



LogMatrix
NerveCenter

Integrating NerveCenter with a Network Management Platform

**UNIX and Windows
Version 5.1.0***

Copyright

Portions ©1989-2011 LogMatrix, Inc. All rights reserved.

Disclaimers

LogMatrix, Inc. (“LogMatrix”) makes no representations or warranties, either expressed or implied, by or with respect to anything in this manual, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

These applications are available through separate, individual licenses. Not every feature or application described herein is licensed to every customer. Please contact LogMatrix if you have licensing questions.

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, photocopying, recording or otherwise, without prior written consent of LogMatrix. While every precaution has been taken in the preparation of this book, LogMatrix assumes no responsibility for errors or omissions. This publication and the features described herein are subject to change without notice.

The program and information contained herein are licensed only pursuant to a license agreement that contains use, reverse engineering, disclosure and other restrictions.

Trademarks

LogMatrix is registered in the U.S. Patent and Trademark Office. NerveCenter and the LogMatrix Logo are trademarks of LogMatrix, Inc.

All other products or services mentioned in this manual may be covered by the trademarks, service marks, or product names as designated by the companies who market those products.

LogMatrix, Inc.
4 Mount Royal Ave, Suite 250
Marlborough, MA 01752
Toll Free +1 (800) 892-3646
Phone +1 (508) 597-5300
Fax +1 (774) 348-4953
info@logmatrix.com
<http://www.logmatrix.com>

1 Introduction

Overview of this book	2
NerveCenter Documentation	3
Using the Online Help	3
Printing the Documentation	3
The NerveCenter Documentation Library	4
UNIX Systems	5
Document Conventions	5
Documentation Feedback	6
LogMatrix Technical Support	6
Professional Services	6
Educational Services	7
Contacting the Customer Support Center	7
For Telephone Support	7
For E-mail Support	7
For Electronic Support	7
For Online KnowledgeBase Access	7
For User Community Access	8

2 Integrating NerveCenter with a Network Management Platform

NerveCenter and Network Management Platforms	10
Integrating NerveCenter with Network Management Platforms	11
OVPA Integration	13
Co-resident Installation	14
UNIX Installation	15
Windows installation on separate machines	16
How NerveCenter Integration Helps the Platform	17
Universal Platform Adapter Integration	19
IBM Tivoli Netcool/OMNibus	19

3 Integrating with HP OpenView Network Node Manager

The OpenView Platform Adapter	22
Enabling and Disabling the Platform Adapter	22
Enabling the NerveCenter OpenView Platform Adapter	22
Disabling the NerveCenter OpenView Platform Adapter	23
Starting and Stopping the OpenView Platform Adapter	25
Using OpenView as a Node Source	26
Populating the Node List with OpenView as a Data Source	27
Synchronizing with HP OpenView Network Node Manager	29
Filtering Nodes from HP OpenView Network Node Manager	30
Filtering by Node Capabilities	30
Filtering by Node System Object Identifiers	31
Filtering by Node IP Addresses	31
Identifying Parent-Child Relationships	31
Sending NerveCenter Informs to OpenView	34
The Reliability of NerveCenter Informs Sent to OpenView	37
Saving NerveCenter Informs Until Acknowledgement	39
Configuring the Inform Queue Depth	41
Configuring OpenView to Integrate with NerveCenter	43
Configuring a Node Map with NerveCenter Alarm Severity Colors	43
Mapping NerveCenter Alarm Severity Colors to OpenView	44
Modifying NerveCenter-to-OpenView Alarm Severity Color Mapping	45
Reconfiguring OpenView before Removing the NerveCenter OpenView Platform Adapter	47
OpenView Integration Reference	49
Command line reference for the NerveCenter OpenView Platform Adapter	49
Variable Bindings for NerveCenter Informs	53

4 Integrating NerveCenter with IBM Tivoli Netcool/OMNIbus

Overview	56
What is Netcool/OMNIbus?	57
What is NerveCenter?	58
Behavior Models	58
Alarms	59
Inform Messages	60
How NerveCenter Complements Netcool/OMNIbus	60
Smart Polling	60
Intelligent Correlation	61
Distributed Architecture	61
Other Advantages	62

Integrating Netcool with the NerveCenter Universal Platform Adapter	62
Components Required for Integration	63
IBM Tivoli Netcool/OMNIBus Components	64
LogMatrix NerveCenter Components	65
How Integration Components Interact	65
NerveCenter Configuration Settings	69
NerveCenter Server Inform Port Settings	70
How to Specify the Destination of Inform Packets Sent to IBM Tivoli Netcool/OMNIBus 70	
Universal Platform Adapter Settings	73
How to Start and Stop the Universal Platform Adapter	74
Starting and Stopping the Universal Platform Adapter in UNIX	74
Starting and Stopping Platform Integration with IBM Tivoli Netcool/OMNIBus in Win- dows	75
Inform Action Settings	76
Netcool/OMNIBus Configuration Settings	78
Object Server Data Management Settings	78
NerveCenter Probe Settings	79
NerveCenter.props file	79
NerveCenter.rules file	80
Desktop settings	84
Filtered Event Lists	85
Custom Views	86
Alert classes	87
Automated Actions	89
Objective View Map	90
Netcool Integration Reference	92
Command line reference for Integrating NerveCenter with Netcool	92
Variable Bindings for NerveCenter Informs	94
Inform Data Sent from NerveCenter	95
Debug Probe Output	97
Sample NerveCenter.rules File	98
Index	111



Welcome to *Integrating NerveCenter with a Network Management Platform*. This chapter introduces the audience and purpose of this guide, and how you can best use it.

One of the best features of NerveCenter™ is its ability to integrate with a variety of network management platforms. This book provides the following information:

- ◆ Description of foundational platform integration concepts, such as:
 - ◆ Using a network management platform as a node source
 - ◆ The difference between a SNMP trap and a NerveCenter inform
- ◆ Step-by-step explanation of common integration procedures, such as:
 - ◆ How to start and stop the appropriate NerveCenter platform adapter
 - ◆ How to declare the machine hosting your platform as an inform recipient
- ◆ Frequently referenced material, such as:
 - ◆ The syntax for common NerveCenter commands
 - ◆ Information stored in NerveCenter inform packets

This chapter includes the following sections:

Section	Description
<i>Overview of this book on page 2</i>	Includes an overview of the contents of this guide and what you need to know before you use the guide.
<i>NerveCenter Documentation on page 3</i>	Lists and describes the components of the LogMatrix NerveCenter support system, including printed guides, online guides, help, and links to the LogMatrix NerveCenter Web site and the LogMatrix technical support Web site.
<i>LogMatrix Technical Support on page 6</i>	Describes how to access the NerveCenter knowledge base and other LogMatrix support services.

Overview of this book

Because most users will be interested in only one network management platform at a time, this integration guide is arranged by platform. Each chapter address issues specific to that platform.

**NOTE**

The procedures found in this integration guide assume NerveCenter and at least one supported network management platform have been installed on your network. For more information, see *Installing NerveCenter*.

This book contains the following chapters:

Title	Description
<i>Chapter 2, Integrating NerveCenter with a Network Management Platform</i>	Provides an overview of NerveCenter integration and discusses some basic integration concepts.
<i>Chapter 3, Integrating with HP OpenView Network Node Manager</i>	Discusses NerveCenter integration with HP OpenView Network Node Manager.
<i>Chapter 4, Integrating NerveCenter with IBM Tivoli Netcool/OMNIBus</i>	Discusses NerveCenter integration with IBM Tivoli Netcool/OMNIBus.

NerveCenter Documentation

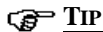
This section describes the available NerveCenter documentation, which explains important concepts in depth, describes how to use NerveCenter, and provides answers to specific questions.


The documentation set is provided in online (HTML) format, as well as PDF for printing or on-screen viewing. See the following topics for more information:

- ◆ *Using the Online Help on page 3*
- ◆ *Printing the Documentation on page 3*
- ◆ *The NerveCenter Documentation Library on page 4*
- ◆ *UNIX Systems on page 5*
- ◆ *Document Conventions on page 5*
- ◆ *Documentation Feedback on page 6*

Using the Online Help

You can view the documentation with browsers such as Microsoft Internet Explorer or Firefox. Refer to the *NerveCenter Release Notes* for the browser versions supported with this release.

**TIP**

For in-depth instructions on using the online documentation, click the Help button  in the upper right of the Help window.

Printing the Documentation

The NerveCenter documentation is also available as Portable Document Format (PDF) files that you can open and print. All PDF files are located in your *installpath/doc* directory.

**NOTE**

You must have Adobe Acrobat Reader to open or print the PDF files. You can download the Reader free from Adobe's Web Site at www.adobe.com.

The NerveCenter Documentation Library

The following documents ship with NerveCenter.

Book Title	Description	Application	Audience	PDF for Print
<i>NerveCenter Release Notes</i>	Describes new NerveCenter features and includes late-breaking information, software support, corrections, and instructions.	All	All	relnotes.pdf
<i>Installing NerveCenter</i>	Helps you plan and carry out your NerveCenter upgrades and new installations. Use the <i>Release Notes</i> in conjunction with this book.	All	Installation team	install.pdf
<i>Managing NerveCenter</i>	Explains how to customize and tune NerveCenter after it has been installed.	NerveCenter Administrator	Administrator	managing_nervecenter.pdf
<i>Integrating NerveCenter with a Network Management Platform</i>	Explains how to integrate NerveCenter with network management platforms.	NerveCenter Administrator	Administrator	integratingNC.pdf
<i>Learning How to Create Behavior Models</i>	Provides step-by-step instructions and examples for creating behavior models.	NerveCenter Client	Users with administrative privileges	learningModel.pdf
<i>Designing and Managing Behavior Models</i>	Explains behavior models in depth, how to create or modify models, and how to manage your models.	NerveCenter Client	Users with administrative privileges	designingModels.pdf
<i>Monitoring Your Network</i>	Explains how NerveCenter works and how you can most effectively monitor your network.	NerveCenter Client and Web Client	Users	monitoringNet.pdf
<i>Behavior Models Cookbook</i>	Describes each behavior model shipped with LogMatrix NerveCenter.	NerveCenter Client	Users with administrative privileges	modsCookbook.pdf
Quick reference cards	Quick reference cards provide convenient reference material for common NerveCenter tasks.	NerveCenter Client and Administrator	All	quickreference.pdf

UNIX Systems

On UNIX systems, NerveCenter man pages provide command reference and usage information that you view from the UNIX shell as with other system man pages. When you specify documentation during NerveCenter installation, the script installs nroff-tagged man pages and updates your system's MANPATH environment variable to point to the NerveCenter man page directory.

Document Conventions

This document uses the following typographical conventions:

Element	Convention	Example
Key names, button names, menu names, command names, and user entries	Bold	Press Tab Enter ovpa -pc
<ul style="list-style-type: none"> ◆ A variable you substitute with a specific entry ◆ Emphasis ◆ Heading or Publication Title 	<i>Italic</i>	Enter <i>./installdb -f IDBfile</i>
Code samples, code to enter, or application output	Code	<code>iifInOctets > 0</code>
Messages in application dialog boxes	Message	Are you sure you want to delete?
An arrow (>) indicates a menu selection	>	Choose Start > Programs > OpenService NerveCenter
A link to a section in the same book	<i>Blue Italic</i>	For more information, see <i>Correlating Conditions</i> .
A link to a section in a different book	<i>Green Italic</i>	For more information, see <i>Correlating Conditions</i> in <i>Monitoring Your Network with NerveCenter</i> .
<p>Note: If you are using a PDF viewer, you may need to use the Go to Previous View button to return to the original PDF file.</p>		



CAUTION

A caution warns you if a procedure or description could lead to unexpected results, even data loss, or damage to your system. If you see a caution, proceed carefully.

**NOTE**

A note provides additional information that might help you avoid problems, offers advice, and provides general information related to the current topic.

**TIP**

A tip provides extra information that supplements the current topic. Often, tips offer shortcuts or alternative methods for accomplishing a task.



If toolbar buttons are available, they are displayed in the margin next to the step in which you can use them. Other shortcuts are noted as tips. Also, shortcut (accelerator) keys are displayed on application menus next to their respective options.

Documentation Feedback

LogMatrix, Inc. is committed to providing quality documentation and to helping you use our products to the best advantage. If you have any comments or suggestions, please send your documentation feedback to:

Documentation
LogMatrix, Inc.
6'O qwpvTq{ criC xg. "Uwkg"472
Marlborough, MA 01752

documentation@logmatrix.com

LogMatrix Technical Support

LogMatrix is committed to offering the industry's best technical support to our customers and partners. You can quickly and easily obtain support for NerveCenter, our proactive IT management software.

Professional Services

LogMatrix offers professional services when customization of our software is the best solution for a customer. These services enable us, in collaboration with our partners, to focus on technology, staffing, and business processes as we address a specific need.

Educational Services

LogMatrix is committed to providing ongoing education and training in the use of our products. Through a combined set of resources, we can offer quality classroom style or tailored on-site training.

Contacting the Customer Support Center

For Telephone Support

Phone: 1-800-892-3646 or 1-508-597-5300

For E-mail Support

E-mail: techsupport@logmatrix.com.

For Electronic Support

LogMatrix has a Web-based customer call tracking system where you can enter questions, log problems, track the status of logged incidents, and check the knowledge base.

When you purchased your product and/or renewed your maintenance contract, you would have received a user name and password to access the LogMatrix Call Tracking System using Salesforce. You may need to contact your contracts or NerveCenter administrator for the username and password for your account with Salesforce.

If you have not received or have forgotten your log-in credentials, please e-mail us with a contact name and company specifics at techsupport@logmatrix.com.

We are committed to providing ongoing education and training in the use of our products. Through a combined set of resources, we offer quality training to our global customer base.

For Online KnowledgeBase Access

For additional NerveCenter support information, please go the LogMatrix website www.logmatrix.com for access to the following sections of information:

- ◆ **Patches and Updates** - latest installation files, patches, and updates including documentation for NerveCenter.
- ◆ **Software Alerts** - latest software alerts relative to NerveCenter.

- ◆ **KnowledgeBase Search** - search the NerveCenter KnowledgeBase for answers to your questions whether relating to the installation, usage, or operation of NerveCenter.

For User Community Access

You can seek as well as share advice and tips with other NerveCenter users at <http://community.logmatrix.com/LogMatrix/>

Integrating NerveCenter with a Network Management Platform

One of the best features of NerveCenter™ is its ability to integrate with a variety of network management platforms. This introductory chapter explains some basic NerveCenter concepts and includes the following sections:

Section	Description
<i>NerveCenter and Network Management Platforms on page 10</i>	Describes the role NerveCenter plays in a network management strategy.
<i>Integrating NerveCenter with Network Management Platforms on page 11</i>	Explains that NerveCenter can receive node data from certain platforms as well as send inform notifications to platforms.

NerveCenter and Network Management Platforms

Today's LANs and WANs typically involve a conglomeration of devices from a multitude of vendors connected across vast geographical distances. As businesses become increasingly dependent on complex networks, the dependency on proper management of these networks also increases.

Network administrators frequently rely on network management platforms to discover a network's topology as well as the current status of managed nodes. Although NerveCenter can operate as a stand-alone network management product, many customers integrate NerveCenter with their network management platform to provide additional functionality and control. NerveCenter offers advanced features often weak or lacking in network management platforms. These powerful NerveCenter features include:

- ◆ An advanced polling engine that polls only when required, thus reducing the amount of network traffic devoted to management.
- ◆ Intelligent event correlation based on finite state machines that alarm only after all the necessary conditions are met.
- ◆ Reliable notification actions that use TCP/IP to inform management platforms rather than the less reliable protocol of SNMP.

Integrating NerveCenter with Network Management Platforms

The two primary methods of integrating NerveCenter with a network management platform are:

- ◆ NerveCenter can receive node information from a network management platform
A network management platform can add some or all of its managed nodes to NerveCenter's node list. Afterwards, NerveCenter remains synchronized with the platform to insure the information is as accurate as possible.
A network management platform can also provide parent-child relationship information for any NerveCenter running the Downstream Alarm Suppression behavior models.
- ◆ NerveCenter can send informs to a network management platform
NerveCenter users can design behavior models that will inform their network management platforms. Although NerveCenter can send SNMP traps, it can also send an Inform packet via TCP/IP which tends to be more reliable.

Table 2-1 illustrates how NerveCenter integrates with supported platforms:

TABLE 2-1. How NerveCenter integrates with your network management platform

Platform	Node data	Inform recipient
HP OpenView Network Node Manager	✓	✓
IBM Tivoli Netcool/OMNIBus		✓

Using one of NerveCenter’s platform adapters—OVPA or the Universal Platform Adapter—you can integrate NerveCenter with a network management platform. The type and extent of integration varies with the platform you’re using. [Table 2-2](#) lists the platforms supported along with the adapter and level of integration for each.

TABLE 2-2. NerveCenter Platforms, Adapters, and Integration Level

Platform	Adapter and level of integration
<p>The following have full integration with NerveCenter:</p> <ul style="list-style-type: none"> ◆ Hewlett Packard OpenView Network Node Manager 	<p>NerveCenter uses the OVPA adapter to do the following:</p> <ul style="list-style-type: none"> ◆ Extract node and topology information ◆ Insert events into event console ◆ Change map symbol colors
<p>The following receive events from NerveCenter:</p> <ul style="list-style-type: none"> ◆ IBM Tivoli Netcool/OMNIBus 	<p>NerveCenter uses the Universal Platform adapter to insert events into the event console.</p>

NerveCenter can operate with the network management platforms in various combinations, for example, by integrating with both OpenView Network Node Manager and IBM Tivoli Netcool/OMNIBus. You can also integrate multiple NerveCenters with one or more platforms. Each NerveCenter might monitor one or more subnets and provide event information to the platform.

The following sections describe integration configurations for each of the NerveCenter adapters.

OVPA Integration

NerveCenter's OVPA platform adapter enables integration with Hewlett Packard's OpenView Network Node Manager. In this configuration, the platform typically provides node information to NerveCenter.

**NOTE**

NerveCenter can monitor nodes that are outside the platform's domain or that are not managed by the platform. Such nodes can be added manually to the NerveCenter database, discovered from traps, or loaded using ImportUtil.

OVPA retrieves node and topology information from the platform and forwards this information to the NerveCenter Server. The Server can be configured to monitor all platform nodes, a subset of nodes, or nodes that have certain capabilities or system object IDs (OID). NerveCenter in turn forwards noteworthy events back to the platform by sending informs to OVPA, which relays them to the platform's event console.

When NerveCenter sends informs to the platform, if color changes are required, OVPA sends a message to NerveCenter's NCApp process, which then forwards instructions for color changes to the platform map.

Co-resident Installation

NerveCenter and the platform can have a co-resident installation, that is, they can be installed on the same system. Co-resident installations are supported both on Windows and UNIX.

Figure 2-1 shows a typical co-resident platform integration.

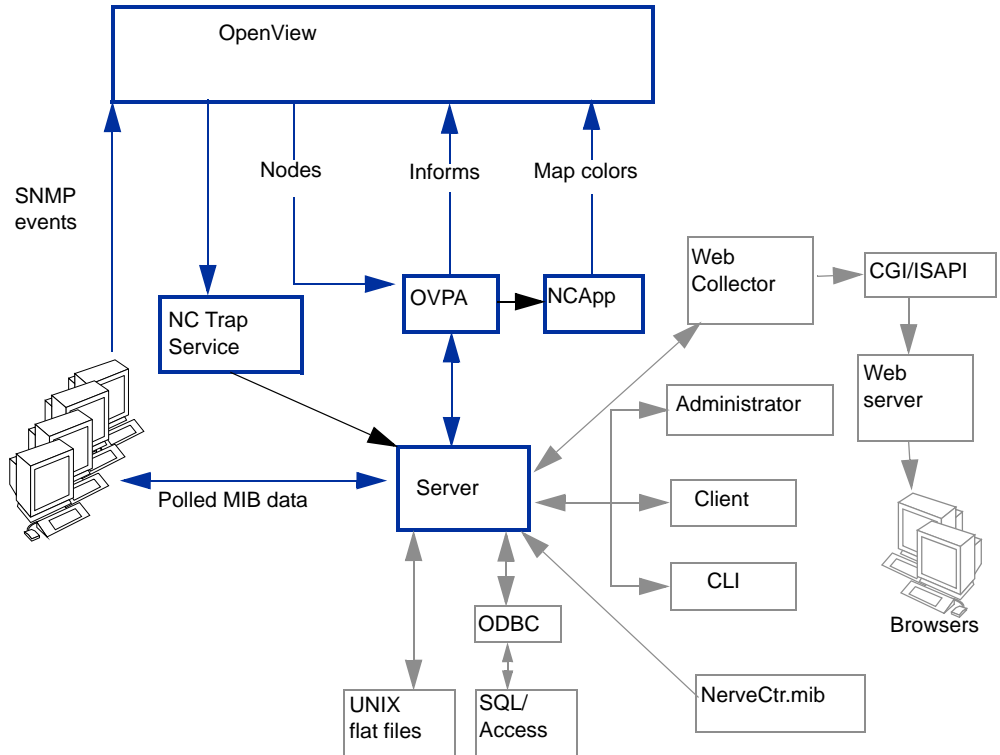


FIGURE 2-1. NerveCenter Co-resident Platform Integration

In a co-resident configuration, SNMP traps are sent to the platform. NerveCenter Trap service detects this information from the platform and forwards the traps to NerveCenter.

UNIX Installation

On UNIX, NerveCenter Trap service is always used whether or not the integration is co-resident. When NerveCenter and the platform are on separate machines, NerveCenter Trap service receives traps directly from the network and forwards the information to NerveCenter. *Figure 2-2* shows NerveCenter and the platform installed on separate machines. In the figure, NerveCenter's OVPA and NCAApp components are located on the platform host machine.

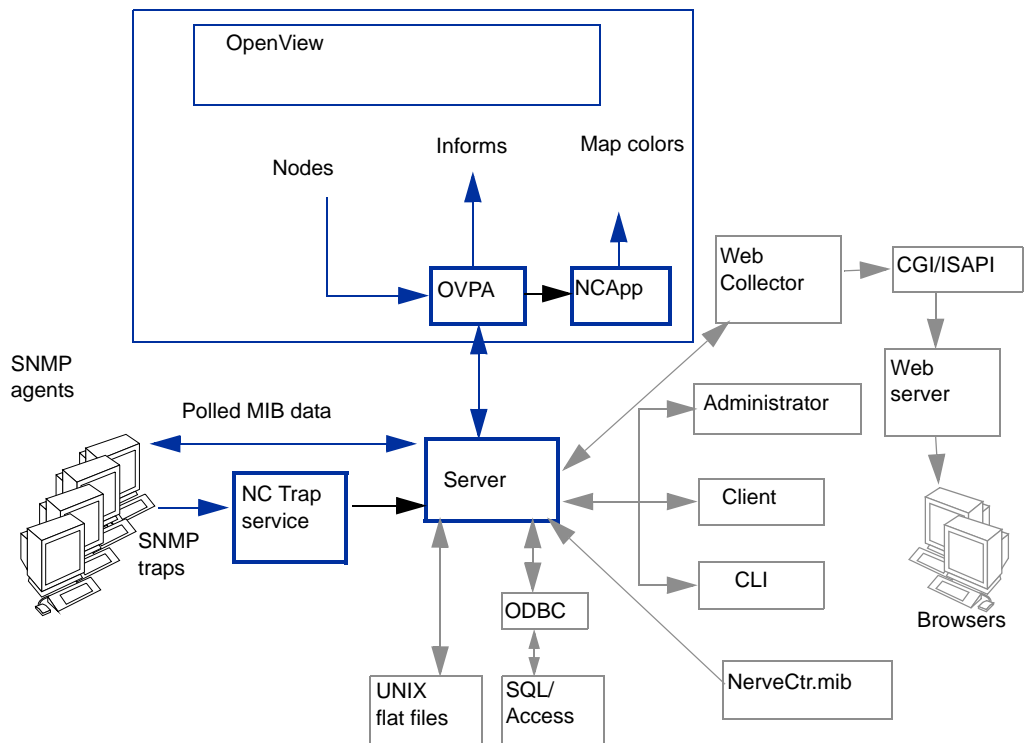


FIGURE 2-2. OVPA Integration on UNIX, Not Co-Resident

Windows installation on separate machines

Figure 2-3 shows OVPA integration with NerveCenter on Windows. NerveCenter and the platform are installed on separate machines. In the diagram, NerveCenter's OVPA and NCAApp components are located on the platform host machine.

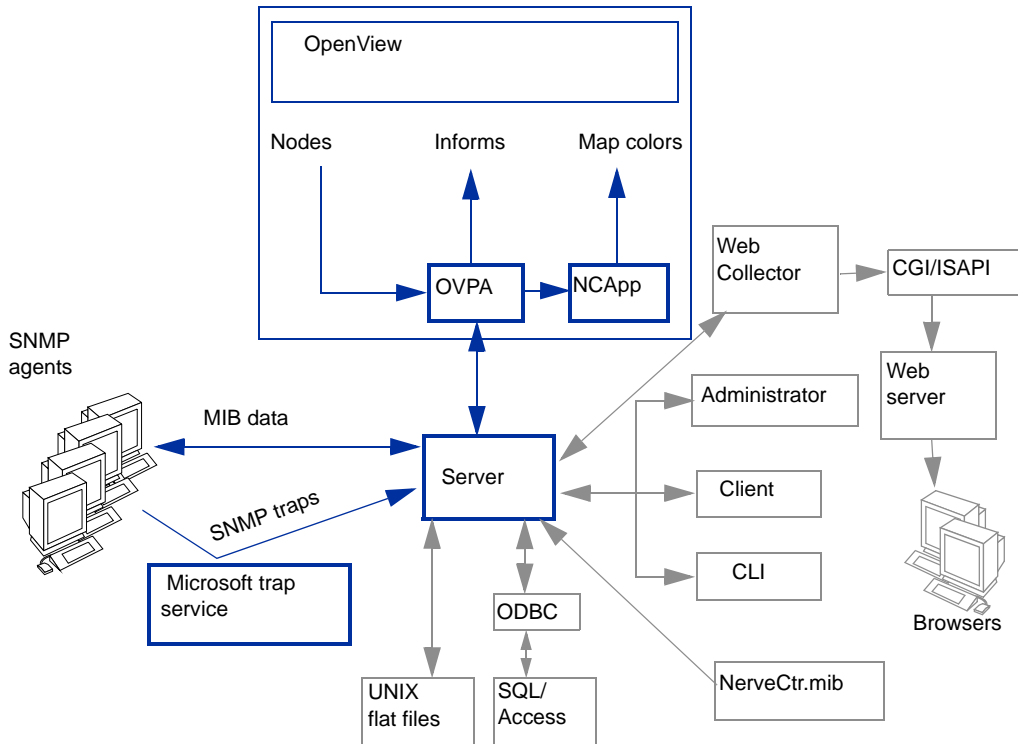


FIGURE 2-3. NerveCenter OVPA Integration on Windows, not Co-resident

Note that NerveCenter Trap service is not required in the Windows configuration shown above. Traps are detected by Microsoft's trap service and forwarded to NerveCenter.

How NerveCenter Integration Helps the Platform

For the platform, integration with NerveCenter means increased efficiency. NerveCenter can be configured to take over all event processing and minimize the number of events that appear in the platform's event console. NerveCenter does this by:

- ◆ Filtering out unimportant events.
- ◆ Correlating related events and notifying the platform only of the underlying problem.
- ◆ Handling an array of events through automated actions so that no notification is necessary.

Figure 2-4 shows an OpenView event browser that contains a number of events all caused by the same problem.

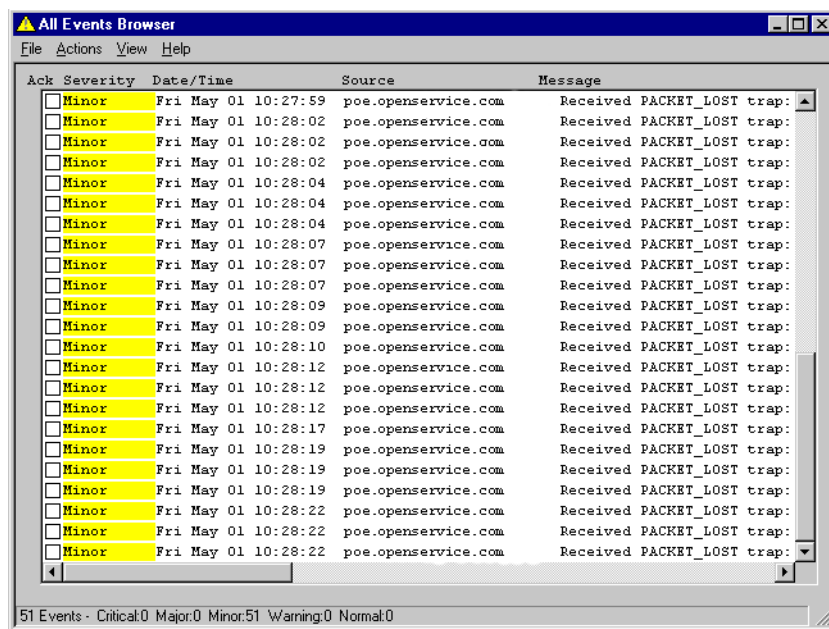


FIGURE 2-4. Redundant and Unfiltered Events

Figure 2-5 shows what might appear in the browser if NerveCenter were used to screen and correlate the conditions and pass on only important information to the platform event browser.

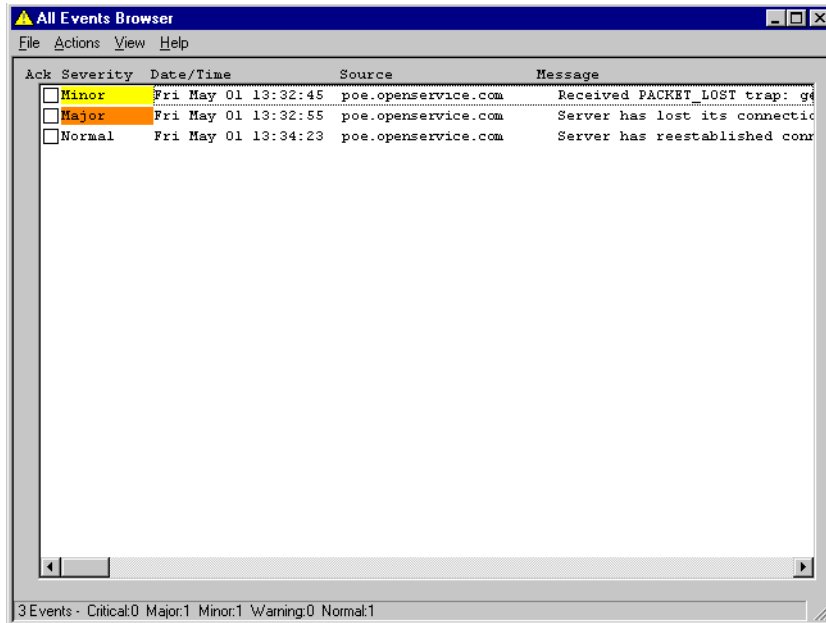


FIGURE 2-5. Filtered, Important Events

Universal Platform Adapter Integration

NerveCenter's universal platform adapter, paserver, enables NerveCenter to send informs to the following platforms:

- ◆ IBM Tivoli Netcool/OMNIBus on UNIX and Windows

The adapter is typically installed on the network management platform host machine, though there are other possible configurations. Other NerveCenter components, including the Server, can be installed on the same machine or on a different machine.

The following sections describe a typical setup for each platform.

IBM Tivoli Netcool/OMNIBus

NerveCenter can forward important events to IBM Tivoli Netcool/OMNIBUS on Windows and on UNIX. *Figure 2-6* shows Netcool/OMNIBUS integration with the other NerveCenter components.

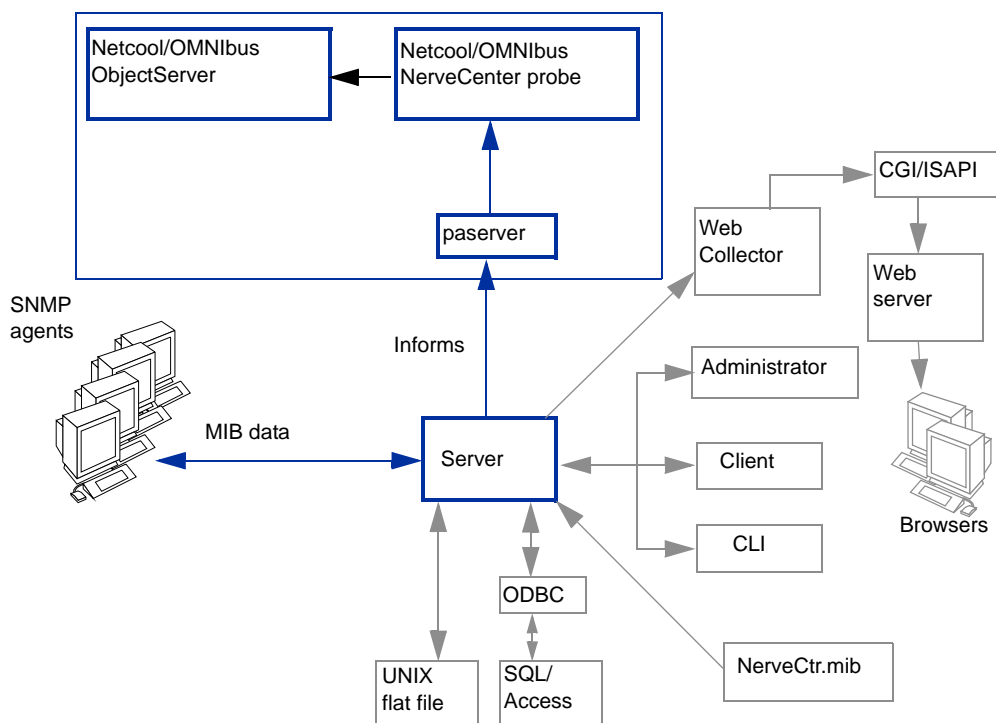


FIGURE 2-6. IBM Tivoli Netcool/OMNIBus Integration

To integrate the universal platform adapter with IBM Tivoli Netcool/OMNIBus, you must obtain a NerveCenter probe from Netcool/OMNIBus. NerveCenter inform messages travel from NerveCenter to the platform adapter, to the probe, and finally to the Netcool/OMNIBus ObjectServer. You can customize inform messages by editing the NerveCenter.rules file in `$OMNIHOME/probes/platform`.

IBM Tivoli recommends that the probe be installed on the same machine as paserver, and this configuration is the default setting for paserver. See your IBM Tivoli documentation for details.

The NerveCenter Server forwards noteworthy events to the platform by sending inform messages to paserver, which relays the informs directly to IBM Tivoli Netcool/OMNIBus. NerveCenter includes information about the associated node, the MIB values that were evaluated, the alarm that generated the inform action, and the severity of the states to and from, which the alarm transitioned when the inform was sent.

The NerveCenter Server and paserver are not required to be installed on the same system as IBM Tivoli Netcool/OMNIBus and the probe. NerveCenter Server can be installed on any supported platform and still relay messages through paserver to Netcool/OMNIBus on Windows, or Solaris.

Integrating with HP OpenView Network Node Manager

NerveCenter provides the event correlation engine that helps reduce the barrage of events typically displayed in HP OpenView Network Node Manager. OpenView can provide NerveCenter with node data and NerveCenter can send inform messages to OpenView when it detects significant events.

This chapter describes how to integrate NerveCenter with OpenView and includes the following sections:

Section	Description
<i>The OpenView Platform Adapter on page 22</i>	Explains how to enable and disable NerveCenter's OpenView Platform Adapter.
<i>Using OpenView as a Node Source on page 26</i>	Explains how to populate NerveCenter's node list with nodes from OpenView's node database.
<i>Sending NerveCenter Informs to OpenView on page 34</i>	Explains how to configure inform recipients for behavior models that send informs.
<i>The Reliability of NerveCenter Informs Sent to OpenView on page 37</i>	Describes the advantage of sending NerveCenter inform over SNMP traps. It also describes how to use NerveCenter's Inform Acknowledgement feature.
<i>Configuring OpenView to Integrate with NerveCenter on page 43</i>	Explains how to configure OpenView for better integration with NerveCenter.
<i>Reconfiguring OpenView before Removing the NerveCenter OpenView Platform Adapter on page 47</i>	Explains how to reconfigure OpenView before removing the NerveCenter OpenView Platform Adapter.
<i>OpenView Integration Reference on page 49</i>	Provides information you may need to reference occasionally when integrating NerveCenter and HP OpenView Network Node Manager.

The OpenView Platform Adapter

The NerveCenter component that allows NerveCenter to integrate with HP OpenView Network Node Manager is known as the OpenView Platform Adapter.

For integration to take place between NerveCenter and OpenView, the following must occur:

- ◆ The NerveCenter OpenView Platform Adapter must be installed on the machine hosting OpenView.

For detailed instructions on installing the NerveCenter OpenView Platform Adapter, see *Installing NerveCenter*.

- ◆ OpenView must be running.
- ◆ The NerveCenter OpenView Platform Adapter must be enabled.

Enabling the NerveCenter OpenView Platform Adapter registers it to start as a service whenever you start OpenView. See *Enabling and Disabling the Platform Adapter on page 22*.

- ◆ The NerveCenter OpenView Platform Adapter must be running.

The NerveCenter OpenView Platform Adapter typically starts at the same time OpenView starts. See *Starting and Stopping the OpenView Platform Adapter on page 25*.

Enabling and Disabling the Platform Adapter

A typical NerveCenter installation automatically enables the NerveCenter OpenView Platform Adapter. However, there may be times you will need to disable the NerveCenter OpenView Platform Adapter. Once you disable it, you will need to enable it manually before NerveCenter will be able to integrate with OpenView again. You can enable or disable the NerveCenter OpenView Platform Adapter by following the procedures described in this section.

Enabling the NerveCenter OpenView Platform Adapter

If the NerveCenter OpenView Platform Adapter is disabled or was never enabled during NerveCenter installation, you can enable it at any time by adding the OpenView Platform Adapter local registration file (ovpa.lrf) to the list of services that are started by ovstart.

TIP

If you are unsure if the NerveCenter OpenView Platform Adapter is registered to run with ovstart, you can check OpenView's start-up file. When the NerveCenter OpenView Platform Adapter is registered, its data, including the pathname to the

executable, appears in the file `ovsuf` typically found in the `conf` directory under OpenView's installation directory. If either the `ovpa` data does not appear or the line with the `ovpa` data begins with a 1, then the NerveCenter OpenView Platform Adapter will not run with `ovstart`.

TO ENABLE THE NERVECENTER OPENVIEW PLATFORM ADAPTER

1. From the command line of the machine hosting HP OpenView Network Node Manager, run **ovstop**.

The OpenView services stop running.

**TIP**

On Windows, you can also select the **Services > Stop** utility from the Windows **Start** menu.

2. Navigate to the following directory:

Windows: `installpath\lrf` where *installpath* is the installation directory for OpenView.

UNIX: `installpath/OV/lrf` where *installpath* is the installation directory for NerveCenter.

3. Run the following command:

```
ovaddobj ovpa.lrf
```

4. Run **ovstart**.

OpenView's services, including NerveCenter OpenView Platform Adapter, start running.

**TIP**

On Windows, you can also select the **Services > Start** utility from the Windows **Start** menu.

The NerveCenter OpenView Platform Adapter now automatically runs every time you start OpenView.

Disabling the NerveCenter OpenView Platform Adapter

If the NerveCenter OpenView Platform Adapter is enabled, you can disable it at any time by removing the OpenView Platform Adapter local registration file (`ovpa.lrf`) from the list of services that are started by `ovstart`.

TO DISABLE THE NERVECENTER OPENVIEW PLATFORM ADAPTER

1. From the command line of the machine hosting HP OpenView Network Node Manager, run **ovstop**.

OpenView's services stop running.

**TIP**

On Windows, you can also select the **Services > Stop** utility from the Windows **Start** menu.

2. Navigate to the following directory:
 - ◆ Windows: *installpath*\lrf where *installpath* is the installation directory for OpenView.
 - ◆ UNIX: *installpath*/OV/lrf where *installpath* is the installation directory for NerveCenter.
3. Run the following command:

```
ovdelobj ovpa.lrf
```

4. Run **ovstart**.

OpenView's services, including NerveCenter OpenView Platform Adapter, start running.

**TIP**

On Windows, you can also select the **Services > Start** utility from the Windows **Start** menu.

The NerveCenter OpenView Platform Adapter will no longer automatically run every time you start OpenView.

Starting and Stopping the OpenView Platform Adapter

The NerveCenter component that allows you to integrate NerveCenter with HP OpenView Network Node Manager is known as the OpenView Platform Adapter. Following a typical installation of the NerveCenter OpenView Platform Adapter component, NerveCenter enabled the OpenView Platform Adapter to run automatically as a service of OpenView.



NOTE

If for some reason, the NerveCenter OpenView Platform Adapter was not enabled during installation or has since been disabled, see [Enabling and Disabling the Platform Adapter on page 22](#).

Therefore, whenever you start OpenView services by typing **ovstart**, the NerveCenter OpenView Platform Adapter will automatically start with any other OpenView services.

In the same way, typing **ovstop** stops all OpenView services, including the NerveCenter OpenView Platform Adapter.

- ◆ If you want to start only the NerveCenter OpenView Platform Adapter, you can type at the command line:

```
ovstart ovpa
```

- ◆ If you want to stop only the NerveCenter OpenView Platform Adapter, you can type at the command line:

```
ovstop ovpa
```

Using OpenView as a Node Source

NerveCenter can integrate with HP OpenView Network Node Manager by receiving information about some or all of the nodes managed by OpenView.

NerveCenter can obtain node information from any or all of the following sources:

- ◆ A network management platform, such as HP OpenView Network Node Manager
- ◆ The NerveCenter Discovery behavior model
- ◆ An administrator's manual entries



NOTE

Though NerveCenter supports SNMP v1, v2c, and v3, when NerveCenter obtains its nodes from a platform, the platform does not provide SNMP version information. By default, NerveCenter deems the SNMP agents on these nodes to be SNMP v1 by default.

If you want NerveCenter to attempt SNMP version classification automatically for the nodes it receives from your platform, you must enable auto-classification. Then, NerveCenter can classify the correct SNMP version for each node with each resynchronization. Refer to *Managing SNMP Settings in Managing NerveCenter* for more information about SNMP auto-classification.



NOTE

If you want NerveCenter to manage SNMP v3 nodes, you must use NerveCenter as your trap source regardless of the node source you configure. Refer to *Managing the NerveCenter Trap Source in Managing NerveCenter* for more information about the SNMP trap source.

When NerveCenter uses information obtained by a network management platform, it does not use the platform's database as its repository for managed nodes. Instead, it stores node information in its own database in a node list.

There are several reasons for NerveCenter maintaining a node list in its own database:

- ◆ There may be a considerable distance between the platform's database and NerveCenter, making frequent access time-consuming and costly.
- ◆ NerveCenter adds configuration data to the node data that the management platform does not necessarily provide.
- ◆ Administrators have the option of adding nodes not in the platform's node database to the node list in NerveCenter's database.

Using OpenView as a node data source involves the following three items:

- ◆ Setting filters that limit the nodes NerveCenter monitors.
For further details, see *Filtering Nodes from HP OpenView Network Node Manager on page 30*
- ◆ Populating NerveCenter's node list
For further details, see *Populating the Node List with OpenView as a Data Source on page 27*.
- ◆ Maintaining current data between NerveCenter and OpenView
For further details, see *Synchronizing with HP OpenView Network Node Manager on page 29*.

Populating the Node List with OpenView as a Data Source

NerveCenter is able to receive information about some or all of the nodes managed by OpenView.



NOTE

To use OpenView to populate NerveCenter's node database, you must have the NerveCenter OpenView Platform Adapter (OVPA) installed and running.

To populate NerveCenter's node list using OpenView, you must specify it as a source for the node data. Each NerveCenter database populates its node list from just one network management platform database. Depending on your filtering, the database may contain all the nodes or just a subset. In either case, there is just one source of the information.



CAUTION

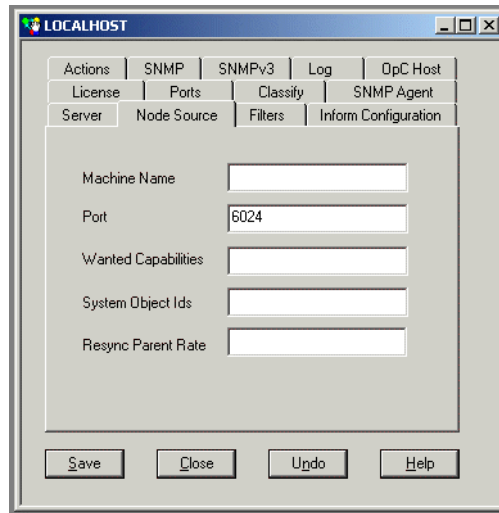
If you wish to map system Object Identifiers (OID) to NerveCenter property groups, you must make the necessary configurations in the NerveCenter Client before naming the node data source. (See *Using OID to Property Group Mappings in Designing and Managing Behavior Models*) After NerveCenter initially populates its node list, any subsequent mapping of OIDs to property groups will affect only new nodes added to the node list.

TO USE OPENVIEW AS A NODE DATA SOURCE

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
For further instructions, see *Connecting to a NerveCenter Server in Managing NerveCenter*.

2. Select the **Node Source** tab.

NerveCenter displays the Node Source tab.



The screenshot shows a dialog box titled "LOCALHOST" with a menu bar containing: Actions, SNMP, SNMPv3, Log, OpC Host, License, Ports, Classify, SNMP Agent, Server, Node Source, Filters, and Inform Configuration. The "Node Source" tab is selected. The dialog contains five text input fields: "Machine Name" (empty), "Port" (containing "6024"), "Wanted Capabilities" (empty), "System Object Ids" (empty), and "Resync Parent Rate" (empty). At the bottom are four buttons: Save, Close, Undo, and Help.

3. In the **Machine Name** field, type either the name or the IP address of a host that runs OpenView.

For example, if you have OpenView running on a machine named Norm, you would type the name **Norm** or its IP address in the Machine Name field.

If the Machine Name field is left blank, NerveCenter does not retrieve nodes from any platform.

4. In the **Port** field, type the number of the port used to communicate with the platform adapter process on the host. The default is 6024.

The platform adapter must be configured to listen on the same port specified in this field.

5. Select **Save**.

NerveCenter now retrieves its initial node data from OpenView's database.

Synchronizing with HP OpenView Network Node Manager

Over time, a network's topology will change. Eventually OpenView will add newly discovered devices to its database. It will also delete nodes and change node information. If NerveCenter depends on OpenView for the data in its node list, it needs to adapt to reflect these changes.

NerveCenter automatically updates its node list to keep in sync with OpenView's node data. This occurs in the following situations:

- ◆ When OpenView adds a node to its node database. After NerveCenter verifies the node meets the criteria set by its filters, it will add the node to its node list.
- ◆ When OpenView deletes a node from its node database. NerveCenter will delete from its node list any node that is set to Autodelete. Autodelete is the default setting for any new node added to the node list. This setting can be changed in the node's Node Definition Window in the NerveCenter Client. (See *Discovering and Defining Nodes in Designing and Managing Behavior Models*.)
- ◆ When OpenView changes information about a node in its node database. NerveCenter will make any necessary changes to its node data, including changes in the community string, address, parenting information or the managed/unmanaged state.



NOTE

If OpenView unmanages a node in the NerveCenter node list, the unmanaged state will be updated in NerveCenter. However, if OpenView unmanages a node not found in NerveCenter's node list, the node will not be added to NerveCenter.

Most often, the node list will only be updated a node at a time. Occasionally, NerveCenter will need to perform a complete resynchronization with the platform. A resynchronization gathers from the platform the most current node data for all nodes. This occurs in the following situations:

- ◆ The NerveCenter Server is started and successfully connects to the OpenView Platform Adapter (OVPA).
- ◆ A connection between the NerveCenter Server and the node source successfully reconnects after being broken.
- ◆ The NerveCenter administrator changes the way in which NerveCenter filters by capabilities or system Object Identifiers (OIDs).
- ◆ A user manually chooses Resync in the Server menu of the NerveCenter Client.

The **Machine Name** field on the **Node Source** tab of the NerveCenter Administrator specifies the name of the host running the platform resynchronizing with NerveCenter. (See *Populating the Node List with OpenView as a Data Source on page 27* for more details on how to declare a node

data source.) The **Node Source** and **Filters** tabs also specify the parameters NerveCenter uses to filter node data. (See *Filtering Nodes from HP OpenView Network Node Manager on page 30.*)

Anyone administering NerveCenter should be aware of two important scenarios involving changes to OpenView's database:

- ◆ If the name changes in OpenView's database, NerveCenter considers it to be a new node.
- ◆ If a node is unmanaged in one of OpenView's maps but is managed in another, the node will remain in the managed state in NerveCenter's node data.

 **CAUTION**

Since OpenView's node is matched to a NerveCenter node using its name, you should use care when changing NerveCenter's node configurations.

Resynchronization adds nodes when it cannot find names that match OpenView's map information. Therefore, if you change a node's name in the Node Definition window, resynchronization will not find a match and will add a node, resulting in two nodes with the same address but different names.

Filtering Nodes from HP OpenView Network Node Manager

When using OpenView as a source of information about nodes, it is important to determine which of the nodes in OpenView's database NerveCenter will manage. NerveCenter does not need to monitor every node on your network.

There are several methods for restricting which nodes in OpenView's node database will be placed in NerveCenter's node list:

- ◆ *Filtering by Node Capabilities on page 30*
- ◆ *Filtering by Node System Object Identifiers on page 31*
- ◆ *Filtering by Node IP Addresses on page 31*

Filtering by Node Capabilities

NerveCenter allows you to monitor managed nodes that have particular capabilities. Typically OpenView assigns these capabilities to a node to determine applicable management activities. Some examples of these capabilities are isRouter, isHub, and isIP.

 **NOTE**

Filtering by capabilities is available only when OpenView has assigned specific capabilities to a node.

See *Filtering Using Node Capabilities in Managing NerveCenter* for details.

When NerveCenter adds the new capabilities filter, it also closes and opens a new connection with the platform adapter. NerveCenter automatically performs a resynchronization with OpenView's database.

New nodes will be added. Any node that is marked Autodelete (the default) will be deleted.

Filtering by Node System Object Identifiers

NerveCenter allows you to monitor managed nodes according to their particular system object identifiers (OIDs).

A node's System Object ID is an SNMP MIB-II object in the system group. It identifies the SNMP agent software running on the device. It is, however, commonly used to identify the type and vendor of the device because a particular vendor's agent usually runs on that vendor's devices.

See *Filtering Using a Node System Object Identifier in Managing NerveCenter* for details.

When NerveCenter adds the new OID filter, it also closes and opens a new connection with the platform adapter. NerveCenter automatically performs a resynchronization with OpenView's database.

New nodes will be added. Any node that is marked Autodelete (the default) will be deleted.

Filtering by Node IP Addresses

In addition to filtering nodes by OIDs and capabilities, NerveCenter allows you to filter out all nodes that do not belong to one or more subnets. NerveCenter determines the subnet by combining a specific IP address with a subnet mask. NerveCenter can filter by subnets of both Class B and Class C networks.

See *Filtering Nodes by IP Address in Managing NerveCenter* for details.

Identifying Parent-Child Relationships

In order to use NerveCenter's Downstream Alarm Suppression behavior model, it is necessary to establish the parent-child relationship between nodes. You can let OVPA extract relationship information from HP OpenView Network Node Manager and either store it in the NerveCenter database or in a text file. You can also create the text file manually. For more information about the Downstream Alarm Suppression behavior model, see *Downstream Alarm Suppression in Designing and Managing Behavior Models*.

**NOTE**

By default, OVPA does not get information about a node's parents from OpenView. You must configure OVPA to collect that information by doing the following steps.

TO IDENTIFY PARENT-CHILD RELATIONSHIPS USING OVPA

1. Make sure HP OpenView Network Node Manager is running. Also make sure the NerveCenter Server is running.
2. Make sure that OpenView is set up as your node source in the NerveCenter Administrator.

See *Populating the Node List with OpenView as a Data Source on page 27* for more details.

3. If OVPA is running, stop it by typing **ovstop ovpa** at the command line.
4. Start OVPA in parenting mode from the command line by typing one of the following commands:

- ◆ `ovpa -pc`

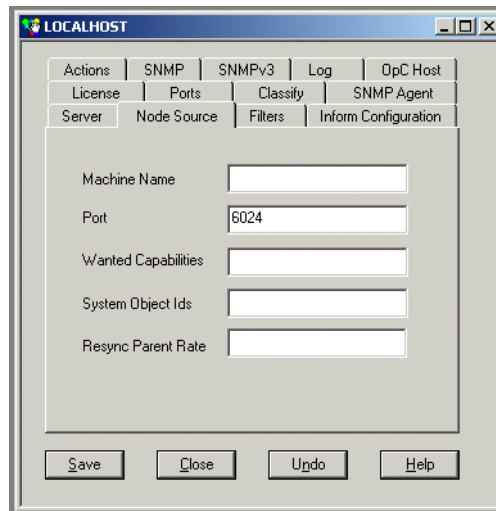
OVPA runs and computes parenting information, resynchronizing the information periodically. The how often OVPA resynchronizes information is configurable through the Node Source tab in the NerveCenter Administrator. The default resync parent rate is 600 seconds.

- ◆ `ovpa -pc -writeParentsToFile hostname`

hostname is the name of the machine on which the NerveCenter Server runs. OVPA computes the parenting information, writes it to a file named **hostname_PC.dat**, and then stops.

TO CHANGE THE RESYNC PARENT RATE

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
For further instructions, see *Connecting to a NerveCenter Server in Managing NerveCenter*.
2. Select the **Node Source** tab.
NerveCenter displays the Node Source tab.



3. In the **Resync Parent Rate** field, type the number of seconds you want between each resync attempt.
If left blank, the default resync parent rate is 600.
4. Select **Save**.

TO IDENTIFY PARENT-CHILD RELATIONSHIPS MANUALLY

1. Open a new text file.
2. Include a line for each node that has parents. Use the following syntax:

```
child parent
```

where *child* is the name of the node and *parent* is the name of each node on which the child is dependent. If you have more than one parent, separate parents by typing a space between each one.



NOTE

If NerveCenter uses a full domain name for the node, use the full name in this file to refer to that node.

For example, if nodeA is dependent on nodeB.domain.com and nodeC; and nodeB.domain.com is dependent on nodeD, then the text file would look like this:

```
nodeA nodeB.domain.com nodeC
nodeB.domain.com nodeD
```

3. Save and close the file.
-

Sending NerveCenter Informs to OpenView

One of NerveCenter's powerful platform integration features is its ability to send inform packets to HP OpenView Network Node Manager. Since the inform packets use the Transmission Control Protocol (TCP), the alert sent to OpenView is more reliable than a standard SNMP trap.

To configure NerveCenter to keep track of acknowledges sent in response to an inform, see [The Reliability of NerveCenter Informs Sent to OpenView on page 37](#).

As you create or modify a behavior model to notify OpenView, you determine the specific inform number it will receive. However, before you can use this behavior model, NerveCenter must know which machine or machines will receive the inform.

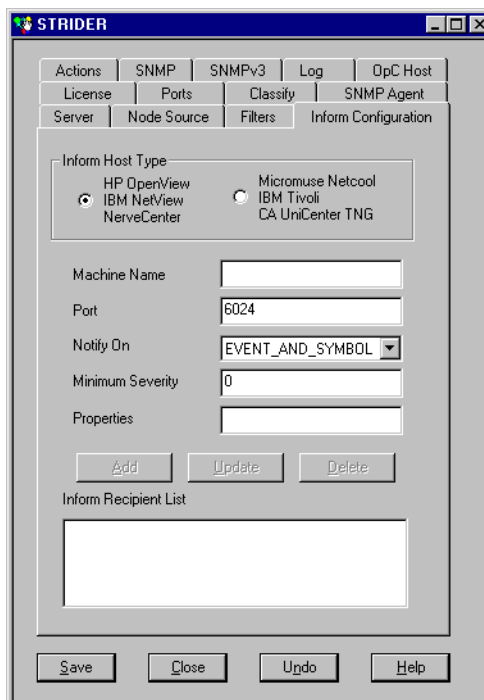
The following procedure will step you through the process of declaring one or more recipients of NerveCenter informs.

TO SPECIFY THE DESTINATION OF NERVECENTER INFORMS SENT TO OPENVIEW

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. If you need help opening NerveCenter Administrator or connecting to a NerveCenter Server, see [Connecting to a NerveCenter Server in Managing NerveCenter](#).

2. Select the **Inform Configuration** tab.

The Inform Configuration tab appears.



3. In the **Inform Host Type** field, select **HP OpenView NerveCenter**.
Selecting this option associates the NerveCenter OpenView Platform Adapter with the machine hosting OpenView.
4. In the **Machine Name** field, type the name of the machine hosting OpenView.
5. In the **Port** field, type the port number your NerveCenter Server will use when communicating with the NerveCenter OpenView Platform Adapter.
By default, NerveCenter uses the port number 6024.
6. In the **Notify On** field, select one of the following:
 - ♦ **EVENT_ONLY**—Events are sent to this platform host when an Inform action is invoked. No symbol color change messages are sent.
 - ♦ **SYMBOL_ONLY**—Messages causing symbol color changes on the platform map are sent to this host. Events are not sent.
 - ♦ **EVENT_AND_SYMBOL**—Both event and symbol messages are sent.

**NOTE**

Selecting either **SYMBOL_ONLY** or **EVENT_AND_SYMBOL** causes the node's object to update according to the NerveCenter color in OpenView's map.

The recommended setting for OpenView is **EVENT_AND_SYMBOL**. Even though this setting increases traffic overhead, the other two options prevents data from reaching your platform.

This setting need to be consistent with your platform's map configuration settings. For example, if you configured your platform to allow symbol color changes from NerveCenter but now select **EVENT_ONLY**, you will see no symbol color changes.

7. In the **Minimum Severity** field, type the number representing the minimum severity an alarm must reach before triggering a message to this platform.

This option enables you to be selective about which events are sent to particular platforms. For example, a local platform could get all events, while a lead or central platform could get only critical events. When NerveCenter sends Informs to your platform, NerveCenter first checks the minimum severity value entered here to ensure that the trap value for the Inform matches or exceeds that severity.

**NOTE**

There is one case when NerveCenter disregards the minimum severity value specified in Administrator: After NerveCenter sends an Inform, if the condition returns to a normal state (a state below the minimum severity threshold) it's important that NerveCenter notify the platform of this change. Therefore, if a node transitions the alarm from a severity above the minimum value to a severity below the minimum value, and the transition includes an Inform action, NerveCenter will send a Normal Inform to the platform. This allows the platform to reset the mapped severity color associated with the node.

**NOTE**

The values associated with each severity in NerveCenter can be viewed and altered in the NerveCenter Client in the **Severity List** found under the **Admin** menu. See *NerveCenter Severities in Designing and Managing Behavior Models* for more details.

8. In the **Properties** field, type zero or more properties.

NerveCenter will only send an inform packet to this platform if the managed node's property group contains at least one of the properties listed in this field. If no events are listed, NerveCenter sends events for all managed nodes.

This option enables you to be selective about which events are sent to particular platforms. For example, one platform could receive informs prompted by only routers.

9. Select **Add**.

The platform's host machine is added to the Inform Recipient List.

10. Repeat steps 3 through 9 for each machine hosting OpenView that will receive a NerveCenter inform packet.

11. Select **Save**.

When a behavior model performs an Inform alarm action, each machine within the Inform Recipient List that is associated with the NerveCenter OpenView Platform Adapter will receive the inform as long as the alarm meets the relevant criteria.

The Reliability of NerveCenter Informs Sent to OpenView

Since NerveCenter informs are sent via the Transmission Control Protocol (TCP) they tend to be more reliable than SNMP traps. TCP allows for the following:

- ◆ NerveCenter informs take priority over SNMP traps.
- ◆ A direct connection is made between the NerveCenter Server and the NerveCenter OpenView Platform Adapter.
- ◆ The NerveCenter OpenView Platform Adapter can acknowledge the receipt of a NerveCenter inform.

The following diagram illustrates how the NerveCenter OpenView Platform Adapter acknowledges informs.

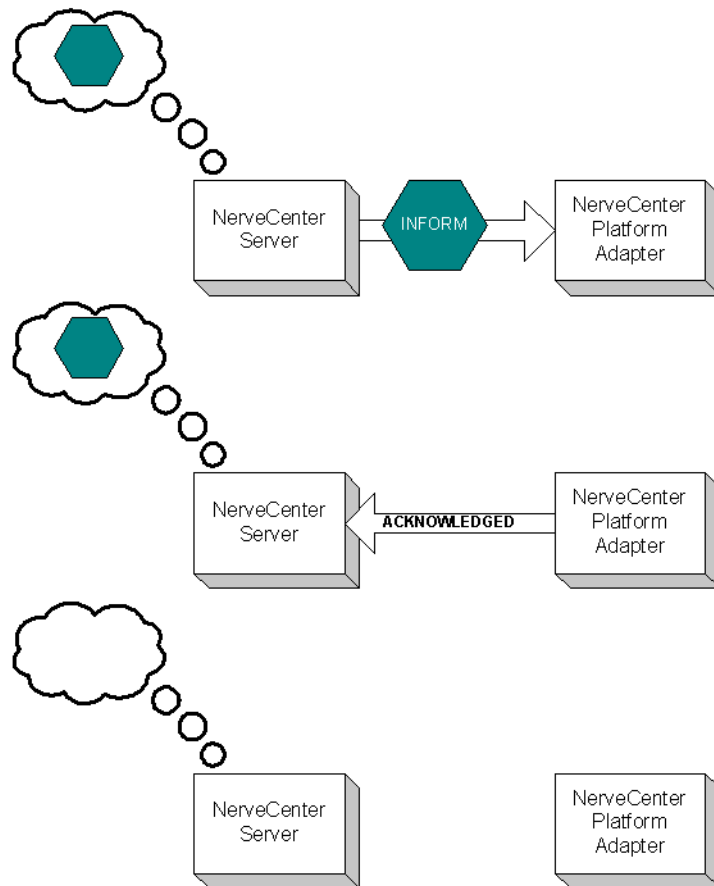


FIGURE 3-1. How NerveCenter handles inform acknowledgement

When a NerveCenter behavior model performs the Inform alarm action:

- ◆ The NerveCenter Server sends an Inform message to the NerveCenter OpenView Platform Adapter. The NerveCenter Server places the inform in a queue.
- ◆ The NerveCenter OpenView Platform Adapter sends the inform message to OpenView. At the same time, the NerveCenter OpenView Platform Adapter sends a packet back to the NerveCenter Server acknowledging it has received the inform packet.
- ◆ Once the NerveCenter Server receives the inform acknowledgement, it deletes the inform from its queue.

If for any reason the NerveCenter Server loses its connection to the NerveCenter OpenView Platform Adapter the informs will be held in its inform queue. Once the server regains its connection, any inform that was not acknowledged as received will be sent again.

By default, the NerveCenter Server is not set to have informs acknowledged. If you want NerveCenter to keep track of informs and their acknowledgement, you must enable the inform acknowledgement feature.

The section [Saving NerveCenter Informs Until Acknowledgement on page 39](#) explains how to enable the inform acknowledgement feature.

**NOTE**

To use the Inform acknowledgement feature, you must use version 3.6 or later of NerveCenter Server and the NerveCenter OpenView Platform Adapter.

While the connection between the NerveCenter Server and the OpenView Platform Adapter is down, any new informs will be placed in the inform queue. The queue length is limited. If the number of informs waiting to be sent exceeds the queue limit, NerveCenter will delete the oldest inform so the newest inform can be added to the queue. If NerveCenter drops informs it will also fire the predefined trigger NC_INFORMS_LOST. (See [Built-In Triggers in Designing and Managing Behavior Models](#) for complete details on using this and other predefined triggers in behavior models.) By default, the queue depth is set to 10000 informs. You can however set the queue to whatever depth you prefer.

The section [Configuring the Inform Queue Depth on page 41](#) explains how to set the inform queue depth.

Saving NerveCenter Informs Until Acknowledgement

By default, NerveCenter Server does not save informs to wait acknowledgement. If you want NerveCenter to keep track of informs until the NerveCenter OpenView Platform Adapter acknowledges their receipt, you must enable the inform acknowledgement feature. This section explains how to enable the inform acknowledgement feature.

**NOTE**

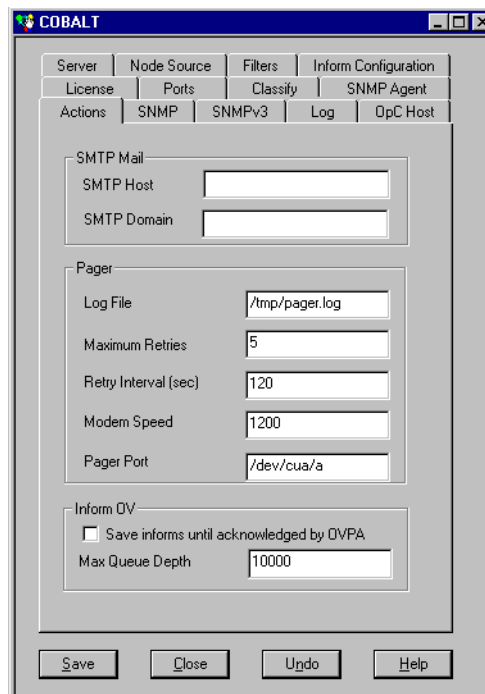
To use the Inform acknowledgement feature, you must use version 3.6 or later of NerveCenter Server and the NerveCenter OpenView Platform Adapter.

TO ENABLE INFORM ACKNOWLEDGEMENT

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. If you need help opening NerveCenter Administrator or connecting to a NerveCenter Server, see *Connecting to a NerveCenter Server in Managing NerveCenter*.

2. Select the **Actions** tab.

NerveCenter displays the Actions tab.



In the Inform OV area is a checkbox labeled **Save informs until acknowledged by OVPA**. By default, this box is not selected.

3. Check the box labeled **Save informs until acknowledged by OVPA**.
4. Select **Save**.

The NerveCenter Server will now save Informs in its queue until the NerveCenter OpenView Platform Adapter acknowledges their receipt.

Configuring the Inform Queue Depth

When the NerveCenter inform acknowledgement feature is enabled, any new informs will be placed in the Inform Queue should the NerveCenter Server lose its connection to the NerveCenter OpenView Platform Adapter. Once the connection is restored, NerveCenter will resend any unacknowledged informs.

The section *Saving NerveCenter Informs Until Acknowledgement on page 39* explains how to enable the inform acknowledgement feature.

The inform queue length is limited. If the number of informs waiting to be sent exceeds the queue limit, NerveCenter will delete the oldest inform so the newest inform can be added to the queue. If NerveCenter drops informs it will also fire the predefined trigger NC_INFORMS_LOST. (See *Built-In Triggers in Designing and Managing Behavior Models* for complete details on using this and other predefined triggers in behavior models.)

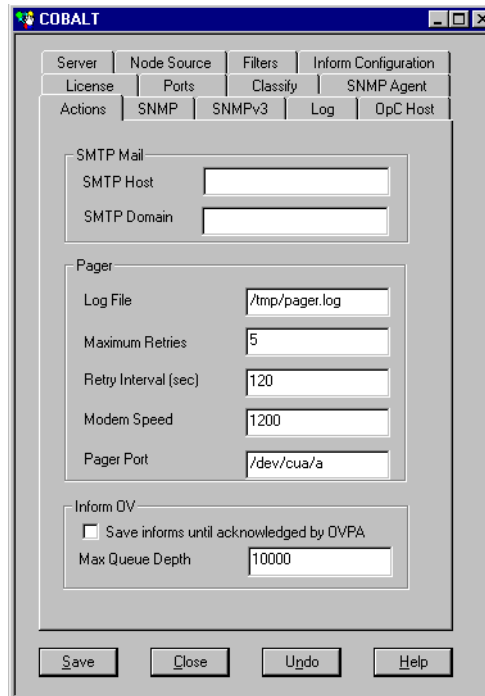
This section will step you through the process of setting the maximum Inform Queue depth.

TO CONFIGURE THE MAXIMUM DEPTH FOR THE INFORM QUEUE

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. If you need help opening NerveCenter Administrator or connecting to a NerveCenter Server, see *Connecting to a NerveCenter Server in Managing NerveCenter*.

2. Select the **Actions** tab.

NerveCenter displays the Actions tab.



3. In the **Max Queue Depth** box enter the desired number of informs you want to be saved in the Inform Queue.
By default, the Inform Queue depth is set to a maximum of 10000 informs.
4. Select **Save**.

When the number of informs in the Inform Queue exceed the number entered in Max Queue Depth, NerveCenter will begin dropping informs to include more recent informs.

NerveCenter's trace counters list how many OVPA Informs are queued at any given time. See *Using Trace Counters to Troubleshoot NerveCenter in Managing NerveCenter* for more information about trace counters.

NerveCenter ships with a behavior model, *InformConnectStatus.mod*, which detects the status of Informs and their host connections. See *InformConnectStatus in Behavior Models Cookbook* for details.

Configuring OpenView to Integrate with NerveCenter

Once you have configured NerveCenter to integrate with HP OpenView Network Node Manager, you may want to configure OpenView for an even tighter integration. Three options for modifying OpenView are:

- ◆ Customize notification messages to make them more meaningful.
Your NerveCenter behavior models will send OpenView informs with specific numbers. You may want to configure OpenView to interpret each of these specific numbers as a specific event on your network. See OpenView's documentation for complete instructions on configuring notification messages.
- ◆ Customize device symbol colors in a node map so they will reflect the severity of active NerveCenter alarms.
- ◆ As NerveCenter behavior models are created to send informs to OpenView, you may want to customize OpenView notification messages to make them more meaningful.

The next section will discuss this option in more detail.

Configuring a Node Map with NerveCenter Alarm Severity Colors

NerveCenter can affect symbol colors on OpenView's map in one of several ways:

- ◆ NerveCenter can change the color of the device symbol itself to reflect the highest severity of all active alarms for the device.
- ◆ NerveCenter can change the color of a NerveCenter symbol automatically created under the symbol for each managed device.
- ◆ NerveCenter can leave unaffected symbol colors on the map. This setting is the default.

Keep in mind that although NerveCenter is shipped with 12 different colors for severities, your network management platform may have fewer choices. If you choose either of the first two options listed above, the colors on your network management platform symbols will be restricted to the network management platform's choices.

You must configure an IP map to enable map symbol color changes—either of the first two choices. You can either create a new map or configure an existing map. Follow one of the following procedures:

- ◆ [Mapping NerveCenter Alarm Severity Colors to OpenView on page 44](#)
- ◆ [Modifying NerveCenter-to-OpenView Alarm Severity Color Mapping on page 45](#)

Mapping NerveCenter Alarm Severity Colors to OpenView

TO CONFIGURE OPENVIEW TO REFLECT NERVECENTER ALARM SEVERITY COLORS WITH A NEW MAP

1. Begin the new map wizard.

This typically requires choosing **New** or **New Map** from the Map menu. See your OpenView's documentation for further instructions.

2. Enter a name for the new map. Select **Next**.

3. Choose one of the following:

- ◆ If you do not want NerveCenter to change the colors of any symbol colors at all, or if you want NerveCenter to change only the colors of the NerveCenter symbols on your map, skip to step 7.
- ◆ If you want NerveCenter to change a device's symbol color directly to reflect the highest active alarm severity for the device, select **IP Map** in the Configurable Applications list and select **Configure For This Map**.

4. Change **Should status of nodes be IP/IPX only?** to **False**.

5. Select **Verify**.

OpenView verifies your requested changes and then displays a message at the bottom of the window.

6. Select **OK**.

The Configuration window closes, and returns you to the New Map window.

7. From the **Configurable Applications** list, select LogMatrix NerveCenter and **Configure For This Map**.

8. For the setting **Enable NerveCenter for this map**, choose one of the following:

- ◆ If you do not want NerveCenter to change the colors of any symbols, select **False** and skip to step 10.
- ◆ If you want NerveCenter to change device or NerveCenter symbol colors to reflect the highest active alarm severity, select **True**.

9. For the **Set NerveCenter status on node symbol** setting, choose one of the following:

- ◆ If you want NerveCenter to change the color of the NerveCenter icon in the submap beneath the device symbol, select **False**.

If you select **False**, NerveCenter automatically creates a new NerveCenter icon in the submap beneath the symbol for each managed device on the map. Color changes are applied to the NerveCenter symbol, and the color of the device symbol itself is changed according to the normal OpenView settings for the map.

- ◆ If you want NerveCenter to change a device's symbol color to reflect the highest active alarm severity, select **True**.

10. Select **Verify.**

OpenView verifies your requested changes and then displays a message at the bottom of the window.

11. Select **OK.**

The Configuration window closes, and returns you to the New Map window.

12. Select **Next.**

13. Select an appropriate value for Compound Status. See your OpenView's documentation for further instructions.

14. Select **Next.**

The final wizard screen is displayed.

15. Select **Finish.**

OpenView configures the new map to handle symbol data sent by a NerveCenter inform.

Modifying NerveCenter-to-OpenView Alarm Severity Color Mapping

TO CONFIGURE OPENVIEW TO REFLECT NERVECENTER ALARM SEVERITY COLORS BY MODIFYING AN EXISTING MAP

1. Open the appropriate map, if it is not currently open.

This typically requires choosing **Open** or **Open/List** from the Map menu. See your OpenView's documentation for further instructions.

2. Open the Map Properties window.

This typically requires choosing **Properties** from the Map menu. See your OpenView's documentation for further instructions.

3. Select the **Applications tab.**

4. Choose one of the following:
 - ♦ If you do not want NerveCenter to change the colors of any symbol colors at all, or if you want NerveCenter to change only the colors of the NerveCenter symbols on your map, skip to step 8.
 - ♦ If you want NerveCenter to change a device's symbol color directly to reflect the highest active alarm severity for the device, select **IP Map** in the Configurable Applications list and select **Configure For This Map**.
5. Change **Should status of nodes be IP/IPX only?** to **False**.
6. Select **Verify**.

OpenView verifies your requested changes and then displays a message at the bottom of the window.
7. Select **OK**.

The Configuration window closes, and you return to the New Map window.
8. From the Configurable Applications list select **LogMatrix** NerveCenter and select **Configure For This Map**.
9. For the **Enable NerveCenter for this map** setting, choose one of the following:
 - ♦ If you do not want NerveCenter to change the colors of any symbols, select **False** and skip to step 11.
 - ♦ If you do want NerveCenter to change device or NerveCenter symbol colors to reflect the highest active alarm severity, select **True**.
10. For the **Set NerveCenter status on node symbol** setting, choose one of the following:
 - ♦ If you want NerveCenter to change the color of the NerveCenter icon in the submap beneath the device symbol, select **False**.

If you select False, NerveCenter automatically creates a new NerveCenter icon in the submap beneath the symbol for each managed device on the map. Color changes are applied to the NerveCenter symbol, and the color of the device symbol itself is changed according to the normal settings for OpenView's map.
 - ♦ If you want NerveCenter to change a device's symbol color to reflect the highest active alarm severity, select **True**.
11. Select **Verify**.

OpenView verifies your requested changes and then displays a message at the bottom of the window.

12. Select **OK**.

The Configuration window closes, and you return to the New Map window.

13. In the Map Properties window, select **OK**.

OpenView configures the current map to handle symbol data sent by a NerveCenter inform.

Reconfiguring OpenView before Removing the NerveCenter OpenView Platform Adapter

During a typical NerveCenter OpenView Platform Adapter installation, some NerveCenter-specific information is configured into OpenView. Before you remove NerveCenter, you should reconfigure OpenView.

TO RECONFIGURE OPENVIEW

1. Shut down the NerveCenter Server and applications.
2. Do one of the following to display the map properties:

UNIX: From the Options menu in the OpenView's Root window, choose **Map Configuration: Modify Map**.

The Map Description window is displayed.

Windows: From the Map menu, select **Properties**.

The Map Properties window is displayed.

3. Do one of the following to display the NerveCenter configuration information:
 - ◆ UNIX: From the Configurable Applications list, select **LogMatrix** NerveCenter.
 - ◆ Windows: From the Applications tab, select **LogMatrix** NerveCenter.

4. Select **Configure For This Map**.

The LogMatrix NerveCenter Configuration window is displayed.

5. For **Enable NerveCenter for this map**, select **False**.
6. Select **Verify**, then **OK**.

7. Do one of the following to display the IP map configuration information:
 - ◆ UNIX: In the Configurable Applications list box, select **IP Map**.
 - ◆ Windows: From the Applications tab, select **IP Map**.

8. Select **Configure For This Map**.

The IP Map Configuration window is displayed.

9. For **Should status of nodes be IP/IPX only?**, select **True**.

10. Select **Verify**, then **OK**.

NerveCenter can no longer change symbols on the IP map.

11. Select **OK** to close the Map Description window.

12. From the Options menu, choose **Event Configuration**.

The Event Configuration window is displayed.

13. Select **LogMatrix_NerveCenter** from the Enterprise Identification list box.

All LogMatrix NerveCenter events are displayed in the Event Identification list box.

14. Select the first event and select **Edit**, then **Delete**, then **Event**. Repeat this step for every NerveCenter event.

Removing events isn't required but does leave your platform installation less cluttered with old information.

15. While **LogMatrix_NerveCenter** is still selected, select **Edit**, then **Delete**, then **Enterprise**.

16. Close the Event Configuration window.

17. Exit OpenView and stop all OpenView's processes.

Typically you can do this by typing **ovstop** at the command line and pressing **Enter**. (On Windows, you can also select **NNM Services > Stop** from the **Start** menu.)

18. On Windows, delete the LogMatrix NerveCenter registration, help, and symbol files. (On UNIX, these files will be deleted automatically.)

These files are:

- ◆ *installpath\registration\C\ncapp.reg*
- ◆ *installpath\symbols\C\Software\NCApp*
- ◆ *installpath\bin\ovdelobj*
- ◆ *installpath\lrf\ovpa.lrf*

- ◆ *installpath*\fields\C\ncapp.fields

installpath is the installation directory for OpenView.

19. Restart OpenView processes.

Typically you can do this by typing **ovstart** at the command line and pressing **Enter**. (On Windows, you can also select **NNM Services > Stop** from the **Start** menu.)

After you have reconfigured OpenView so NerveCenter is no longer integrated with it, you can remove NerveCenter.

OpenView Integration Reference

The following section includes information you may need to reference occasionally when integrating NerveCenter and HP OpenView Network Node Manager.

This section includes:

- ◆ [Command line reference for the NerveCenter OpenView Platform Adapter on page 49](#)
- ◆ [Variable Bindings for NerveCenter Informs on page 53](#)

Command line reference for the NerveCenter OpenView Platform Adapter

You can control certain aspects of the NerveCenter OpenView Platform Adapter (OVPA) from the command line. The proper syntax is:

```
ovpa [ -logLevel Error | Warning | Debug | Trace ]
      [ -traceOutputFile FileName ] [ -traceOutputStdout ]
      [ -traceResync ] [ -traceInform ] [ -traceLoadDb ]
      [ -traceParentComp ] [ -traceMapUpdates ] [ -traceConnects ]
      [ -traceAll ] [ -pc ] [ -writeParentsToFile NcHostName ]
      [ -heartbeat seconds ] [ -ignoreUnmanagedIntf ]
      [ -initNcStatusInOV ] [ -ignoreCapability ]
      [ -noResolveCommunityOnResync ]
      [ -defGlobalCommunity filename ]
      [-preferredAddrOnly]
```

**NOTE**

OpenView (OV) must be running before you can run OVPA. To issue an **ovpa** command (with or without switches) by itself and without also using **ovstart**, you must first remove the OVPA local registration file (ovpa.lrf) from the list of services that are started by OV. For more information, refer to [Configuring OpenView to Integrate with NerveCenter on page 43](#). All of these command line switches are optional.

`-logLevel Error | Warning | Debug | Trace`

Set logging level, default value is Warning.

`-traceOutputFile FileName`

Set the output of logging messages to be a file with *FileName*.

`-traceOutputStdout`

Set the output of logging messages to be standard output. This option can be used with `-traceOutputFile filename` so that logging messages can be written both to the screen and the output file.

`-traceResync`

Enable logging messages for resync (node source) and resync parents. Logging messages contain node information including parent information OVPA is sending to Nerve Center.

To use this option, the log level must also be set to either debug or trace.

`-traceInform`

Enable logging messages for status informing from NerveCenter to OpenView. It shows status change messages from NerveCenter, whether the status change information is sent to OpenView or is logged into OpenView database.

To use this option, the log level must also be set to either debug or trace.

`-traceLoadDb`

Enable logging messages for loading nodes and interfaces from the OpenView database at initialization stage. It shows each node and each interfaces are loaded.

To use this option, the log level must also be set to either debug or trace.

`-traceParentComp`

Enable logging messages for the details of how the parents are computed for each Nerve Center host.

To use this option, the log level must also be set to either debug or trace.

`-traceMapUpdates`

Enable logging messages for the processing of OpenView events regarding changes. This option displays the significant data that arrived with an OpenView event, how ovpa is processing that event, and whether network topology needs to be updated or not.

To use this option, the log level must be set to either debug or trace.

`-traceConnects`

Enable logging messages for connections from the NerveCenter server, and from OpenView. It also shows why the port is disconnected.

To use this option, the log level must also be set to either debug or trace.

`-traceAll`

Enable logging messages for resync, inform, loadDb, parentComp, mapUpdates, and connects.

To use this option, the log level must also be set to either debug or trace.

`-pc`

Enable parent computing. Without this option, no parent information is computed for any NerveCenter host, including the requests of resync parents from NerveCenter server, and the command line option `-writeParentsToFile`

`-writeParentsToFile NcHostName`

When `-pc` is turned on, OVPA computes parent-child relationships, writes to the file `NcHostName_PC.dat`, and exits.

`-heartbeat seconds`

The ovpa process and the NerveCenter server process exchange heartbeats. You can configure how often the heartbeat is sent from OVPA using the `-heartbeat` switch. If OVPA doesn't receive a heartbeat from NerveCenter in the specified time then it will close the connection to that server's port. Default heartbeat rate is 300 seconds.

`-ignoreUnmanagedIntf`

If this switch is used OVPA does not send NerveCenter server any unmanaged interfaces within a node. You may want to restrict the interfaces that NerveCenter server must poll to only interfaces that OpenView is currently managing. If you have many subobject scope alarms and nodes, this switch to improves polling performance.

`-initNcStatusInOV`

Synchronizes the status of a NerveCenter icon and its managed nodes.



NOTE

NerveCenter must be configured so that its node data source and inform host are the same. All IP filters must be empty.

`-ignoreCapability`

Determines whether OVPA keeps track of OpenView's capability information, such as IsRouter. For more information, see [Filtering Nodes from HP OpenView Network Node Manager on page 30](#).

`-ignoreCapability` prevent ovpa from keeping track of capability data. This increases ovpa's speed and decreases its memory usage.

This switch can be turn on when a user wants to force the node to be added to NC server, regardless of capabilities.



NOTE

The `-ignoreCapability` switch disables NerveCenter's ability to filter nodes based on capabilities.

`-noResolveCommunityOnResync`

When this option is turned on, OVPA does not obtain the community strings during a resync or resync parents. You can use this argument when you are sure that the community has not changed since the initial OpenView database import to improve resync performance.

`-defGlobalCommunity filename`

Instead of retrieving community strings through OpenView, you can retrieve them from a file. This option is used to give the filename where user defines default community strings.

`-preferredAddrOnly`

When you start OVPA with this argument, only the preferred SNMP address is sent from OVPA to NerveCenter on resync.

If you use this argument and the preferred SNMP address for a node is changed manually from OpenView, you must restart OVPA for OVPA to function correctly.

If you use the `-preferredAddrOnly` option, you should start HP OpenView netmon with the following command:

```
netmon -k pickSnmpAddrPolls=false
```

These netmon arguments prevent netmon from attempting to test other SNMP addresses.

`-info`

print out the version of OVPA

`-help`

Help information for OVPA

If you want these settings to take effect every time ovpa is started, edit the file `ovpa.lrf` to include these switches. On Windows, the file is located in the `OpenView/lrf` directory. On UNIX, the file is located in the `opt/OSInc/nc/OV/lrf` directory. For example, to have ovpa compute parenting information you would modify `ovpa.lrf` to read:

```
ovpa: /opt/OSInc/nc/OV/bin/ovpa:
OVs_YES_START:ovwdb:-pc:OVs_WELL_BEHAVED:5:
```

Variable Bindings for NerveCenter Informs

Depending on how its behavior models are designed, a NerveCenter detecting particular network conditions can send Inform packets to HP OpenView Network Node Manager. Although these Inform packets use TCP/IP, they are similar in content to an SNMP trap, containing trap numbers (generic and specific), an enterprise OID, and a variable-binding list. The lengthy `varbinds` contains information about the alarm that performed the Inform action, such as the name of alarm, the object the alarm was monitoring, and the names of the origin and destination alarm states.

The OpenView receiving the trap can make use of the information in the variable bindings much the same way it would use variable bindings found in an SNMP trap.

Table 3-1 explains the contents of this variable-binding list.

TABLE 3-1. Inform Trap Variable Bindings

Binding	Value
0	The name of the domain where NerveCenter is running
1	The name of the host machine running the NerveCenter Server
2	The name of the managed node associated with the alarm

TABLE 3-1. Inform Trap Variable Bindings (Continued)

Binding	Value
3	The base object associated with the alarm (for example, ifEntry for a monitored interface)
4	The base object instance associated with the alarm (for example, 4 for the fourth interface)
5	The name of the subobject. This would include the null string if the alarm is not associated with an alarm.
6	The property group assigned to the node or the subobject
7	The name of the alarm
8	The alarm's property
9	The name of the trigger that caused the alarm transition
10	The state of the alarm before the transition
11	The severity of the state of the alarm prior to the transition
12	The state of the alarm after the transition
13	The severity of the state of the alarm after the transition
14	The maximum severity of all active alarms for the managed node before this alarm transition
15	The maximum severity of all active alarms for the managed node after this alarm transition
16	The variable bindings in the poll or trap that caused the transition. These variable bindings are formatted as follows: Attribute ncTransitionVarBinds = attribute.instance=value;attribute=value;...
17	The identification number of the alarm instance

Integrating NerveCenter with IBM Tivoli Netcool/OMNIBus

NerveCenter provides the event correlation engine that helps reduce the barrage of events typically displayed in IBM Tivoli Netcool/OMNIBus, instead forwarding significant network events to IBM Tivoli Netcool/OMNIBus. Netcool/OMNIBus receives these events and distributes the information to the operators, administrators, help desk systems, or other applications responsible for monitoring the related devices or services.

In order to integrate NerveCenter with Netcool, you must have the following:

- ◆ Netcool's current NerveCenter probe. If you do not have this probe, please contact your IBM Tivoli representative.
- ◆ NerveCenter's Universal Platform Adapter. See *Installing NerveCenter* for installation instructions.

This document contains the following sections:

Title	Description
<i>Overview on page 56</i>	Introduces the NerveCenter and Netcool/OMNIBus applications and discusses how NerveCenter complements Netcool/OMNIBus.
<i>Components Required for Integration on page 63</i>	Lists and describes each application's components required for integration.
<i>How Integration Components Interact on page 65</i>	Describes briefly how NerveCenter and Netcool/OMNIBus communicate with each other.
<i>NerveCenter Configuration Settings on page 69</i>	Explains what can or must be configured in NerveCenter.
<i>Netcool/OMNIBus Configuration Settings on page 78</i>	Explains what can or must be configured in Netcool/OMNIBus.
<i>Netcool Integration Reference on page 92</i>	Provides a command line reference, variable bindings for informs, inform data, debug probe output, and a sample NerveCenter.rules file.

Overview

Recent trends within the communication industry have given rise to software tools designed to provide immediate support for real-time business and technology services. In the event of an outage, administrators can quickly determine which device caused the problem, which customers and services are impacted, and how the condition affects their service level agreements. This type of service-level management is illustrated in *Figure 4-1*.

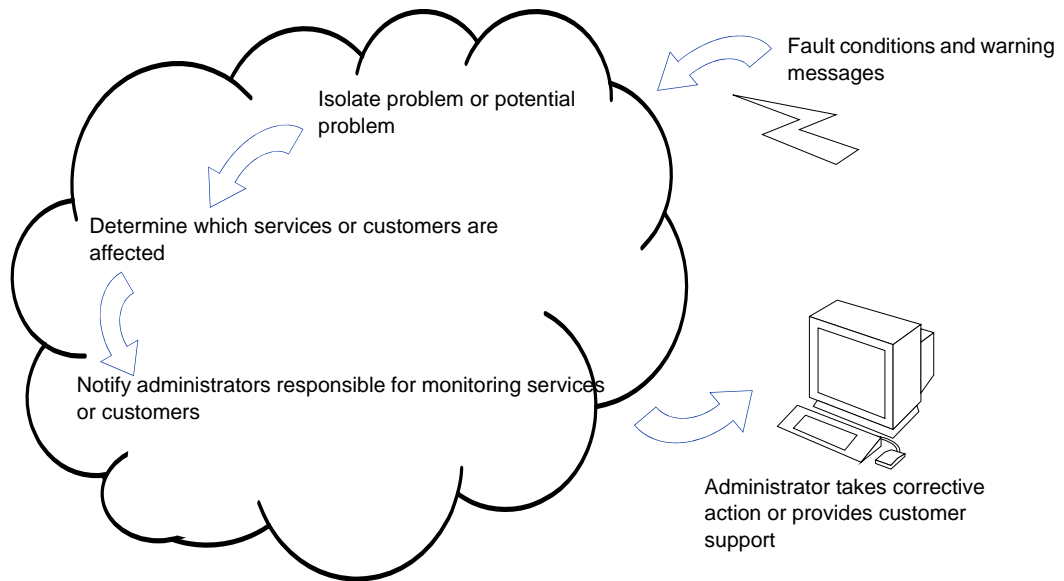


FIGURE 4-1. Service-level Management

Internet, cellular, network, and other service providers all need a way to ensure the availability of services, devices, and applications in any IT environment. Service-oriented tools provide administrators the status of business and technology services being provided to the users. These tools do not necessarily replace network management platforms; rather, they can extend a platform's capability to provide real-time service-oriented views of the network.

The IBM Tivoli Netcool/OMNIBus suite of tools has emerged as a viable solution for enterprises concerned with the availability, reliability, and quality of direct communication technologies. By partnering with key vendors like LogMatrix, IBM Tivoli enables network administrators to organize, measure, predict, and improve service levels for applications, database systems, network devices, and business-critical services such as virtual private networks or Internet connections. In the event of an outage, administrators can quickly determine which device caused the problem, which customers and services are impacted, and how the condition affects service level agreements.

What is Netcool/OMNIbus?

Netcool/OMNIbus is a suite of management tools that collect and distribute network events to the administrators responsible for monitoring related services. Netcool/OMNIbus uses specialized software agents, called probes, to intercept data coming from network systems, devices, and applications. IBM Tivoli has over 100 probes, each designed to identify, collect, and format data from a particular management environment or network application. The IBM Tivoli NerveCenter probe was developed specifically to gather network and system data from NerveCenter servers.

A probe formats events into Netcool alerts and forwards these alerts to the Object Server, Netcool's active database (*Figure 4-2*). The Object Server manipulates alerts based on user-defined associations, groups the alerts into logical units, and then directs status information to operators, administrators, help desk systems, or other applications responsible for monitoring the related services. Administrators who receive alerts can compare each alert against existing service level agreements and determine which services—and ultimately which users—are affected by particular faults. Desktop tools enable administrators to design personalized views of service availability.

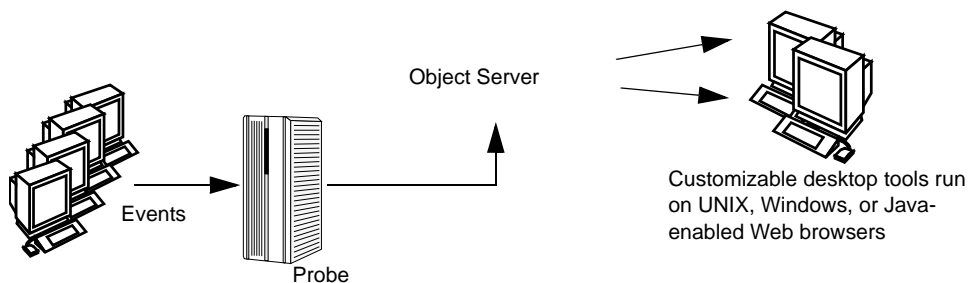


FIGURE 4-2. Netcool/OMNIbus

In addition to the Object Server, probes, and desktop tools, Netcool/OMNIbus includes:

- ◆ Gateways that allow event data to be shared with other software programs.
- ◆ Process Control systems that enable you to configure and manage UNIX processes remotely. Processes can be started in sequential order after any defined dependencies have been met.
- ◆ Java Event Lists that enable lists and views of service availability from Web browsers.

IBM Tivoli also provides the following tools:

- ◆ Netcool/Reporter uses data from the Object Server and historical databases to generate graphical reports that can be configured to match service agreement specifications.
- ◆ Netcool/CNMview provides service-level information from remote Desktops at the customer site.

- Netcool/Internet Service Monitors are active probes that provide availability information about Internet services (HTTP, FTP, DNS, SMTP, POP-3, NNTP, and RADIUS). Each Monitor periodically attempts to access a URL or perform a file transfer. It then reports the time it took to get a response or indicates that the service is unavailable.

What is NerveCenter?

NerveCenter obtains data from SNMP agents running on managed nodes by processing incoming SNMP traps and polling nodes for specific MIB values. NerveCenter interprets and correlates this data to detect predefined network conditions and determine which actions should be performed.

Behavior Models

To correlate network data, NerveCenter relies on configurable network and system models, or behavior models, for each type of managed resource. A behavior model is a group of NerveCenter objects that detect and handle a particular network or system behavior. A typical behavior model consists of one or more alarms with all their supporting polls and masks (*Figure 4-3*).

When a predefined network condition is detected, NerveCenter generates alarm instances that track the status of the interface, node, or enterprise being monitored. The alarm waits for subsequent events or issues polls to determine if the condition warrants further action. Each transition in an alarm can trigger actions, including notifying an administrator or a network management platform, executing a program or Perl script, modifying the node's properties, changing SNMP values, and logging the critical data.

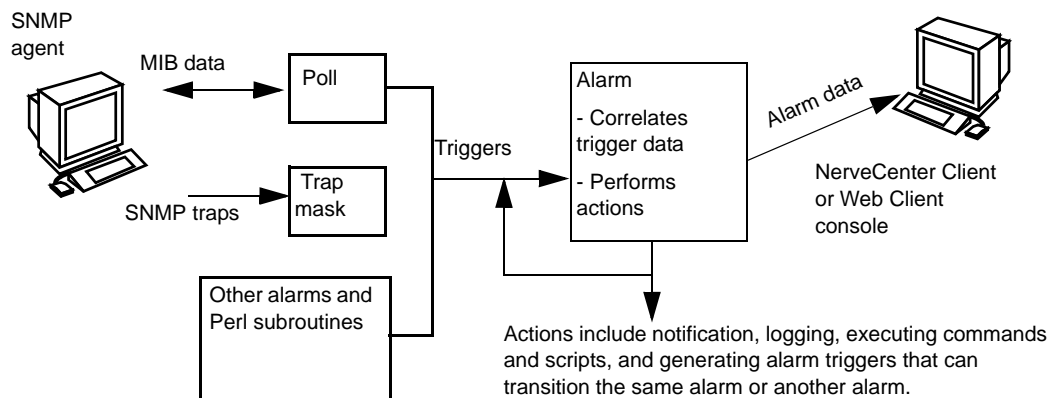


FIGURE 4-3. A NerveCenter Behavior Model

Alarms

Alarms are key to the correlation of events. Each alarm defines a set of operational states (such as Normal or Down) and transitions between the states. Transitions are caused by trigger-generating objects such as polls, trap masks, Perl scripts, or other alarms. When the alarm receives the proper trigger, one or more transitions occur. If actions are associated with a transition, the NerveCenter Server performs these actions each time the transition takes place.

Figure 4-4 illustrates an alarm that monitors each interface on managed nodes and determines whether device load is low, medium, or high. Load is the amount of interface traffic compared to the media's capacity. The *IfLoad* alarm can give an immediate impression of network and system utilization. By measuring traffic against capacity, you can determine, for example, whether more file servers need to be added to the network.

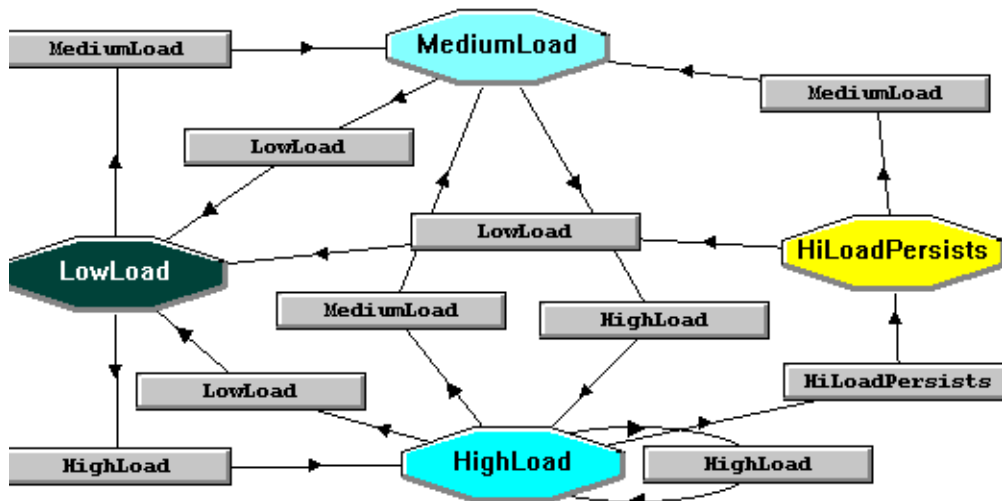


FIGURE 4-4. IfLoad Alarm

The alarm transitions to a corresponding state when it receives a MediumLoad trigger or a HighLoad trigger. The HighLoad state fires a trigger after the alarm has received its third HighLoad trigger, transitioning the alarm to the HighLoadPersists state.

Once the alarm has entered the MediumLoad, HighLoad, or HighLoadPersists state, receiving a LowLoad trigger returns the alarm to Low and clears any alarm instances.

Inform Messages

Actions can be associated with alarm transitions. NerveCenter has several actions that notify an administrator, network management platform, or another NerveCenter of an alarm transition. One important notification action is the NerveCenter inform action. An inform message contains the variable bindings associated with the event that caused the alarm to transition.

NerveCenter sends informs to Netcool/OMNIBus when an alarm transition occurs and the transition includes an inform with Netcool designated as the recipient. The NerveCenter Server forwards the inform data to Netcool's NerveCenter probe, which formats the data and sends alerts to the Object Server.

You can configure NerveCenter to send informs to Netcool when certain network conditions are detected, thereby greatly reducing the number of alerts sent to the corresponding Netcool Event List. For example, in the previous alarm (see *Figure 4-4*), NerveCenter sends an inform only when the IfLoad alarm transitions to the HiLoadPersists state.

How NerveCenter Complements Netcool/OMNIBus

NerveCenter extends Netcool's ability to measure, predict, and improve service levels in several ways.

Smart Polling

Netcool/OMNIBus is designed to respond to events, such as SNMP traps, coming from managed resources. NerveCenter not only detects and filters SNMP traps, but it can poll resources at predefined intervals for specific network and system data. This information allows administrators to track network and system behavior and identify potential problems before they occur.

NerveCenter uses a feature called smart polling to minimize unnecessary network traffic. With smart polling, NerveCenter issues polls only when the outcome of the poll can trigger an alarm. For example, if a behavior model correlates high traffic followed by high error rates, a device is not polled for error rates unless it fulfills the high traffic condition. Using this same technology, NerveCenter is able to suppress polling to nodes that are unreachable because either they or their parent devices are down.

For service-level management, NerveCenter can help track the following statistical information over local and wide area networks:

- ◆ Who is on the network, what tasks they are performing, and what tools they are using
- ◆ What applications are in use at the application and device layers and how much bandwidth they are consuming
- ◆ What is the total volume of data on different parts of the network at the busiest time of the day and whether this traffic is seasonal
- ◆ How the traffic will grow with time, taking into account increases in the number of devices or users, and changes to the applications used

This information is essential to guarantee application and network service levels to both internal and external customers.

Intelligent Correlation

Event correlation is the mechanism by which NerveCenter evaluates pre-defined events and determines how they are related, what may have caused them, and whether the condition is serious enough to notify an administrator or take corrective action. NerveCenter's complex correlation engine filters out redundant or mundane events so that only important messages are sent to probes — which facilitates network management and limits network traffic.

By reducing the amount of raw data sent to administrators, event correlation makes it easier for them to identify critical conditions quickly. Event correlation also results in the delivery of important information that administrators can use to establish baselines, monitor thresholds, determine network utilization patterns, track system performance, identify potential bottlenecks and other critical conditions, and plan for future network needs.

Distributed Architecture

NerveCenter's client-server architecture supports distributed polling across large networks. NerveCenter can be configured so that all polling is accomplished on local area networks rather than across a wide area network. Using this capability, you can reduce bandwidth and increase scalability by limiting the information to be monitored for each subnet and the number of nodes to be polled. NerveCenter servers running at remote sites can notify a centrally located NerveCenter Server or management platform of the noteworthy network conditions at those sites. Because the server can run as a daemon on UNIX systems or as a service on Windows, the branch NerveCenters can be managed remotely.

Other Advantages

NerveCenter offers the following additional advantages:

- ◆ NerveCenter can parse MIB values and obtain the variable bindings related to network events. You can extend the MIB values monitored as new devices are added to your network.
- ◆ NerveCenter's alarm actions include the ability to execute commands and scripts that can remedy the problem that caused the alarm, resulting in further reduction of the number of events that need to be reported to Netcool/OMNIBus.
- ◆ NerveCenter includes a set of predefined behavior models that you can use to monitor and manage your network. These behavior models contain all the required mask, poll, alarm, and property group definitions for basic network management using MIB-II objects. NerveCenter also ships with predefined vendor-specific models for monitoring Cisco, Compaq, and Wellfleet devices.
- ◆ NerveCenter includes a Web Client that makes it easy to monitor alarms from any machine that has a Web browser.
- ◆ NerveCenter includes tools for generating reports about the network.

Integrating Netcool with the NerveCenter Universal Platform Adapter

As long as the NerveCenter Universal Platform Adapter and the Netcool's NerveCenter probe are installed and running, NerveCenter can send inform packets to IBM Tivoli Netcool/OMNIBus. There are a few additional ways you can configure the integration between NerveCenter and Netcool. Some of these configurations involve altering Netcool's NerveCenter.rules file, which is typically found in `$OMNIHOME/probes/platform/`, where `$OMNIHOME` is the installation directory used by IBM Tivoli, which is typically `/opt/Omnibus`. See page 98 for a sample NerveCenter.rules file.

The following list describes some ways you can customize your Netcool/NerveCenter integration.

- ◆ Should the platform adapter's host or port change, you must configure the platform adapter settings accordingly. See [Command line reference for Integrating NerveCenter with Netcool on page 92](#) for further instructions. You may also need to reconfigure the Inform Configuration setting in NerveCenter Administrator to reflect the new host or port. See [How to Specify the Destination of Inform Packets Sent to IBM Tivoli Netcool/OMNIBus on page 70](#).
- ◆ If you change the probe's host so that it differs from the machine on which the platform adapter resides, you must change the default `-nhost` setting for the adapter to the machine that is running the probe. See [Command line reference for Integrating NerveCenter with Netcool](#)

[on page 92](#) for further instructions. You may also need to change your Netcool configuration. See your IBM Tivoli Netcool/OMNIBus documentation for further details.

- ◆ You may want to create a new event list or modify an existing list to display only traps received from NerveCenter. Creating or modifying a list would involve creating a new filter that displays only informs sent from NerveCenter. See your IBM Tivoli Netcool/OMNIBus documentation for further instructions.
- ◆ You may want to customize an Event List to display fields in the NerveCenter.rules file.
- ◆ By default, messages received from NerveCenter display in the Event List as “See Details.” You may want to associate more meaningful messages with NerveCenter informs.

To create more meaningful messages, edit the **@Summary field** value in the NerveCenter.rules file.

- ◆ You can create a new or change an existing NerveCenter inform recipient at any time. See [How to Specify the Destination of Inform Packets Sent to IBM Tivoli Netcool/OMNIBus on page 70](#) for further instructions.

Components Required for Integration

This section summarizes the main components required for NerveCenter integration with Netcool/OMNIBus. The summary does not attempt to list all the components that you should install for each product. Rather, it describes those components that are involved with NerveCenter-Netcool communication or that should be configured specifically for integration.



NOTE

Be sure you have compatible versions of both products before integrating NerveCenter with Netcool/OMNIBus. Contact your sales representative for information about recent versions and patches.

For a full description of all components available for NerveCenter and Netcool/OMNIBus, refer to each product's documentation.

IBM Tivoli Netcool/OMNIBus Components

The Netcool/OMNIBus components listed in *Table 4-1* must be configured for integration:

TABLE 4-1. IBM Tivoli Components Required for Integration

Component	Description
Netcool/OMNIBus Object Server	The Object Server is an active database that stores and manages all Netcool events. Events are passed to the Object Server from external programs such as probes and gateways. The Object Server filters out redundant events and make decisions about the data based on user-defined parameters.
Netcool/OMNIBus NerveCenter probe	The NerveCenter probe is distributed by IBM Tivoli and is neither a LogMatrix product nor a component of NerveCenter. The probe collects network data from NerveCenter and then formats and forwards that data to the Object Server. The probe has associated rules and properties that define how the probe operates and how NerveCenter events are mapped to Netcool alerts. Note: The NerveCenter probe is not included with a standard Netcool/OMNIBus package but must be obtained separately from IBM Tivoli. Probes are identified by the platform and version of NerveCenter you are using.
Netcool/OMNIBus desktop tools	Desktop tools provide event lists and views of service availability. By manipulating the data using Netcool utilities, you can design personalized views of network services and devices managed by NerveCenter.

LogMatrix NerveCenter Components

LogMatrix NerveCenter is a distributed client/server application that includes a server, files used for storing NerveCenter data, user-interface software, and several additional tools. The NerveCenter components listed in [Table 4-2](#), all part of the standard package, are required for integration:

TABLE 4-2. NerveCenter Components Required for Integration

Component	Description
NerveCenter Server	The Server carries out all the major tasks that NerveCenter performs and manages NerveCenter communications and processes.
Universal Platform Adapter	NerveCenter's Universal Platform Adapter establishes a connection with Netcool/OMNIBus and relays events from the NerveCenter Server to Netcool's NerveCenter probe.
NerveCenter Administrator	After installing NerveCenter, you use the NerveCenter Administrator to configure various settings for the connected Server. These settings include the host name and port number of the machine on which NerveCenter's Universal Platform Adapter resides. This information is required for sending informs to Netcool/OMNIBus.
NerveCenter Client	The NerveCenter Client lets you monitor the network as well as create and modify the behavior models managed by the NerveCenter Server. In the Client, you set up alarm actions that send NerveCenter informs to Netcool when defined network conditions are detected.

How Integration Components Interact

For integration to occur, NerveCenter must be configured to send informs to Netcool/OMNIBus, and Netcool/OMNIBus must be configured to receive and process those informs. See [NerveCenter Configuration Settings on page 69](#) and [Netcool/OMNIBus Configuration Settings on page 78](#) for details.

The following must be running:

- ◆ Netcool/OMNIBus Object Server
- ◆ Netcool/OMNIBus NerveCenter probe
- ◆ NerveCenter Server
- ◆ NerveCenter Universal Platform Adapter

You may also want to start the Netcool/OMNIBus desktop applications, especially the Event List, to view messages and alerts.

Once the applications are running, the probe immediately creates a TCP socket and listens for connections from the NerveCenter platform adapter process. When the adapter sends a connection request, the probe confirms the connection.

The following illustrations show the components and their paths of communication for a simple integration configuration.



NOTE

The illustrations do not suggest the physical location of the components, which can all reside on the same system or on different systems. However, it's worth noting that installing the NerveCenter probe on the same machine as the NerveCenter Server and platform adapter can reduce network traffic, since all messaging among those components is contained within a single system.

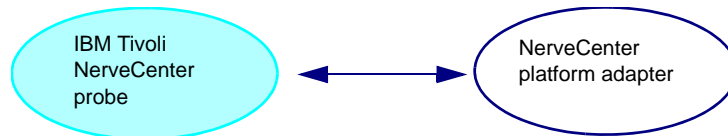


FIGURE 4-5. The Probe Connects with NerveCenter's Platform Adapter

When the two applications have established a connection, the probe relays the connection status to the Netcool Object Server, and the Netcool Event List viewer displays the status messages received from the Object Server. The following sample Event List window shows the messages received when the probe is started and connects with the Universal Platform Adapter (pserver.exe).

Node	Summary
	Connection accepted from NC_PAServer_cobalt
cobalt	A Probe process running on cobalt has connected.
cobalt	nervecenter probe on cobalt: Running ...

FIGURE 4-6. Event List Messages Upon Connection

After the probe and adapter establish a connection, NerveCenter sends informs to Netcool/OMNIBus. The NerveCenter Server issues an inform when an alarm transition occurs and the transition includes an inform action targeted for Netcool/OMNIBus. The Server sends the inform to the platform adapter, which passes the inform data to Netcool's NerveCenter probe.

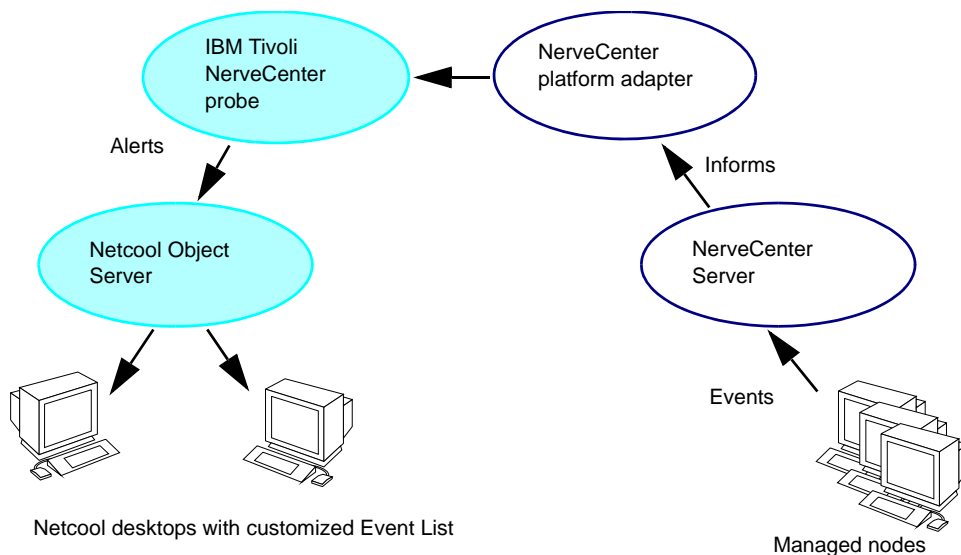


FIGURE 4-7. NerveCenter Sends Informs to Netcool/OMNIBus

The probe receives the inform data and generates an identifier that uniquely identifies the event. The probe also converts the inform to an alert format that the Object Server can recognize and then forwards the alert to the Object Server.

The Object Server stores alerts in an alert table that is part of its active database. The Object Server can manipulate alerts by associating them into classes, filtering them, and assigning automated actions to them. Based on defined parameters, the Object Server determines the destination for each alert and forwards the information to the appropriate desktop applications.

The Event List displays the alerts forwarded from the Object Server, as seen in [Figure 4-8](#).

Node	Summary	Last Occurrence	Count
	Connection accepted fro	06/16/99 12:39:30	1
ATHENA	See details	06/16/99 12:32:10	1
ATHENA	See details	06/16/99 12:34:51	19
BLUERIDGE	See details	06/16/99 12:32:02	1
BLUERIDGE	See details	06/16/99 12:34:53	20
CAREY	See details	06/16/99 12:34:49	19
CAREY	See details	06/16/99 12:32:08	1
HATTERAS	See details	06/16/99 12:32:08	1
HATTERAS	See details	06/16/99 12:34:49	19
MOZART	See details	06/16/99 12:32:10	1

FIGURE 4-8. Netcool/OMNIBus Event List



NOTE

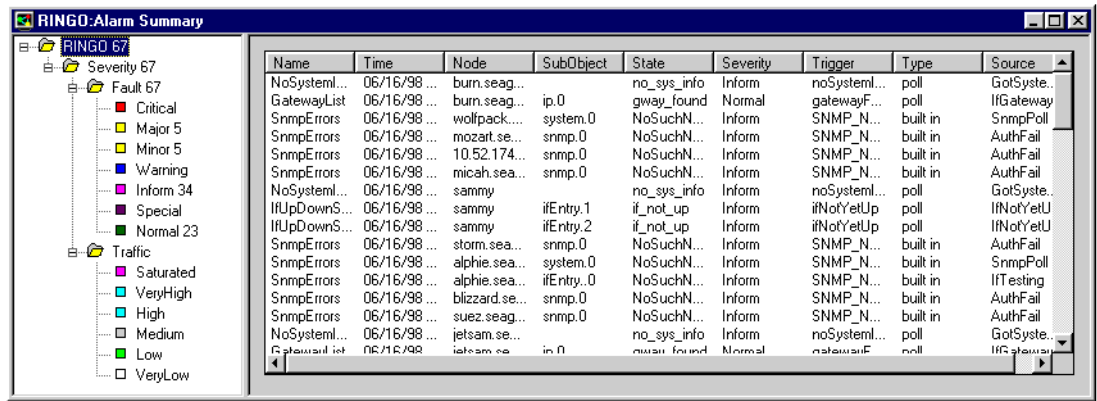
Double-click an alert to see all information for a particular inform. See [Inform Messages on page 60](#) for a description of the data sent with NerveCenter informs.

A heartbeat message monitors the connection between the probe and the platform adapter. If the probe goes down, its new operational status appears immediately in the Netcool Event List. Once the probe comes back up, the platform adapter attempts to reestablish the connection.

When the platform adapter goes down or comes back up, the following occur:

- The Netcool Object Server immediately sends the status update to the Event List, which displays a message indicating the current connection status.
- After a configurable amount of time, the NerveCenter platform adapter notifies the NerveCenter Server of the status, and the Server sends a message to all connected clients.

While NerveCenter alerts are being monitored in Netcool/OMNIBus, you can also view the original alarms in the NerveCenter Client or Web Client. [Figure 4-9](#) shows a sample Alarm Summary window in the NerveCenter Client.



The screenshot shows the RINGO Alarm Summary window. On the left is a tree view with 'Severity 67' expanded, showing categories like Fault 67 (Critical, Major 5, Minor 5, Warning, Inform 34, Special, Normal 23) and Traffic (Saturated, VeryHigh, High, Medium, Low, VeryLow). The main area is a table with the following columns: Name, Time, Node, SubObject, State, Severity, Trigger, Type, and Source.

Name	Time	Node	SubObject	State	Severity	Trigger	Type	Source
NoSysteml...	06/16/98 ...	burn.seag...		no_sys_info	Inform	noSysteml...	poll	GotSyste..
GatewayList	06/16/98 ...	burn.seag...	ip.0	gway_found	Normal	gatewayF...	poll	IFGateway
SnmpErrors	06/16/98 ...	wolfpack...	system.0	NoSuchN...	Inform	SNMP_N...	built in	SnmpPoll
SnmpErrors	06/16/98 ...	mozart.se...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
SnmpErrors	06/16/98 ...	10.52.174...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
SnmpErrors	06/16/98 ...	micah.sea...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
NoSysteml...	06/16/98 ...	sammy		no_sys_info	Inform	noSysteml...	poll	GotSyste..
IFUpDownS...	06/16/98 ...	sammy	ifEntry.1	if_not_up	Inform	ifNotYetUp	poll	IFNotYetU
IFUpDownS...	06/16/98 ...	sammy	ifEntry.2	if_not_up	Inform	ifNotYetUp	poll	IFNotYetU
SnmpErrors	06/16/98 ...	storm.sea...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
SnmpErrors	06/16/98 ...	alphie.sea...	system.0	NoSuchN...	Inform	SNMP_N...	built in	SnmpPoll
SnmpErrors	06/16/98 ...	alphie.sea...	ifEntry.0	NoSuchN...	Inform	SNMP_N...	built in	IFTesting
SnmpErrors	06/16/98 ...	blizzard.se...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
SnmpErrors	06/16/98 ...	suez.seag...	snmp.0	NoSuchN...	Inform	SNMP_N...	built in	AuthFail
NoSysteml...	06/16/98 ...	jetsam.se...		no_sys_info	Inform	noSysteml...	poll	GotSyste..
GatewayList	06/16/98 ...	jetsam.se...	in.0	gway_found	Normal	gatewayF...	poll	IFGateway

FIGURE 4-9. NerveCenter Client Alarm Summary Window

The Client has filters that reduce the alarms to those matching specified property groups, severity levels, and IP subnets for associated servers. For more information about monitoring alarms, refer to *Monitoring Alarms in Monitoring Your Network*.

NerveCenter Configuration Settings

This section explains which NerveCenter components are configured to integrate with Netcool/OMNIBus and how each component should be configured.

NerveCenter configuration involves specifying the ports that allow the NerveCenter Server and adapter to transfer data to Netcool/OMNIBus. NerveCenter must be set up to detect noteworthy conditions and send inform actions to Netcool/OMNIBus. Procedures for entering these settings are included in the NerveCenter documentation, as well as the rights and privileges required for configuring NerveCenter applications.



NOTE

Verify all component versions before configuring NerveCenter. Contact your sales representative for information about recent versions and patches, which you can download from the [LogMatrix](#) site if you have an active maintenance contract.

NerveCenter Server Inform Port Settings

After installation, you enter settings in the NerveCenter Administrator to specify which hosts are to receive NerveCenter informs. When setting up a Netcool/OMNIBus recipient host, you must provide the host name and the port number to use for sending informs to the Universal Platform Adapter (see *Universal Platform Adapter Settings on page 73*). The default port is 32509.

How to Specify the Destination of Inform Packets Sent to IBM Tivoli Netcool/OMNIBus

One of the most powerful characteristics of NerveCenter's platform integration is how it informs IBM Tivoli Netcool/OMNIBus. Since NerveCenter's inform packets use the Transmission Control Protocol (TCP), the alert sent to Netcool is more reliable than a standard SNMP trap. As you create or modify a behavior model to notify Netcool, you determine the specific inform number it will receive. However, before you can use this behavior model, NerveCenter must know which machine or machines will receive the inform.

While setting up the inform configuration, you can specify a minimum severity level for informs or limit the informs to those with particular property groups. The following procedure will step you through the process of declaring one or more recipients of NerveCenter informs.

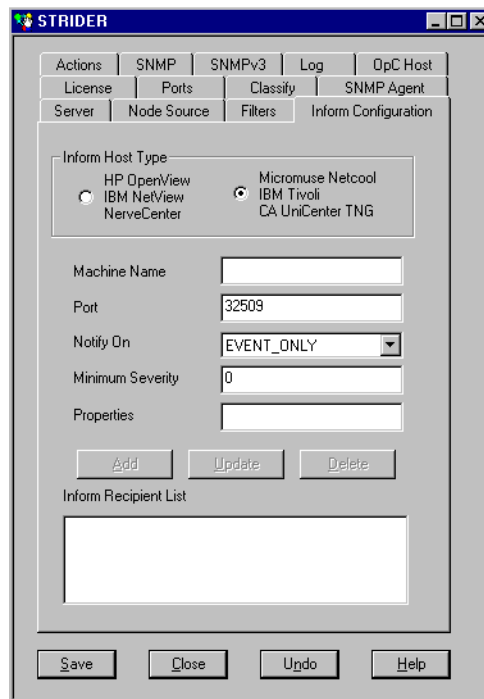
TO SPECIFY THE DESTINATION OF NERVECENTER INFORMS SENT TO NETCOOL

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

If you need help opening NerveCenter Administrator or connecting to a NerveCenter Server, see *Connecting to a NerveCenter Server in Managing NerveCenter*.

2. Select the **Inform Configuration** tab.

The Inform Configuration tab appears.



3. In the **Inform Host Type** field, select the radio button beside Netcool.

Selecting this option enables the NerveCenter Server to recognize the inform recipient as a platform needing the NerveCenter Universal Platform Adapter.

4. In the **Machine Name** field, type the name of the machine hosting Netcool.
5. In the **Port** field, type the port number your NerveCenter Server will use when communicating with the NerveCenter Universal Platform Adapter.
By default, NerveCenter uses the port number 32509.
6. In the **Notify On** field, select **EVENT_ONLY**.

NerveCenter send events to Netcool when the Inform Platform action is invoked.

7. In the **Minimum Severity** field, type the number representing the minimum severity an alarm must reach before triggering a message to this platform.

This option enables you to be selective about which events are sent to particular platforms. For example, a local platform could get all events, while a lead or central platform could get only critical events. When NerveCenter sends Informs to your platform, NerveCenter first

checks the minimum severity value entered here to ensure that the trap value for the Inform matches or exceeds that severity.

**NOTE**

There is one case when NerveCenter disregards the minimum severity value specified in Administrator: After NerveCenter sends an Inform, if the condition returns to a normal state—that is, a state below the minimum severity threshold you configure—it's important that NerveCenter notify the platform of this change. Therefore, if a node transitions the alarm from a severity above the minimum value to a severity below the minimum value, and the transition includes an Inform action, NerveCenter will send a Normal Inform to the platform. This allows the platform to reset the mapped severity color associated with the node.

**NOTE**

The values associated with each severity in NerveCenter can be viewed and altered in the NerveCenter Client in the Severity List found under the Admin menu.

8. In the **Properties** field, type zero or more properties.

NerveCenter will only send an Inform packet to this platform if the managed node's property group contains at least one of the properties listed in this field. If no events are listed, NerveCenter sends events for all managed nodes.

This option enables you to be selective in which events are sent to particular platforms. For example, one platform could receive informs only prompted by routers.

9. Select **Add**.

The platform's host machine is added to the Inform Recipient list.

10. Repeat steps 3 through 9 for each different machine hosting a network management platform that will receive a NerveCenter inform packet.
11. Select **Save**.

When an alarm performs an Inform Platform action, the relevant platforms included in this list will receive the inform data.

Universal Platform Adapter Settings

During installation, the platform adapter is configured with default settings that specify the adapter's host machine and the ports used to communicate with the NerveCenter Server and the NerveCenter probe. Depending on your configuration, you may need to change the default settings.



NOTE

If you install the Universal Platform Adapter on a different machine from the one on which the probe is installed, you must change the adapter's default `-nhost` setting to the machine that is running the probe.

The command to change the platform adapter settings resembles the following:

```
paserver -o -p listeningport -n ON -nhost hostname -nport sendingport
```

The command contains the switches shown in [Table 4-3](#):

TABLE 4-3. Switches for Reconfiguring the NerveCenter Platform Adapter

Switch	Description
-o	Windows only, records values into the registry. Any options (other than <code>-scm</code>) become a part of the standard configuration. To use this switch, you should first stop the Universal Platform Adapter. You must then restart the Universal Platform Adapter.
-p	Defines the platform adapter's listening port. The adapter uses this port to communicate with the NerveCenter Server. Note: This number must match the port number specified in NerveCenter Administrator for sending informs. The default is 32509.
-n ON OFF	Enables or disables NerveCenter integration with IBM Tivoli Netcool/OMNIbus.
-nhost	Defines the machine on which the NerveCenter probe is located. Note: The default is the local host where the adapter is installed. If the adapter and probe are on different machines, this value must be changed.
-nport	Defines the port the NerveCenter platform adaptor uses to communicate with the probe. The default is 32510. See NerveCenter.props file on page 79 for more information about probe settings. Note: This number must match the number used by the probe to communicate with NerveCenter, as specified in the probe's property file. Some early versions of the probe used port 10001; these should be replaced with the current version.

Figure 4-10 summarizes the port settings for NerveCenter-Netcool communication.

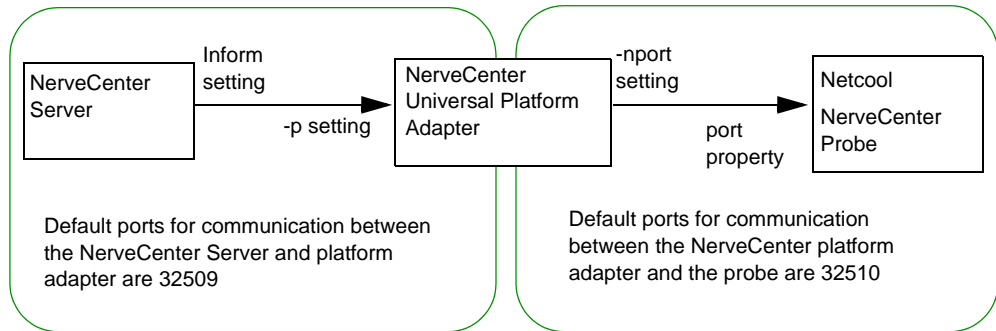


FIGURE 4-10. Ports for Communication Between NerveCenter and Netcool/OMNIBus

How to Start and Stop the Universal Platform Adapter

The NerveCenter Universal Platform Adapter is typically installed as a Windows service or UNIX daemon so it starts automatically when the IBM Tivoli Netcool/OMNIBus host machine boots.

There may be times when you want to start or stop the Universal Platform Adapter manually. The procedure depends on your operating system:

- ◆ [Starting and Stopping the Universal Platform Adapter in UNIX on page 74](#)
- ◆ [Starting and Stopping Platform Integration with IBM Tivoli Netcool/OMNIBus in Windows on page 75](#)

Starting and Stopping the Universal Platform Adapter in UNIX

During a typical installation, the NerveCenter Universal Platform Adapter is installed as a daemon. Therefore, whenever you boot Netcool's host machine, the NerveCenter Universal Platform Adapter will automatically start.

However, there may be times when you want to start or stop the Universal Platform Adapter manually.

- ◆ To start the Universal Platform Adapter, type at the command line:

```
pastart
```


**NOTE**

The command **pastart** is a script that runs paserver with set options. To change those options edit the script **pastart**. See [Command line reference for Integrating NerveCenter with Netcool on page 92](#) for further instructions.

- ◆ To stop the Universal Platform Adapter, type at the command line:

```
pastop
```

Starting and Stopping Platform Integration with IBM Tivoli Netcool/OMNIBus in Windows

During a typical installation, the NerveCenter Universal Platform Adapter is installed as an automatic Windows service. Therefore, whenever you boot Netcool's host machine, the NerveCenter Universal Platform Adapter will automatically be started.

However, there may be times when you wish to start or stop the NerveCenter Universal Platform Adapter manually.

- ◆ To start the Universal Platform Adapter, type at the command line:

```
paserver -n ON
```

- ◆ You can stop the Universal Platform Adapter in the Windows Services applet in the Control Panel or by typing at the command line:

```
paserver -n OFF
```

For a complete discussion of command line switches and how they are used, see [Command line reference for Integrating NerveCenter with Netcool on page 92](#).

Inform Action Settings

After installation, you can customize or create new behavior models in the NerveCenter Client. The alarms used in behavior models define the types of actions performed when specific network conditions are detected. For each alarm you want to forward to Netcool/OMNIBus, you must define an inform action in the corresponding alarm.

Figure 4-11 shows a sample alarm along with the dialog box used to define transitions and configure actions for the transitions. Also shown is the pop-up menu containing available actions.

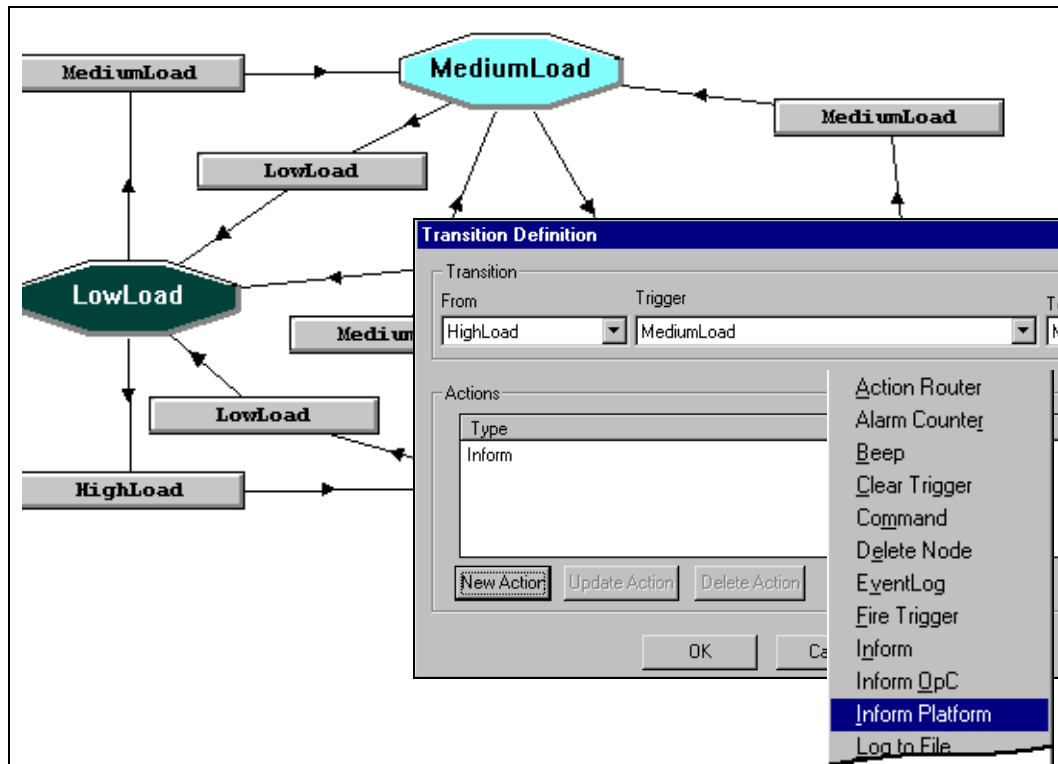


FIGURE 4-11. IfLoad Alarm and Transition Definition Dialog Box

Figure 4-12 shows the dialog box used to define inform actions.

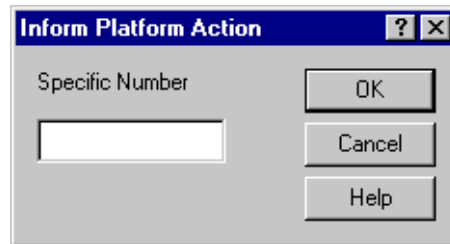


FIGURE 4-12. Inform Action Dialog Box

When creating an inform action, you have the option of providing a specific inform number. This number becomes the \$MesgID value sent with each inform and helps Netcool/OMNIbus identify the type of event. If you don't provide a specific number, the message ID defaults to the value 1000 for any transition whose destination state has a severity less than 9 (Warning). For severity levels of 9 or greater, the \$MesgID defaults to the value 1000 plus the destination state's severity level.



NOTE

Although the message that the inform action sends to its recipients contains the same information as a trap, the message is not sent via UDP. Because the delivery mechanism must be reliable, the message is sent via TCP.

You can define one or more informs for as many alarms as you want. Once you have defined the inform action and enabled an alarm, the inform is sent each time the associated transition occurs. For more information about designing behavior models using the Client, refer to *Designing and Managing Behavior Models*.

Netcool/OMNIBus Configuration Settings

This section explains how to configure Netcool/OMNIBus components to integrate with NerveCenter. Netcool/OMNIBus configuration involves specifying the type of information you want processed when Netcool receives a NerveCenter inform. You can also specify how the Netcool/OMNIBus Object Server should classify and distribute alerts and automated actions.

The \$OMNIHOME directory mentioned in this section is the directory where Netcool/OMNIBus is installed (/opt/Omnibus for Solaris 2.x, for example).

Complete configuration procedures are included in the Netcool/OMNIBus documentation. The IBM Tivoli documentation also includes information about the rights and privileges required for access.

Object Server Data Management Settings

After installing Netcool/OMNIBus, you create an interfaces file that specifies the server name, host name, and port for the Object Server, the proxy server, the gateway if installed, and the process control. Without this file, neither the probe nor any other component can communicate with the Object Server.

Any changes to your configuration are made to this file using the Servers Editor (run \$OMNIHOME/bin/ncs_xigen).

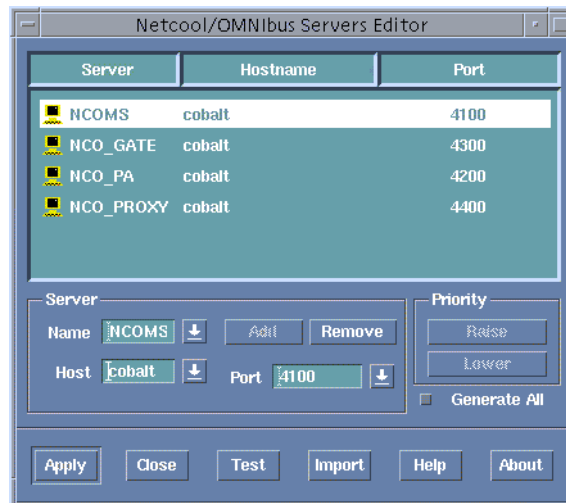


FIGURE 4-13. Servers Editor

NerveCenter Probe Settings

The NerveCenter probe transfers data between the NerveCenter Universal Platform Adapter and the Netcool/OMNIBus Object Server. A probe has two associated files that determine the probe's behavior:

TABLE 4-4. NerveCenter Probe Files

File	Description
NerveCenter.props	Identifies the probe and displays the location of associated files, the port that connects with the NerveCenter platform adapter, and other information about the probe.
NerveCenter.rules	Defines the precise set of information relayed to the Object Server.

Both files are stored in the `$OMNIHOME/probes/platform` directory. As with the probe, both files are written in a IBM Tivoli proprietary scripting language.

You can design *.props and *.rules files of your own and command the probe to use those files when you start the probe. Refer to the Netcool/OMNIBus *Probe and Gateway Reference* manual for more information about probes. After making changes to these files, you must stop and restart the probe before it can recognize the changes you made.

NerveCenter.props file

The properties file contains default settings for the NerveCenter probe. You can change the properties using the Properties Editor (`$OMNIHOME/bin/nco_xprops`). If preferred, you can run the Properties Editor when the Object Server is not running.

Say you wanted to raise the severity level of messages logged from the NerveCenter probe. Messages are logged based on the message and a logging level. There are five message levels:

- ◆ Fatal
- ◆ Error
- ◆ Warning
- ◆ Info
- ◆ Debug

To change the minimum severity level of messages that are logged, you could change the message level value from “Warn” to “Error”.

The following illustration shows the NerveCenter.props values in the Properties Editor.



FIGURE 4-14. Properties Editor

If all the properties are not listed in the Properties Editor, you can open the NerveCenter.props file directly using your text editor. In the text editor, you must uncomment a line before any changes you make to the line take effect.

CAUTION

The NerveCenter.props file lists the port used by the probe to communicate with the NerveCenter Universal Platform Adapter. For NerveCenter to communicate with Netcool/OMNIBus, this port number must match the `-nport` setting defined for the NerveCenter Universal Platform Adapter. If they don't match, you must change the adapter setting. See *Universal Platform Adapter Settings on page 73* for details.

You can set or override some probe property settings from the command line when you start the probe. This is described in the Netcool documentation.

NerveCenter.rules file

The NerveCenter.rules file defines the precise set of information relayed to the Object Server. You can modify the rules file to specify how the probe maps NerveCenter events to Netcool/OMNIBus alerts. Prior to reading the file, the probe maps the message's attributes to the fields of an event and

creates a list of all attributes and values for insertion into the Object Server's status table. The rules allow you to supersede and change this preformed alert.

The rules file uses tokens to indicate variables, such as the node that caused the message to be sent, with a \$ symbol. The @ symbol identifies field values that are transferred to the alerts table in the Object Server database. You can define your own tokens and fields in the rules file.

The default rules file is divided into two main sections, each section is part of an *if.. else* statement. In the first section, the probe generates its own ProbeWatch events to monitor the status of the probe and display messages contingent with the particular case.

Probe Watch Code

The following sample shows this section:

```
if( match( @Manager, "ProbeWatch" ) )
{
switch(@Summary)
{
case "Running ...":
@AlertGroup = "probestat"
@Type       = 2
case "Going Down ...":
@AlertGroup = "probestat"
@Type       = 1
default:
}
@AlertKey = @Agent
@Summary  = @Agent + " probe on " + @Node + ": " + @Summary
}
```

Message Code

The second section of the rules file manages the information transmitted when a connection is established or rejected, when the connection is terminated, and when an inform message is received from NerveCenter.

The following sample shows default statements that determine what appears in the Event List when NerveCenter sends an inform:

```
case "Inform Netcool":
@Identifier = $MessageType + $ServerID + $NodeName + $IPAddr + $OSN + $OSS
+ $DSN + $DSS + $TrapGN + $TrapSN + $TrapEID
@Node = $NodeName
@NodeAlias = $IPAddr
@Summary = "See details"
@Severity = 2
```

The Identifier field maps to the data sent by NerveCenter for each inform and uniquely identifies the inform event. The Identifier field also enables the elimination of duplicate alerts. If the Netcool Object Server receives two informs with the exact same identifier values, only the first inform is forwarded to the Event List. Netcool processes the duplicate inform but does not display it as a separate event instance.

Suggestions for Customizing the Rules File

You can customize the rules file supplied by IBM Tivoli to optimize your own network management strategy. The Netcool/OMNIBus *Probe and Gateway Reference* manual describes a host of statements and functions that help you manage network data sent to the Object Server and relayed to the Event List. To modify inform data, you would first define any tokens and fields you need and then add the statements and functions to the code residing within the Inform Netcool case.

For example, you can filter unnecessary events using the Netcool *Discard* function. When used prudently, this function is an effective way to prevent an inform from being sent to the Object Server. Other functions enable you to recover discarded alarms, compare variables with strings, extract portions of strings or fields, perform mathematical operations, and insert information into an event using a table format consisting of keys and values.



CAUTION

Make a backup copy of the default NerveCenter.rules file before modifying that file. Changes you make to the rules file affect the probe's compatibility with incoming NerveCenter inform data. As with customizing any software, you should know both the Netcool and NerveCenter products thoroughly and test each change you make to the rules file before proceeding with further changes.

When making changes to a rules file, you must follow the established Netcool syntax. If the syntax in a rules file is incorrect, the probe cannot be started. Netcool/OMNIBus includes a syntax probe that you can use to test rules file syntax.

You may want to make the following changes to the Inform Netcool section of the rules file:

- ◆ Add a comment symbol (#) in front of the line that contains the following text:

```
details($*)
```

This line of code passes to the Object Server a set of variables (\$*) and their values for each inform. These variables are then displayed in the Alert Details portion of an alert in the Event List. While this information is useful for the development and debug of a rules file, the extra data can overload the Object Server once you start processing large numbers of events. (Commenting this line has no affect on the @ field values sent to the Object Server.)

- ◆ If you choose to keep the `details()` statement for one or more types of detected conditions, you may want to change the message associated with the details. By default, messages received from NerveCenter display as “See Details” in the Event List. You can associate more meaningful messages with events by replacing the `@Summary` value with some other text string or with a variable, such as `$MessageType`.
- ◆ Define a class of alerts that you can later use to group NerveCenter informs. To do this, you associate an arbitrary number with the class you define, for example:

```
@Class = number
```

- ◆ Change the severity level associated with informs. There are a total of five possible levels. To change the severity from Warning to Major, for example, you would replace the `@Severity` value with 4.
- ◆ Make more information available to the Event List by adding new fields to the code. The following example would enable you to filter or group alerts by NerveCenter Server:

```
@AlertKey = $ServerID
```

or, to filter or group alerts by the specific inform number you provided when creating the inform action in NerveCenter, you could enter the following:

```
@AlertKey = $MesgID
```

- ◆ Change the data associated with informs by changing the variables in the `@Identifier` field.

When changing the `@Identifier` field values, it’s important to make sure the identifier is specific enough to filter unwanted duplicate alerts without overloading network traffic. For example, adding more variables to the identifier code above (see [Message Code on page 81](#)) would lessen the probability of exact duplicates, resulting in fewer deletions. This change, however, would also generate more messages managed by the Object Server. This may overload the Object Server with events relating to conditions that may in fact be redundant.

On the other hand, an identifier that doesn’t contain enough fields might filter out important, non-duplicated events.

For example, say you were to change the identifier definition to the following:

```
@Identifier = $MessageType + $ServerID + $NodeName+ $IPAddr
```

The above code excludes the variables that identify origin or destination state for informs. As a result, the Object Server sends an alert to the Event List upon receipt of an alarm instance for a managed node. The Object Server does not, however, forward alerts for subsequent transitions of the same alarm instance to different destination states. The subsequent informs have data identical to the first instance and are therefore filtered out.

- ◆ Finally, you can enhance or change the summary information for the probe status by changing or adding text in the Probe Watch section of the files.

For a list and description of all the variables that are sent with NerveCenter informs, see *Inform Messages on page 60*. For other ways to configure the rules file, refer to the Netcool/OMNIBus *Probe and Gateway Reference* manual. *Sample NerveCenter.rules File on page 98* contains a sample rules file you can use as a reference.

Desktop settings

When you start the Netcool/OMNIBus desktop tools, the Conductor is the first window you see. From the Conductor, you can access tools that filter events, customize how the events are displayed in the Event List, associate informs with a particular class, automate commands and actions for informs, and generate service-level views for specified geographical regions.

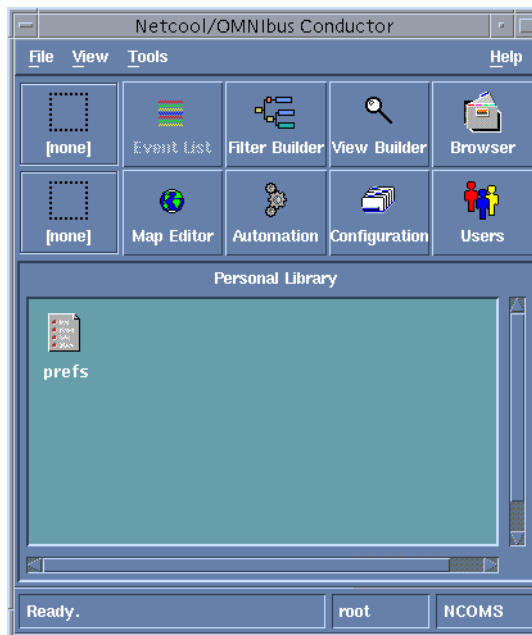


FIGURE 4-15. Netcool/OMNIBus Conductor

The following sections describe the different ways you may want to customize your desktop for NerveCenter events. Refer to the Netcool/OMNIBus *Administration Guide* for complete information about these and any other settings.

Filtered Event Lists

You may want to create a new event list or modify an existing list to display only informs received from NerveCenter. Creating or modifying a list involves creating a new filter that displays certain types of alerts—in this case, informs sent from NerveCenter. Filters limit the information that you receive on your desktop.

The Filter Builder enables you to filter incoming alerts according to the values defined for your informs. *Figure 4-16* shows the Filter Builder used to create filters.

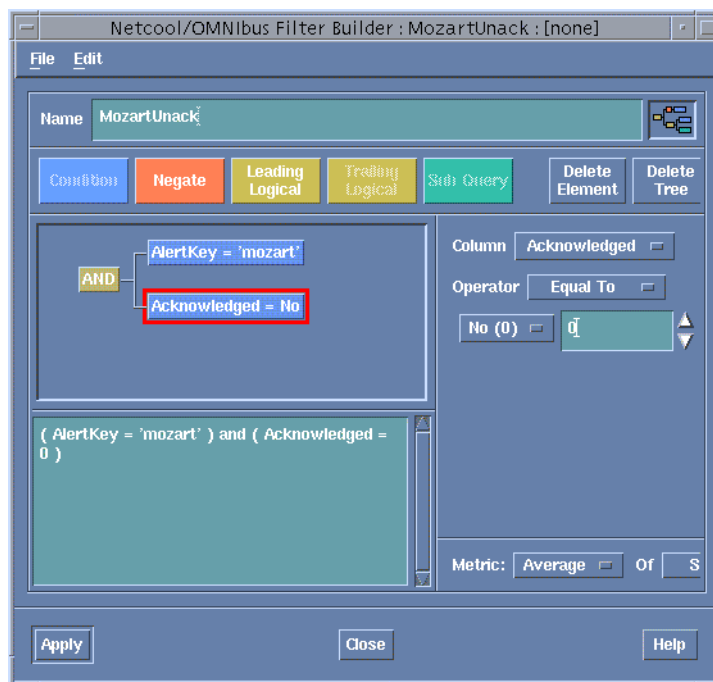


FIGURE 4-16. Filter Builder

By stringing together logical OR or AND conditions, you can generate filters based on different combinations of Event List fields.

Custom Views

Use the View Builder to customize the columns displayed in the Event List, change the column headers, and organize events by a defined sorting order.

To customize the fields, you choose from the available fields the ones you want displayed in the list. You can include fields, for example AlertKey, that you defined at the probe level in the NerveCenter.rules file.

Figure 4-17 shows sample settings in the View Builder. The settings displayed here add the AlertKey field to the Event List and sort by this field, with a secondary sort on the node name.

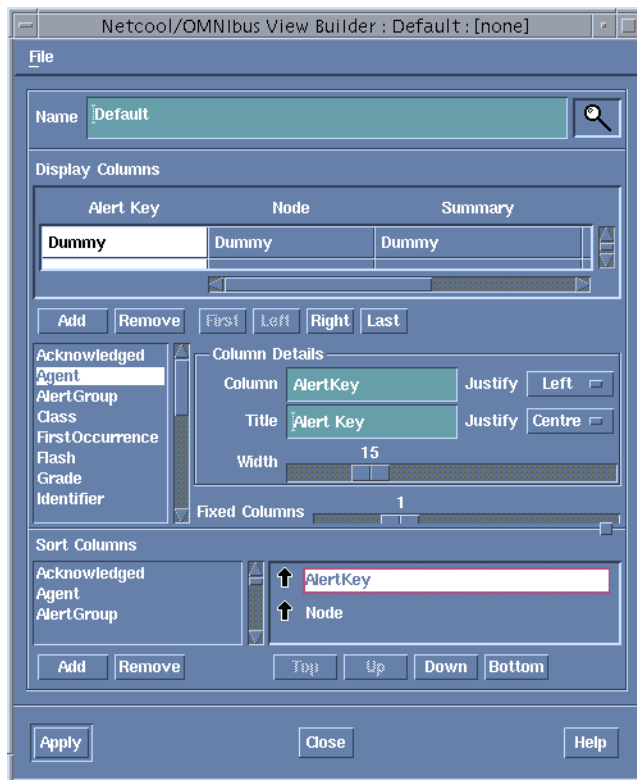


FIGURE 4-17. View Builder

The results of these view settings are shown in *Figure 4-18* in the Event List:

The screenshot shows a window titled "Netcool/OMNibus Event List : Filter='Default', View='Default'". The window contains a menu bar (File, Edit, View, Alerts, Tools, Help) and a toolbar with buttons for Filters, Edit, Views, Edit, Services, and Jump. Below the toolbar is a table with the following data:

Alert Key	Node(+)	Summary(+)	Last
NerveCenter Server: strider	10.52.174.249	Inform Netcool	06/16
NerveCenter Server: strider	10.52.174.249	Inform Netcool	06/16
NerveCenter Server: strider	10.52.174.252	Inform Netcool	06/16
NerveCenter Server: strider	10.52.174.252	Inform Netcool	06/16
NerveCenter Server: mozart	ATHENA	Inform Netcool	06/16
NerveCenter Server: mozart	ATHENA	Inform Netcool	06/16
NerveCenter Server: mozart	BLUERIDGE	Inform Netcool	06/16
NerveCenter Server: mozart	BLUERIDGE	Inform Netcool	06/16
NerveCenter Server: mozart	CAREY	Inform Netcool	06/16
NerveCenter Server: mozart	CAREY	Inform Netcool	06/16

At the bottom of the window, there is a status bar showing "No rows selected.", "06/16/99 15:20:19", "root", and "NCOMS".

FIGURE 4-18. Configured and Sorted Event List

Alert classes

You can tag NerveCenter events with a class value, which is assigned in the NerveCenter.rules file. Once you have associated NerveCenter alerts with a specific class, you can create and associate custom menus with the NerveCenter class of alerts. This allows you to automate actions or commands for these events. For menus associated with alerts, the menu options can include commands that reference fields in the alert.

For example, to define a class for NerveCenter events, you would first enter in the NerveCenter.rules file the class value, such as `$Class = number`, where *number* is an arbitrary numerical value that you assign. For this example, we'll use the number 7500. Afterward, from the Configuration Manager, you define a conversion for the class and then define the class itself and link it with a menu.

Figure 4-19 shows the dialogs used to define a conversion and a class.

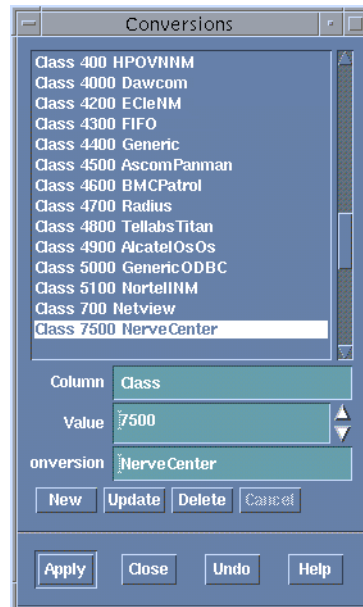


FIGURE 4-19. Defining a Conversion and a Class

After defining a class, you can create a menu specifically for that class. If there's a particular action, such as a ping that may need to be performed for NerveCenter alerts, you can define the action as a menu item.

Figure 4-20 shows the Menus dialog box used to define menu items for a class.

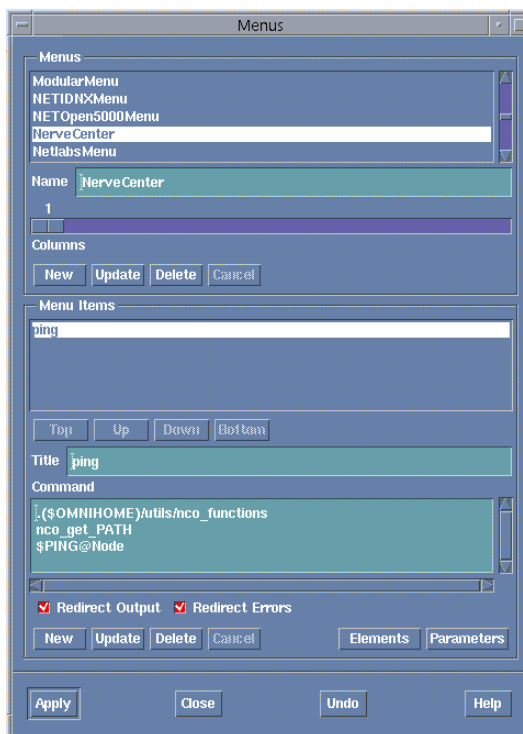


FIGURE 4-20. Associating Menu Items with the Class

When administrators receive NerveCenter informs tagged with the class you create, they can select the menu items defined for the class from the Tools menu of any window in which alerts are displayed.

Automated Actions

The Automation Builder allows you to automate actions or commands for certain types of events. For example, you can notify an administrator of critical events after a specific period of time has elapsed. You do this by creating triggers to detect particular states, for example, an alert severity value of 4, and actions that define the responses to those states. The triggers and states are stored in the Object Server and are created using Object Server SQL.

Figure 4-21 shows the Automation Builder dialog box used to create and associate actions. You can customize the predefined actions listed in the box or create new actions.

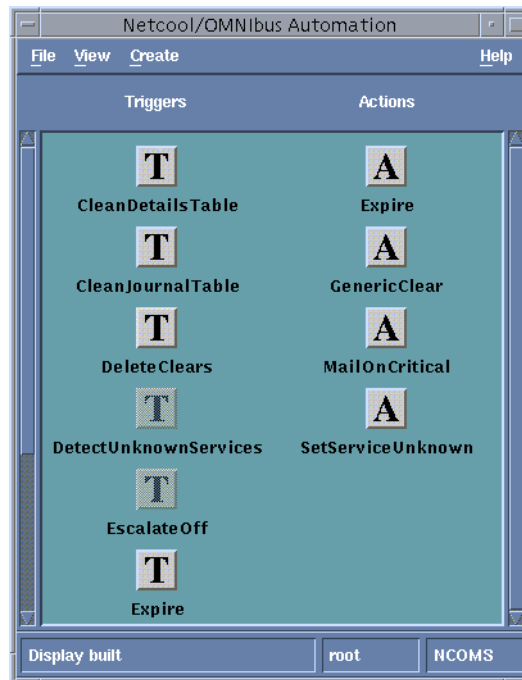


FIGURE 4-21. Automation Builder

NerveCenter also provides automated actions that can be performed when an alarm transitions. Besides the inform action described in this paper, NerveCenter has 21 other actions designed for notification, logging data, or correcting a condition.

Objective View Map

With the Objective View Map Editor, you can map services and devices to geographical regions. This gives you a service-level view of managed objects.

The Objective View displays map books, each containing a number of map pages with graphical objects called symbols. Each page of a book might include symbols representing management sites for a different region of a country or sites in different countries. From these symbols, you can display the status of your services and devices for the region or country.

When creating map pages, layers make it easy to create background-layer images, second-layer annotations, and object-layer editable symbols that can be dynamically manipulated to show status and associations.

Figure 4-22 shows a U.S. map in the background layer and a symbol in the object layer. This particular symbol was associated with the NerveCenter class in the Classes dialog box; double-clicking it displays a dialog in which you can edit the symbol appearance and define associations.

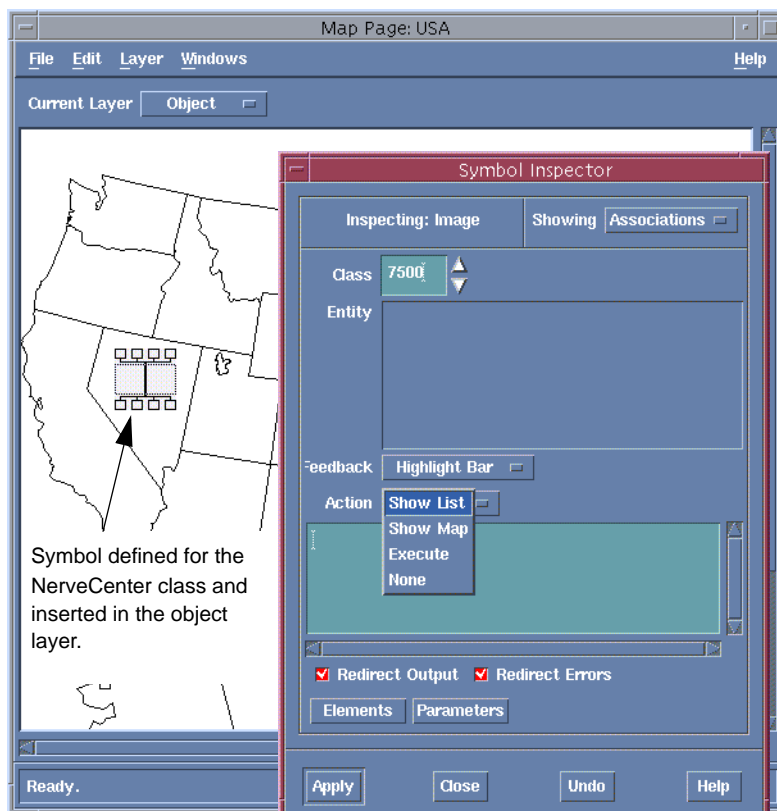


FIGURE 4-22. Map Page Symbol Associated with a Class and an Action

The action that you associate with a symbol determines what happens when you or someone else double-clicks the icon in the Objective View map. For example, if you select the Show List option, double-clicking the icon in the Objective View starts the Event List with the filter and view of the associated entity.

The class you associate with a symbol determines the appropriate Tools menu for the symbol when viewed in the Objective View.

Netcool Integration Reference

The following section includes information you may need to reference occasionally when integrating NerveCenter and IBM Tivoli Netcool/OMNIBus.

This section includes:

- ◆ [Command line reference for Integrating NerveCenter with Netcool on page 92](#)
- ◆ [Variable Bindings for NerveCenter Informs on page 94](#)
- ◆ [Inform Data Sent from NerveCenter on page 95](#)
- ◆ [Debug Probe Output on page 97](#)
- ◆ [Sample NerveCenter.rules File on page 98](#)

Command line reference for Integrating NerveCenter with Netcool

During a typical NerveCenter installation, the NerveCenter Universal Platform Adapter is installed as a service or daemon. However, there may be times in which you will want to make changes from the command line. From a command prompt type the following:

```
paserver [-n ON|OFF] [-d] [-g] [-h] [-nhost] [-nport] [-o] [-p] [-r]
[-scm a|m|r|s] [-tcfg] [-v] [-?]
```

TABLE 4-5. Command Line Switches for Integrating NerveCenter with IBM Tivoli Netcool/OMNIBus

Switch	Description
-n ON OFF	Enables or disables NerveCenter integration with Netcool. On Windows, when starting paserver from the command line, you must specify either -d or -scm option in combination with either -n or -u .
-d	Runs the Universal Platform Adapter from the command line in debug mode and outputs debug messages to the console. The next time the host machine boots, the Universal Platform Adapter will run as a service or daemon again.
-g	(Windows only) Registers the Universal Platform Adapter as an Event Source.
-h	Displays help information for the Universal Platform Adapter switches.
-nhost	Defines the machine on which the Netcool probe is located. The default is localhost.
-nport	Defines the port the NerveCenter platform adapter uses to communicate with IBM Tivoli Netcool/OMNIBus. The default is 32510.

TABLE 4-5. Command Line Switches for Integrating NerveCenter with IBM Tivoli Netcool/OMNIbus (Continued)

Switch	Description
-o	(Windows only) Records values into Registry. Any options (other than -scm) become a part of the standard configuration. To use this switch, you should first stop the Universal Platform Adapter. You must then restart the Universal Platform Adapter.
-p	Defines the platform adapter's listening port. The default is 32509. Note: This number must match the Inform Recipient port in NerveCenter Administrator.
-r	Removes the Universal Platform Adapter as a service. It also removes Registry entries created at install time.
[-scm a m r s]	(Windows only) Changes settings in the service control manager: a Installs the Universal Platform Adapter as a service, making it as autostart. The service will start following this command. m Installs the Universal Platform Adapter as a service, making it start on demand. The service will not start following this command. r Removes the Universal Platform Adapter as a service. If the service is running, it will be stopped. s Starts the Universal Platform Adapter as a service. This may be combined with a or m.
-tcfg	Defines the full qualified path/filename for the Event Adapter configuration file. The default is /opt/OSInc/userfiles/nctec.cfg
-v	Views current Universal Platform Adapter settings.
-?	Displays help information for the Universal Platform Adapter switches.

**NOTE**

paserver has many options specific to other integrations. For a complete list of options, see *paserver in Managing NerveCenter*.

Variable Bindings for NerveCenter Informs

Depending on how its behavior models are designed, a NerveCenter detecting particular network conditions can send Inform packets to IBM Tivoli Netcool/OMNIBus. Although these Inform packets use TCP/IP, they are similar in content to an SNMP trap, containing trap numbers (generic and specific), an enterprise OID, and a variable-binding list. The lengthy varbinds contains information about the alarm that performed the Inform action, such as the name of alarm, the object the alarm was monitoring, and the names of the origin and destination alarm states.

The Netcool receiving the trap can make use of the information in the variable bindings much the same way it would use variable bindings found in an SNMP trap.

Table 4-6 explains the contents of this variable-binding list.

TABLE 4-6. Inform Trap Variable Bindings

Binding	Value
0	The name of the domain where NerveCenter is running
1	The name of the host machine running the NerveCenter Server
2	The name of the managed node associated with the alarm
3	The base object associated with the alarm (for example, ifEntry for a monitored interface)
4	The base object instance associated with the alarm (for example, 4 for the fourth interface)
5	The name of the subobject. This would include the null string if the alarm is not associated with an alarm.
6	The property group assigned to the node or the subobject
7	The name of the alarm
8	The alarm's property
9	The name of the trigger that caused the alarm transition
10	The state of the alarm before the transition
11	The severity of the state of the alarm prior to the transition
12	The state of the alarm after the transition
13	The severity of the state of the alarm after the transition
14	The maximum severity of all active alarms for the managed node before this alarm transition
15	The maximum severity of all active alarms for the managed node after this alarm transition

TABLE 4-6. Inform Trap Variable Bindings (Continued)

Binding	Value
16	The variable bindings in the poll or trap that caused the transition. These variable bindings are formatted as follows: Attribute ncTransitionVarBinds = attribute.instance=value;attribute=value;...
17	The identification number of the alarm instance

Inform Data Sent from NerveCenter

The Inform Netcool message is the mechanism that NerveCenter uses to send events to Netcool/OMNIBus. The following table describes the data sent with NerveCenter informs. The first three items are contained in the inform header. The remaining items are listed alphabetically.



NOTE

Variables that are included by default in the **Identifier** field of the NerveCenter.rules file are listed as a Default Identifier value..

TABLE 4-7. Inform Data

Variable	Description
\$LDT	Local date timestamp.
\$MessageType	The type of message being sent is Inform Netcool. There are seven possible types of messages: <ul style="list-style-type: none"> ◆ Connection Request ◆ Connection Accepted ◆ Connection Rejected ◆ Exit Notification ◆ Heartbeat Query ◆ Heartbeat Response ◆ Inform Netcool
\$ServerID	The unique identifier for the NerveCenter Server that manages the current alarm transition and alert. The identifier consists of hostname or IP address. This identification becomes important when there are multiple NerveCenter Servers sending informs to the same Universal Platform Adapter.

TABLE 4-7. Inform Data (Continued)

Variable	Description
\$AlrmDN	The alarm definition name for the alarm that transitioned and generated the inform.
\$AlrmProp	The property assigned to the alarm that transitioned and generated the inform.
\$BOI	The base object instance for the interface that triggered the transition and generated the inform. If the base object is associated with an interface and, therefore, is listed as a table in the MIB .ASN1 file, the instance corresponds to a row in that table.
\$BON	The base object name for the SNMP base object that triggered the transition and generated the inform.
\$DSN	Default Identifier value. The destination state name (the name of the state to which the alarm transitioned).
\$DSS	Default Identifier value. The destination state severity (the severity level associated with the state to which the alarm transitioned).
\$IPAddr	Default Identifier value. The IP address of the node that caused the event. If the node has more than one IP address, the number provided denotes the IP address associated with the event.
\$MesgID	Default Identifier value. The specific number that you enter into the NerveCenter Client alarm definition when you define the Netcool/OMNIBus inform action. If no number is entered, the message ID defaults to the value 1000 for any transition whose destination state has a severity less than 9 (Warning). For severity levels of 9 or greater, the \$MesgID defaults to the value 1000 plus the destination state's severity level.
\$NodeName	Default Identifier value. The name of the node that caused the event to be sent. The name consists of hostname or IP address.
\$NPG	The node property group assigned to the node associated with the inform.
\$OSN	Default Identifier value. The originating state name (the name of the state from which the alarm transitioned).
\$OSS	Default Identifier value. The originating state severity (the severity level associated with the state from which the alarm transitioned).
\$ROCom	The read only community string of the node associated with the inform.
\$RWCom	The read-write community string of the node associated with the inform.
\$TrapEID	Default Identifier value. The trap's enterprise ID. If the transition was caused by a trigger fired from an SNMP trap, the enterprise ID is included here. If the transition was not from a trap, this variable is empty.

TABLE 4-7. Inform Data (Continued)

Variable	Description
\$TrapGN	Default Identifier value. The trap's generic number. If the transition was caused by a trigger fired from an SNMP trap, the generic trap number is included here. If the transition was not from a trap, this variable defaults to -2.
\$TrapSN	Default Identifier value. The trap's specific number. If the transition was caused by a trigger fired from an SNMP trap, the specific trap number is included here. If the transition was not from a trap, this variable defaults to -2.
\$TrigName	The name of the trigger that caused the state transition.
\$VarBinds (n)	Variable binding pair for the nth variable binding, in text format: attribute = value

Debug Probe Output

By running the probe in debug mode, you can confirm that the probe is receiving the correct data from NerveCenter. This helps you establish that:

- ◆ NerveCenter is communicating with the probe—it's especially important to determine this when NerveCenter and Netcool components are installed on different machines.
- ◆ The probe is receiving the correct information from NerveCenter—this helps identify whether you have the correct versions of both the probe and NerveCenter.

The command for running the probe in debug mode is:

```
$OMNIHOME/probes/nco_p_NerveCenter -messagelevel debug &
```

This command forces Netcool/OMNIBus to log the parsed values received from the probe to a file named *NerveCenter.log*, located in the *\$OMNIHOME* directory.

The NerveCenter inform variables are logged each time NerveCenter issues an inform to the probe. Each log entry is appended to the *NerveCenter.log* file. The log file lists each variable received from NerveCenter along with the variable's current value. You can compare the variables and values against the NerveCenter inform data for each of your inform actions.

Sample NerveCenter.rules File

The NerveCenter.rules file is the starting point for customizing the Netcool/OMNIBus desktop, eliminating duplicate alerts in the Event List, and associating events with fields that can be manipulated by the Object Server. This section contains a customized NerveCenter.rules file, modified to do the following:

- ◆ Define lookup tables for adding information to an event. The tables for this example are located in \$OMNIHOME/probes/platform and are accessed using the *lookup* keyword.
- ◆ Create generic variable token definitions that can be assigned to database fields.
- ◆ Assign default values to database fields to ensure all fields contain a value. The rules file later passes these fields to the Object Server when the probe receives certain types of events.
- ◆ Create several different types of classes, along with associated token variables. The class value is set depending on the type of event. This field is also used to associate tools and actions in the Netcool desktop tool.
- ◆ Parse the node names, which follow a predefined set of naming conventions, and extract certain values that can be expanded into more readable text.
- ◆ Define separate cases for events and make field assignments for each case. Each case is based on an incoming \$MsgID value, which the user provides when creating the inform action in NerveCenter.
- ◆ Modify the Identifier field and add a value for the AlertKey field. The AlertKey field forwards to the Object Server the name of the alarm, node, and interface if applicable for an inform.

If you intend to develop your own NerveCenter.rules file, first map out the message for a single NerveCenter alarm and then create the case statement for that \$MsgID value. This makes it easy to format the @Summary field to include the information you want displayed. The rules file can grow quite large; working with one alarm case at a time makes customization easier.

CAUTION

Back up of the default NerveCenter.rules file before modifying it. Changes you make to the rules file affect the probe's compatibility with incoming NerveCenter inform data. You should know the Netcool and NerveCenter products thoroughly and test each change to the rules file before proceeding further changes.

When making changes to a rules file, you must follow the established Netcool syntax. If the syntax in a rules file is incorrect, the probe cannot be started. Netcool/OMNIBus includes a syntax probe that you can use to test the syntax of a rules file.


```
#####
# Copyright (C) 1998 Omnibus Transport Technologies Ltd.
# All Rights Reserved
# RESTRICTED RIGHTS:
# This file may have been supplied under a license.
# It may be used, disclosed, and/or copied only as permitted
# under such license agreement. Any copy must contain the
# above copyright notice and this restricted rights notice.
# Use, copying, and/or disclosure of the file is strictly
# prohibited unless otherwise provided in the license agreement.
# Ident: $Id: NerveCenter.rules 1.1 1998/07/07 09:23:42 nic Development $
#####
table interfaces= "/opt/Omnibus/probes/hpux10/Tables/interfaces.lookup"
    default="NoMatch"
table fr= "/opt/Omnibus/probes/hpux10/Tables/fr.lookup"
    default="NoMatch"
table bgp= "/opt/Omnibus/probes/hpux10/Tables/bgp.lookup"
    default="NoMatch"
table junction= "/opt/Omnibus/probes/hpux10/Tables/junction.lookup"
    default="NoMatch"
table abbreviations =
"/opt/Omnibus/probes/hpux10/Tables/abbreviations.lookup"
    default="9999"
table supportdefinitions =
"/opt/Omnibus/probes/hpux10/Tables/supportdefinitions.lookup"
    default="0"
table exceptionscity =
"/opt/Omnibus/probes/hpux10/Tables/exceptionscity.lookup"
    default="Unknown"
table exceptionsdevfun =
"/opt/Omnibus/probes/hpux10/Tables/exceptionsdevfun.lookup"
    default="Unknown"
#####
# Other Generic Definitions
$DISPLAYON          = 1
$DISPLAYOFF         = 0
$DONOTFORWARDEVENT = 0
$FORWARDEVENT       = 1
$ACKFORWARDEVENT    = 2
$UNDEFINED_SERVICE  = 999
$UNDEFINED_CLASS    = 9999
```

```

$NULL                = 0
$UNKNOWN             = "Unknown"
$UNDEFINED           = "Undefined"
$CUSTOMER            = "xxx"
$STARTEVENT         = 1
$ENDEVENT           = 0
#
# Put ProbeWatch Specific messages here, ie to customise Agent names
# !! This is not part of management system event processing
#
if( match( @Manager, "ProbeWatch" ) )
{
    switch(@Summary)
    {
        case "Running ...":
            @AlertGroup = "probestat"
            @Type       = 2
            @Rise       = $STARTEVENT
        case "Going Down ...":
            @AlertGroup = "probestat"
            @Type       = 1
            @Rise       = $ENDEVENT
        default:
    }
    @AlertKey = @Agent
    @Summary  = @Agent + " probe on " + @Node + ": " + @Summary
} else
{
    switch($MessageType)
    {
        case "Connection Accepted":
            @Identifier = $MessageType + $ServerID
            @Summary    = "Connection accepted from " + $ServerID
            @Severity   = 0

        case "Connection Rejected":
            @Identifier = $MessageType + $ServerID
            @Summary    = "Connection rejected from " + $ServerID
            @Severity   = 3

        case "Exit Notification":
    }
}

```

```

@Identifier = $MessageType + $ServerID
@Summary = "Exit notification received from " + $ServerID
@Severity = 1

case "Inform Netcool":
    @Agent      = "NerveCenter-site"
    @Node       = $NodeName
    @NodeAlias  = $IPAddr
    @Summary    = "?: " + $MesgID
    @Manager    = "manager1"
    @Severity   = 1
    @Type       = 1

#####
#
# The Informs start here for main rules section (non ProbeWatch alerts).
# !! This IS where management system event processing starts
#
#Default User specific fields
@RemedyFlag      = $DONOTFORWARDEVENT
@ProcessedFlag   = $NULL
@LoggedFlag      = $NULL
@DatabaseFlag    = $DONOTFORWARDEVENT
@AlertClass      = $NULL
@DevFuncCode     = $UNKNOWN
@CustomerCode    = $UNKNOWN
@CityCode        = $UNKNOWN
@Location        = $UNKNOWN
@Class           = $UNDEFINED_CLASS
@DisplayFlag     = $DISPLAYOFF
@Interface       = $UNDEFINED
@Rise            = $STARTEVENT
#
# SupportClass is a reference indicating the Support for a particular event
#
$UNDEFINED_SUPPORT = "0"
$INTERNAL          = "1"
$REMOTE            = "2"
$BUSINESS          = "3"
$OPERATIONS        = "4"

```

```
@SupportClass = $UNDEFINED_SUPPORT
#
# VendorClass is defined by the division of the Class by the value of 100
# VendorClass table will look like:
#
#      10      Cisco
#      11      Bay
#      99      Other
#
$CISCO       = "10"
$BAY        = "11"
$OTHER      = "99"

@VendorClass = $OTHER
#
# AlertClass Definitions
$LINKFAILURE      = "LinkFailure"
$ROUTINGFAILURE   = "RoutingFailure"
$COMPONENTFAILURE = "CompentFailure"
$ENVFAILURE       = "EnvironmentalFailure"
$NODEFAILURE      = "NodeFailure"
$PERFORMANCE      = "Performance"
$OTHERCLASS       = "Other"

# EventType Definitions
$TRAFFICDROP      = "TrafficDrop"
$TRAFFICLOAD      = "TrafficLoad"
$PACKETLOSS       = "PacketLoss"
$BGPFAILURE       = "BGPFailure"
$SERVICEFAILURE  = "ServiceFailure"
$MEMORYFAILURE    = "MemoryFailure"
$LINKDOWN         = "LinkDown"
$NODEDOWN         = "NodeDown"
$SECURITY         = "Security"
$CPUFAILURE       = "CPUFailure"
$LINKERROR        = "LinkError"
$SNMPFAILURE      = "SnmPFailure"

# If the Node Name is NOT an IP address - parse out info from the name
#
```

```

if (regmatch(@Node, "^[a-zA-Z][a-zA-Z][a-zA-Z][0-9][0-9][a-zA-Z][a-zA-Z][a-
zA-Z]")) {
    @DevFuncCode      = extract(@Node, "([a-zA-Z][a-zA-Z][a-zA-Z]+)")
    $junction         = lookup(@Node, junction)
    if (match($junction, "NoMatch")) {
        @SupportClass = lookup (@DevFuncCode, supportdefinitions)
    } else {
        @SupportClass = lookup ($junction, supportdefinitions)
    }
    @Class             = lookup (@DevFuncCode, abbreviations)
    @CustomerCode      = "abc"
    @CityCode          = extract(@Node, ".*([a-zA-Z][a-zA-Z][a-zA-Z])")
    @VendorClass       = int(@Class)/100
    @Location          = $RWCom
    @DisplayFlag       = $DISPLAYON
} else

#
if (regmatch(@Node, "^[a-zA-Z][a-zA-Z][0-9][0-9][a-zA-Z][a-zA-Z][a-zA-Z]"))
{
    @DevFuncCode      = extract(@Node, "([a-zA-Z][a-zA-Z]+)")
    @CustomerCode      = "abc"
    @CityCode          = extract(@Node, ".*([a-zA-Z][a-zA-Z][a-zA-Z])")
    @Class             = lookup (@DevFuncCode, abbreviations)
    @VendorClass       = int(@Class)/100
    @SupportClass      = lookup (@DevFuncCode, supportdefinitions)
    @Location          = $RWCom
    @DisplayFlag       = $DISPLAYON
} else

#
if (regmatch(@Node, "^[a-zA-Z][a-zA-Z][a-zA-Z][0-9][a-zA-Z][a-zA-Z][a-zA-
Z][a-zA-Z]")) {
    @DevFuncCode      = extract(@Node, "([a-zA-Z][a-zA-Z][a-zA-Z]+)")
    @CustomerCode      = "abc"
    @CityCode          = extract(@Node, ".*([a-zA-Z][a-zA-Z][a-zA-Z])")
    @Class             = lookup (@DevFuncCode, abbreviations)
    @VendorClass       = int(@Class)/100
    @SupportClass      = lookup (@DevFuncCode, supportdefinitions)
    @Location          = $RWCom
    @DisplayFlag       = $DISPLAYON
}

```

```

} else

#
if (regmatch(@Node, "^yes[0-9][0-9][0-9]i[0-9][0-9][a-zA-Z][a-zA-Z][a-zA-Z]")) {
    @DevFuncCode      = extract(@Node, "([a-zA-Z][a-zA-Z][a-zA-Z]+)")
    @CustomerCode     = "abc"
    @CityCode         = extract(@Node, ".*([a-zA-Z][a-zA-Z][a-zA-Z])")
    @Class             = lookup (@DevFuncCode, abbreviations)
    @VendorClass      = int(@Class)/100
    @SupportClass     = lookup (@DevFuncCode, supportdefinitions)
    @Location         = $RWCom
    @DisplayFlag      = $DISPLAYON
} else

{
#
# Pick up the odd nodes
    @DevFuncCode      = lookup(@Node, exceptionsdevfun)
    $junction         = lookup(@Node, junction)
    if (match($junction, "NoMatch")) {
        @SupportClass = lookup (@DevFuncCode, supportdefinitions)
    } else {
        @SupportClass = lookup ($junction, supportdefinitions)
    }
    @CustomerCode     = "abc"
    @CityCode         = lookup(@Node, exceptionscity)
    @Class             = lookup (@DevFuncCode, abbreviations)
    @VendorClass      = int(@Class)/100
    @Location         = $RWCom
    @DisplayFlag      = $DISPLAYON
}

#
# Check to see if @Class was not set (Normally because not found in any
lookup
if (int(@Class) == 0) {
    @Class             = $UNDEFINED_CLASS
    @VendorClass      = $OTHER
}

```

```

# Next Enterprise: NetLabs_NerveCenter .1.3.6.1.4.1.78
switch($MesgID)
{
case "3004":
    # NC_alarm1
    @AlertKey      = $AlrmDN + $NodeName
    @AlertGroup    = $AlrmDN + "FreeBusy"
    @Summary       = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
CPU Utilization back to normal"
    @Severity      = "2"
    @AlertClass    = $COMPONENTFAILURE
    @EventType     = $CPUFAILURE
    @Rise          = $ENDEVENT
case "3007":
    # NC_alarm2
    details($VarBind1)
    @AlertKey      = $AlrmDN + $NodeName
    @AlertGroup    = $AlrmDN + "FreeBusy"
    @Summary       = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
CPU Utilization >= 75% <= 90%"
    @Severity      = "3"
    @AlertClass    = $COMPONENTFAILURE
    @EventType     = $CPUFAILURE
    @Rise          = $STARTEVENT
case "100000":
    # NC_alarm3
    @AlertKey      = $AlrmDN + $NodeName
    @AlertGroup    = $AlrmDN + "UpDown"
    @Summary       = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
unreachable."
    @Severity      = 4
    @AlertClass    = $NODEFAILURE
    @EventType     = $NODEDOWN
    @Rise          = $STARTEVENT
case "100001":
    # NC_alarm4
    @AlertKey      = $AlrmDN + $NodeName
    @AlertGroup    = $AlrmDN + "UpDown"
    @Summary       = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
unreachable. Problem with network path to node."
    @Severity      = 4

```

```

        @AlertClass      = $NODEFAILURE
        @EventType       = $NODEDOWN
        @Rise            = $STARTEVENT
    case "100003":
        # NC_alarm5
        @AlertKey        = $AlrmDN + $NodeName
        @AlertGroup     = $AlrmDN + "UpDown"
        @Summary        = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
Down."
        @Severity       = 5
        @AlertClass     = $NODEFAILURE
        @EventType      = $NODEDOWN
        @Rise          = $STARTEVENT
    case "100004":
        # NC_alarm6
        @AlertKey        = $AlrmDN + $NodeName + $INTERFACE
        @AlertGroup     = $AlrmDN
        @Summary        = "NC: " + $AlrmDN + ": High Error Rate (>5%)
on NODE: " + $NodeName + " interface " + $INTERFACE + "."
        @Severity       = 4
        @AlertClass     = $PERFORMANCE
        @EventType      = $LINKERROR
        @Rise          = $STARTEVENT
    case "100008":
        # NC_alarm7
        details($VarBind1)
        @AlertKey        = $AlrmDN + $NodeName + $INTERFACE
        @AlertGroup     = $AlrmDN + "UpDown"
        $LookupKey      = $NodeName + $INTERFACE
        $SpecialInfo    = lookup ($LookupKey, interfaces)
        if(match($SpecialInfo,"NoMatch")) {
            $Exclamation = ""
            $SpecialInfo = ""
            @Severity     = 4
        }
        else {
            $Exclamation = "Hot!! "
            @Severity     = 5
        }
        @Summary        = $Exclamation + "NC: " + $AlrmDN + ": Node
" + $NodeName + " Interface: " + $INTERFACE + " is Down " + $SpecialInfo
        @Interface      = $INTERFACE

```



```

        @AlertClass      = $LINKFAILURE
        @EventType       = $LINKDOWN
        @Rise            = $STARTEVENT
case "100011":
    # NC_alarm8
    details($VarBind1)
        @AlertKey        = $AlrmDN + $NodeName + $INTERFACE
        @AlertGroup      = $AlrmDN + "UpDown"
    $LookupKey          = $NodeName + $INTERFACE
    $SpecialInfo         = lookup ($LookupKey, interfaces)
    if(match($SpecialInfo,"NoMatch")) {
        $Exclamation     = ""
        $SpecialInfo     = ""
        @Severity        = 4
    }else {
        $Exclamation     = "Hot!! "
        @Severity        = 5
    }
        @Summary         = $Exclamation + "NC: " + $AlrmDN + ": Node
" + $NodeName + "Interface: " + $INTERFACE + " is Flapping " + $SpecialInfo
        @Interface       = $INTERFACE
        @AlertClass      = $LINKFAILURE
        @EventType       = $LINKDOWN
        @Rise            = $STARTEVENT
case "100016":
    # NC_alarm9
        @AlertKey        = $AlrmDN + $NodeName + $INTERFACE
        @AlertGroup      = $AlrmDN + "UpDown"
        @Summary         = "NC: " + $AlrmDN + ": Node: " + $NodeName + "
Session: " + $INTERFACE + " is Up."
        @Severity        = 2
        @AlertClass      = $ROUTINGFAILURE
        @EventType       = $BGPFAILURE
        @Rise            = $ENDEVENT
case "100017":
    # NC_alarm10
        @AlertKey        = $AlrmDN + $NodeName
        @AlertGroup      = $AlrmDN + "Reboot"
        @Summary         = "NC: " + $AlrmDN + ": NODE: " + $NodeName + "
has rebooted."
        @Severity        = 3

```

```

        @AlertClass      = $NODEFAILURE
        @EventType       = $NODEDOWN
        @Rise            = $STARTEVENT
    case "100063":
        # NC_alarm11
        details($*)
        @AlertKey        = $AlrmDN + $NodeName
        @AlertGroup      = $AlrmDN + "LowOK"
        @Summary         = "NC: " + $AlrmDN + ": NODE: " + $NodeName +
"Low Memory! Current " + $VarBind1
        @Severity        = 4
        @AlertClass      = $PERFORMANCE
        @EventType       = $MEMORYFAILURE
        @Rise            = $STARTEVENT
    case "100064":
        # NC_alarm12
        details($*)
        @AlertKey        = $AlrmDN + $NodeName
        @AlertGroup      = $AlrmDN + "LowOK"
        @Summary         = "NC: " + $AlrmDN + ": NODE: " + $NodeName +
"Returned to normal memory utilization. Current memory = " + $VarBind1
        @Severity        = 2
        @AlertClass      = $PERFORMANCE
        @EventType       = $MEMORYFAILURE
        @Rise            = $ENDEVENT
        default:
        @Summary         = "Unknown specific trap number (" + $MsgID +
") received for enterprise " + $enterprise-name
        @Severity        = 1
        # details ($TrapEID, $MsgID, $ServerID, $NodeName, $INTERFACE,
$NPG, $AlrmDN, $AlrmProp)
    }
#####
#
# This is the end of the main rules section
#
#####
#
# The identifier is built here to take into account the AlertKey
#
@Identifier = $ServerID+": "+$NodeName+": "+$MsgID+": "+@AlertKey+": "+@Rise

```

```
@Initial_Severity = @Severity  
  
default:  
}  
  
# details($*)  
}
```

4

Integrating NerveCenter with IBM Tivoli Netcool/OMNibus

Index

I

A

alarm severity colors 43
autodelete of nodes 31

C

capabilities, filtering by 30

D

Discovery behavior model 26
documentation
 conventions 5
 feedback 6
downstream alarm suppression 31

H

HP OpenView Network Node
 Manager
 Integration with NerveCenter 21
 receiving NerveCenter informs 43

I

IBM Tivoli Netcool/OMNIBus 19, 92
Inform alarm action 37
 variable bindings 53, 94
informs
 acknowledging to OVPA 39
 sending to the Universal Platform
 Adapter 70
 supported platforms 11
IP address, filtering by 31

N

Ncapp 13
NerveCenter Trap service 15
network management platform 26, 30
 as a node source 29

 resynchronization 29
 setting up downstream alarm
 suppression 31
 synchronization with
 network management platform
 integration 19
 IBM Tivoli Netcool/OMNIBus 19
 OpenView Network Node Manager
 integration 13
 platform adapter 12
node
 renaming 30
node data
 source 26
 Supported platforms 11
node list
 filtering by capabilities 30
 filtering by IP address 31
 filtering by system object
 identifier 31
 populating using a network
 management platform 27
Node Source tab
 resync parent rate 32

O

online knowledgebase 7
OpenView 13
OpenView Network Node Manager
 13
 Ncapp 13
 NerveCenter Trap service 15
OpenView Platform Adapter 22
 changing settings 49
 enabling 22, 23
OpenView Platform Adapter
 (OVPA) 27
OVPA
 command line switches 49

integration 13
resync parent rate 32
setting destination of informs 34

P

parent-child information 31
paserver 19
platform adapter 12

R

Resync 29
resync parent rate 32

S

system Object Identifiers 27
 filtering by 31

T

technical support 6
 contacting 7
 educational services 7
 professional services 6
traps
 Inform variable bindings 53, 94
 NerveCenter Trap service 15

U

Universal Platform Adapter
 integration 19
 starting and stopping 74
UNIX
 Universal Platform Adapter 74

V

variable bindings 53, 94
 Inform traps

