

Parents

sendMsg

Children

None

viewCust ELEMENT

Format

```
<!ELEMENT viewCust
    (action,id,ipAddr,viewName) >
```

Description

Qualifies an auditEntry to represent a view that was customized and saved at the NetView management console topology server. The view customization was saved, removed, or deleted.

Parents

auditEntry

Children

One each of the following elements: action, id, ipAddr, viewName.

viewName ELEMENT

Format

```
<!ELEMENT viewName
    (#CDATA) >
```

Description

Contains a name of a view. Before a view name is written to the audit log, if the `auditTranslateToBrowseableChars` attribute in the `server.properties` file is set to 1, then a subset of characters in the view name that cannot be browsed is found and translated to a format that can be browsed. The following characters are examples:

- `&` is changed to `&`
- `<` is changed to `<`
- `>` is changed to `>`

For more information, see "Customizing Special Characters to be Browsed" on page 23.

Parents

viewCust, viewNav

Children

None

viewNav ELEMENT

Format

```
<!ELEMENT viewNav
    (action,id,ipAddr,viewType,((viewName)|(initRes)|(locateName,
    extSearch))) >
```

Description

Qualifies an auditEntry to represent view navigation. Examples include: opening a network or exception view, double-clicking a resource for more detail, configuration parents, locating a resource, closing a view.

Parents

auditEntry

Children

One each of action, id, ipAddr, viewType. One of the following elements:

- One each of locateName and extSearch
- One of initRes
- One of viewName

viewType ELEMENT

Format

```
<!ELEMENT viewType
    (#CDATA) >
```

Description

Contains the type of view.

Parents

viewCust, viewNav

Children

None

Audit Entry Actions

The actions for the elements are explained in the following sections. All of the audit entries are *children* of the auditEntry element. Though action elements are strings, they are considered an extension of the auditEntry element and are, therefore, not translated. If the attribute (as shown in the following tables) in the `server.properties` file is equal to 1 (one), then the corresponding action is written to the audit log.

aggregation ELEMENT

The action element data for an `aggregation` audit entry is described in Table 60.

Table 60. Description of the aggregation action element

Action element data	Other elements	Description	Attribute
`priority`	**`id`** User name **`ipAddr`** IP address of console **`rodmId`** RODM object ID **`aggPri`** Aggregation priority value	Aggregation priority is updated for this resource.	auditAggregationChange
`threshold`	**`id`** User name **`ipAddr`** IP address of console **`rodmId`** RODM object ID **`thresholdDeg`** Degraded threshold value **`thresholdSevDeg`** Severely degraded threshold value **`thresholdUnsat`** Unsatisfactory threshold value	At least one of the aggregation threshold fields is updated for this resource.	auditAggregationChange

command ELEMENT

The action element data for a `command` audit entry is described in Table 61 on page 189.

Table 61. Description of the command action element

Action element data	Other elements		Description	Attribute
execution	**id** **ipAddr** **corrId** **res** **rodmId** **menuText** **cmd**	User name IP address of console, if not a tserver hostcmd Correlation ID to correlate commands to their responses Resource name if command is resource dependent RODM object ID if command is resource dependent Menu text of command being processed Command string of the command being processed	A command is being initiated. Commands that are run in the Tivoli NetView for z/OS program are also logged at the NetView program; the auditing of commands and command responses, at the NetView management console server, has no effect on the netlog of Tivoli NetView for z/OS.	auditCommand
response	**id** **ipAddr** **corrId** **cmdResp**	User name, if not a tserver hostcmd response IP address of console, if not a tserver hostcmd response Correlation ID to correlate commands to their responses Command response string	The command response string. There might be one or more audit log entries for command responses, depending on how the target of the command provides chunks of the response. Commands that are started in Tivoli NetView for z/OS are also logged at the NetView program; the auditing of commands and command responses at the NetView management console server has no effect on netlog of Tivoli NetView for z/OS.	auditCommandResponse

cpe ELEMENT

The action element data for a cpe audit entry is described in Table 62.

Table 62. Description of the cpe action element

Action element data	Other elements		Description	Attribute
signon	**id**	User name	Successful CPE signon	auditCPESignonSignoff
signonFailed	**id**	User name	Unsuccessful CPE signon	auditCPESignonSignoff
signoff	**id**	User name	CPE signoff	auditCPESignonSignoff

flag ELEMENT

The action element data for a flag audit entry is described in Table 63 on page 190.

Table 63. Description of the flag action element.

Action element data	Other elements	Description	Attribute
changeFlag	**id** User name **ipAddr** IP address of console **rodmId** RODM object ID **flagMask** *0xFFFFFFFF* value with bits on to represent the flags being changed **flagValue** *0xFFFFFFFF* value with value of the bits specified by flagMask	A flag (UserStatus field in RODM) was set on a resource. Multiple flags can be set on the resource; all changes are represented in this audit log entry.	auditFlagChange
changeNote	**id** User name **ipAddr** IP address of console **rodmId** RODM object ID **noteMask** *0xFFFFFFFF* value with bits on to represent the note being changed **note** Note was changed on the bit represented by noteMask for this resource.	A note (UserStatusNote field in RODM) was set on a resource. Multiple notes can be set on the resource; a separate audit log entry is written for each note.	auditFlagChange
listSuspended	**id** User name **ipAddr** IP address of console	A list suspended resources request occurred. This audit entry does not indicate if the request was successful.	auditFlagChange

netconv ELEMENT

The action element data for a netconv audit entry is described in Table 64.

Table 64. Description of the netconv action element

Action element data	Other elements	Description	Attribute
up	**nvDomain** NetView domain ID **ipAddr** NetView IP address if over IP **hostname** NetView TCP host name if over IP **lu62name** LU name if over LU 6.2	NETCONV start	auditNetconvUpDown

Table 64. Description of the netconv action element (continued)

Action element data	Other elements	Description	Attribute
down	**nvDomain** NetView domain ID **ipAddr** NetView IP address if over IP **hostname** NetView TCP host name if over IP **lu62name** LU name if over LU 6.2	NETCONV stop	auditNetconvUpDown

nmcConsole ELEMENT

The action element data for an nmcConsole audit entry is described in Table 65.

Table 65. Description of the nmcConsole action element

Action element data	Other elements	Description	Attribute
signon	**id** User name **ipAddr** IP address of console **hostname** TCP host name of console **admin** Either yes or no if administrator access was requested	Successful NetView management console topology console signon	auditConsoleSignonSignoff
signonFailed	**id** User name **ipAddr** IP address of console **hostname** TCP host name of console **admin** Either yes or no if administrator access was requested	Unsuccessful NetView management console topology console signon	auditConsoleSignonSignoff
signoff	**id** User name **ipAddr** IP address of console **hostname** TCP host name of console **admin** Either yes or no if administration access was requested	NetView management console topology console signoff	auditConsoleSignonSignoff

nmcServer ELEMENT

The action element data for an nmcServer auditEntry is described in Table 66 on page 192.

Table 66. Description of the nmcServer action element

Action element data	Description	Attribute
`start`	NetView management console topology server is started	auditServerStartStop
`initialized`	NetView management console topology server is initialized	auditServerStartStop
`stop`	NetView management console topology server is stopped	auditServerStartStop

sendMsg ELEMENT

The action element data for a `sendMsg` audit entry is described in Table 67.

Table 67. Description of the sendMsg action element

Action element data	Other elements	Description	Attribute
`broadcast`	**`fromId`** User name of the console sending the message or *topology server* if message was initiated from the NetView management console topology server **`fromIpAddr`** IP address of the console or server sending the message **`fromHostname`** Host name of the console or server sending the message **`msg`** Message text that was sent	A broadcast message was sent.	auditSendMessage

Table 67. Description of the sendMsg action element (continued)

Action element data	Other elements	Description	Attribute
specificUser	**fromId** User name of the console sending the message or *topology server* if message was initiated from the NetView management console topology server **fromIpAddr** IP address of the console or server sending the message **fromHostname** Host name of the console or server sending the message **toId** Targeted console user. Can also be ″topology server″ if targeted to the NetView management console topology server. **toIpAddr** Targeted console IP address **toHostname** Targeted console host name. **msg** Message text that was sent	A specific message was sent.	auditSendMessage

viewCust ELEMENT

The action element data for a viewCust audit entry is described in Table 68.

Table 68. Description of the viewCust action element

Action element data	Other elements	Description	Attribute
save	**id** User name **ipAddr** IP address of console **viewName** View name	The customization of a view was saved in the database of the NetView management console server.	auditViewCustomizationSaveRemove
remove	**id** User name **ipAddr** IP address of console **viewName** View name	The customization of a view was removed from the databases of the NetView management console server.	auditViewCustomizationSaveRemove

Table 68. Description of the viewCust action element (continued)

Action element data	Other elements	Description	Attribute
delete	**id** User name **ipAddr** IP address of console **viewName** View name	A customized view was deleted.	auditViewCustomizationSaveRemove

viewNav ELEMENT

The action element data for a viewNav audit entry is described in Table 69.

Table 69. Description of the viewNav action element

Action element data	Other elements	Description	Attribute
openRequest	**id** User name **ipAddr** IP address of console **viewType** Type of view requested, (for example: locateResource, moreDetail, configParents, configChildren, configLogPhy, configLog, ConfigPhy, configBackbone, fastpath, network, exception, configPeers) **initRes, viewName or locateName** For network and exception views, the audit entry contains viewName which is the name of the view. For Locate Resource, the audit entry contains locateName, which is the resource name as entered by the operator, for example, RALV4. For the more detail, configuration or fastpath views, the audit entry contains initRes which is the RODM id of the initiating resource name. **extSearch** For locate resource requests, indicates if extended search was requested. Valid values are yes or no.	A view request was sent to Tivoli NetView for z/OS.	auditViewOpenRequest

Table 69. Description of the viewNav action element (continued)

Action element data	Other elements	Description	Attribute
openResponse	**id** User name **ipAddr** IP address of console **viewType** Type of view request for example: moreDetail, configParents, configChildren, configPeers, configLogPhy, configLog, configPhy, configBackbone, network, exception, fastpath. **viewName** View name	A RODM-based view was opened.	auditViewOpenResponse
open	**id** User name **ipAddr** IP address of console **viewType** Type of view request, for example server. **viewName** View name	A server-based view was opened.	auditViewOpenRequest or auditViewOpenResponse
close	**id** User name **ipAddr** IP address of console **viewType** Type of view, for example: moreDetail, configParents, configChildren, configPeers, configLogPhy, configLog, configPhy, configBackbone, network, exception, fastpath. **viewName** View name	A view was closed.	auditViewClose

Appendix F. Automatic File Download at Console Log On

This appendix describes how files are downloaded from the server to the console when the console signs on to the server.

The installation of the NetView management console topology console occurs in the following two phases:

1. During the installation process
2. The first time the console logs on to the server

During Installation

During the installation process, the smallest possible set of files is installed on the local workstation. This includes the following types of files:

- NetView management console topology console code
- MRI (readable strings) for the appropriate language
- A subset of background, help, and icon files

Note: The exception to the rule of installing the smallest possible set of files on the workstation during installation occurs when a custom install is performed and the Productivity Kit is selected. This installs all files necessary to run the NetView management console topology console in demonstration mode (not requiring a NetView management console topology server).

During Initial Sign On

The first time the console signs on to the server following installation, files are automatically downloaded from the following server directories:

- For Windows: %BINDIR%/TDS/server/db/current/
- For UNIX: $BINDIR$\TDS\server\db\current

These directories and their descriptions follow:

backgrounds
View background maps/images

bin NetView management console topology console binary support files including scripts

icons Resource and company icons

help Online help support files including:
- NetView management console product specific help files
- Customer Java application help files

lib Java code including the following files:
- NetView management console topology console Java code (`ihseuc.jar`)
- NetView management console product personality file (`nmc.properties`)
- Customer Java application and plug-in JAR files

During Subsequent Sign On

The files downloaded during the initial sign on are automatically checked for updates each time the console successfully signs onto the server. That is, when the console downloads a file, the console stores the time stamp of the file on the server workstation. If the time stamp has changed, the file is downloaded again.

If the console connects to the same server, these files are updated only when changes have been made. However, if the console connects to different servers on different platforms, the time stamps across servers is close, but not identical. Therefore, a value in the `defaultscheme.properties` file can be set so that you can control exactly when the download occurs. This control applies when reconnecting to the same server or to different servers. See "Customizing the Automatic Download of Files At Log On" on page 37 for more detailed information.

The `lib` directory is handled differently than the others. Files installed by the Console, and named in the `contents.properites` file in the `lib` directory, are only downloaded if a different build of the Console is placed in this directory. That is, the time stamps of these files are not cross-checked. The time stamp cross-check does occur for files in the lib directory which were provided by the customer. However, the `defaultscheme.properties` file has not yet been read in; therefore control over when these files are downloaded cannot be customized. See the `defaultscheme.properties` file for more information and for the internal value that the Console uses for the time stamp cross-check.

When a file in the `lib` directory needs to be downloaded, message `IHS1137` is displayed and the file must be downloaded before sign on can continue. These files might need to be reloaded into memory. For files in all other directories, the file is downloaded and sign on processing continues automatically.

See Appendix B, "Topology Console Commands," on page 157 for more information about command-line arguments to suppress this downloading process.

Appendix G. MultiSystem Manager Resource Information Window–Other Data Field

This appendix lists the MultiSystem Manager resource types for the MultiSystem Manager Open Topology Interface and Tivoli Management Region agents, and the contents of the `Other data` field for each resource type.

Note: For information about the MultiSystem Manager resource types for the MultiSystem Manager IBM Tivoli Network Manager and IP agents, see *IBM Tivoli NetView for z/OS IP Management*.

The following tables contain the contents of the `Other data` field in the NetView management console Resource Information window for each of the MultiSystem Manager resource types. This data is stored in the `DisplayResourceOtherData` field in the Resource Object Data Manager (RODM) for each object.

If more than one field is listed, the fields are listed in the order in which they are displayed.

Open Topology Interface Agent

Table 70. Open Topology Interface Other Data Fields on Resource Information Windows

Resource Type	Other Data Field Contents
Group	None
Network aggregate	Agent Application= Agent Level=
Manager	SNA Address= Agent Application= Agent Level=
IP system aggregate	IP Address=[2] [3] OS= OS Ver= OS Release=
LAN workstation aggregate	MAC Address=[1] [3] OS= OS Ver= OS Release=
Open system aggregate	MAC Address=[1] [2] [3] IP Address= OS= OS Ver= OS Release=

Table 70. Open Topology Interface Other Data Fields on Resource Information Windows (continued)

Resource Type	Other Data Field Contents

Notes:
1. When the topology correlation function is used, a typical value example is shown in the following way:
`MAC address=4000A17D006,OS=NT,OS Ver=4.00`
2. When the topology correlation function is used, a typical example value is shown in the following way:
`IPAddress=9.37.36.7,OS=NT,OS Ver=4.00`
3. When the topology correlation function is used, the maximum example value is shown in the following way:
`MAC address=40000A17D006,IP address=9.37.36.7,OS=NT,OS VER=4.00,`
`Segment No.=SEG100B,IPX address=00004444.40000A17D006,`
`SNA Node=PU4657FA,IP HostName=GSMEYERS,Location=E214/503`	

Tivoli Management Region Agent

Table 71. Tivoli Management Region Other Data Fields on Resource Information Windows

Resource Type	Other Data Field Contents
TME 10 Networks - Group	None
TME 10 Network - Network aggregate	None
TME 10 Mgr - Manager	Hostname= Port= TME Agent Level= Operating System= IP Address= OSERV process is Up/Down Tivoli Enterprise Console process is UP
Policy Region Aggregate	Policy Region=
Managed Region Aggregate	TMR Number=
IP system aggregate	IP Address=[2] [3] OS= OS Ver= OS Release=
LAN workstation aggregate	MAC Address=[1] [3] OS= OS Ver= OS Release=
Indicator	Hostname= IP Address= Resource Model=
Open system aggregate	MAC Address=[1] [2] [3] IP Address= OS= OS Ver= OS Release=
Profiles	Hostname= IP Address=

Table 71. Tivoli Management Region Other Data Fields on Resource Information Windows (continued)

Resource Type	Other Data Field Contents
Resource models	Hostname= IP Address= Profile=

Notes:

1. When the topology correlation function is used, a typical value example is shown in the following way:

   ```
   MAC address=4000A17D006,OS=NT,OS Ver=4.00
   ```

2. When the topology correlation function is used, a typical example value is shown in the following way:

   ```
   IPAddress=9.37.36.7,OS=NT,OS Ver=4.00
   ```

3. When the topology correlation function is used, the maximum example value is shown in the following way:

   ```
   MAC address=40000A17D006,IP address=9.37.36.7,OS=NT,OS VER=4.00,
   Segment No.=SEG100B,IPX address=00004444.40000A17D006,
   SNA Node=PU4657FA,IP HostName=GSMEYERS,Location=E214/503
   ```

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

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

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Programming Interfaces

This publication primarily documents information that is NOT intended to be used as Programming Interfaces of Tivoli NetView for z/OS. This publication also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of Tivoli NetView for z/OS. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

```
Programming Interface information
End of Programming Interface information
```

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml .

Adobe and Acrobat and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel is a registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and Windows NT 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.

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

Index

Special characters

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Y

Program Number: 5697-ENV

Printed in USA

GC31-8852-03