

MetaConsole for OpenView: Getting Started

version 2.3



Copyright © 2000-2004 NETAPHOR SOFTWARE, INC.

All rights reserved. No part of the contents of this document may be reproduced or transmitted in any form or by any means without the written permission of NETAPHOR SOFTWARE, INC.

MetaConsole and NETAPHOR are trademarks of NETAPHOR SOFTWARE, INC.

HP OpenView is a trademark of the Hewlett-Packard Company.

Microsoft and Windows are registered trademarks of Microsoft Corporation.

Other company or product names may be trademarks or registered trademarks of their respective owners.

MetaConsole SOFTWARE LICENSE AGREEMENT

1. IMPORTANT - READ CAREFULLY:

This NETAPHOR SOFTWARE, INC. End User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity - "LICENSEE") and NETAPHOR SOFTWARE, INC. for the MetaConsole™ software product identified above, which includes computer software and associated media and printed materials, and may include "online" or electronic documentation ("SOFTWARE PRODUCT" or "SOFTWARE"). BY OPENING THIS PACKAGE, DOWNLOADING OR INSTALLING SOFTWARE TO YOUR COMPUTER YOU AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE, DO NOT OPEN THIS PACKAGE AND PROMPTLY RETURN IT UNOPENED TO THE PLACE WHERE YOU OBTAINED IT FOR A FULL REFUND (subject to shipping and handling charges). If you do not agree to the terms of this EULA, you may not install, copy, or use the SOFTWARE PRODUCT.

The SOFTWARE PRODUCT is copyrighted and licensed (not sold) to you by NETAPHOR SOFTWARE, INC. ("NETAPHOR" or "LICENSER"), 15520 Rockfield Blvd., Suite E, Irvine, CA. This license agreement represents the entire agreement concerning the SOFTWARE between LICENSEE and NETAPHOR and it supersedes any prior proposal, representation, or understanding between the parties. NETAPHOR reserves any rights not expressly granted to LICENSEE.

2. **SOFTWARE PRODUCT LICENSE.** The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed to the purchasing entity.

3. **GRANT OF LICENSE.** This EULA grants LICENSEE the following non-exclusive rights:

3.1 **RUN-TIME LICENSE:** LICENSEE receives the right to use the MetaConsole software as described in this EULA for the purposes of managing devices and services supported by MetaConsole.

4. RESTRICTIONS.

4.1 LICENSEE may not ASSIGN, SUBLICENSE, MODIFY, ADAPT, TRANSFER, PLEDGE, LEASE, RENT OR SHARE LICENSEE's RIGHTS UNDER THIS LICENSE AGREEMENT.

4.2 LICENSEE MAY NOT REVERSE ASSEMBLE, REVERSE COMPILE, OR OTHERWISE TRANSLATE THE SOFTWARE OR USE ANY OTHER METHOD TO DISCOVER THE SOFTWARE'S SOURCE CODE. LICENSEE MAY NOT CREATE DERIVATIVE WORKS BASED UPON THE SOFTWARE OR ANY PART THEREOF EXCEPT AS SPECIFICALLY PROVIDED IN THIS LICENSE AGREEMENT. LICENSEE MAY NOT COPY THE LICENSED ARTICLES EXCEPT AS SPECIFICALLY PROVIDED IN THIS LICENSE AGREEMENT.

5. **LICENSER'S RIGHTS.** LICENSEE acknowledges and agrees that the Licensed Articles are proprietary products of LICENSER or its suppliers and are protected under U.S. copyright law. LICENSEE further acknowledges and agrees that all right, title and interest in and to the Licensed Articles, including associated intellectual property rights, are and shall remain with LICENSER or its supplier. LICENSER'S suppliers may protect their rights in the Licensed Articles in the event of a violation of this License Agreement. This License Agreement does not convey to LICENSEE an interest in or to the Licensed Articles, but only a limited right of use revocable in accordance with the terms of this License Agreement.

6. **LICENSE FEES.** The license fees paid by LICENSEE are paid in consideration of the licenses granted under this License Agreement.

7. **TERMINATION.** This License Agreement is effective upon the purchase and acceptance of the SOFTWARE PRODUCT by LICENSEE or the downloading of the SOFTWARE PRODUCT from NETAPHOR's Web site or other authorized electronic medium and shall continue until terminated. LICENSER may terminate this License Agreement upon the breach by LICENSEE of any term hereof. Upon such termination by LICENSER, LICENSEE agrees to delete the SOFTWARE PRODUCT from the hard drive of the computers that it is installed on and destroy all copies of the SOFTWARE PRODUCT.
8. **DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS:**
 - 8.1 Rental. LICENSEE may not rent, lease or sell the SOFTWARE PRODUCT as a standalone component.
 - 8.2 Support Services. NETAPHOR SOFTWARE, INC. may provide LICENSEE with support services related to the SOFTWARE PRODUCT ("Support Services") described in "online" documentation and/or in other NETAPHOR SOFTWARE, INC.-provided materials.
9. **NO WARRANTIES.** To the maximum extent permitted by applicable law, NETAPHOR expressly disclaims any warranty for the SOFTWARE PRODUCT. The SOFTWARE PRODUCT and any related documentation are provided "as is" without warranty of any kind, either express or implied, including, without limitation, the implied warranties of merchantability or fitness for a particular purpose. The entire risk arising out of use or performance of the SOFTWARE PRODUCT remains with LICENSEE. LICENSEE may, however, enter into a separate support agreement with the LICENSER to obtain ongoing support for the SOFTWARE PRODUCT.
10. **LIMITATION OF LIABILITY.** IN NO EVENT SHALL NETAPHOR OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL, PUNITIVE OR MULTIPLE DAMAGES WHATSOEVER, WHETHER IN CONTRACT, TORT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT OR ACCOMPANYING PRINTED MATERIALS OR DOCUMENTS OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF NETAPHOR AND ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR IF THE DAMAGES ARE FORSEEABLE. IN ANY CASE, NETAPHOR'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS EULA SHALL BE LIMITED TO THE GREATER OF THE AMOUNT ACTUALLY PAID BY LICENSEE FOR THE SOFTWARE PRODUCT OR U.S. \$5.00; LICENSEE ACKNOWLEDGES THAT THE LICENSE FEES REFLECT THIS ALLOCATION OF RISK. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO LICENSEE.
11. **GOVERNING LAW.** This License Agreement shall be construed and governed in accordance with the laws of the State of California. Should any term of this License Agreement be declared void or unenforceable by any court of competent jurisdiction, such declaration shall have no effect on the remaining terms hereof.
12. **COSTS OF LITIGATION.** If any action is brought by either party to this License Agreement against the other party regarding the subject matter hereof, the prevailing party shall be entitled to recover, in addition to any other relief granted, reasonable attorney fees and expenses of litigation.
13. **NO WAIVER.** The failure of any party to enforce any rights granted hereunder or to take action against the other party in the event of any breach hereunder shall not be deemed a waiver by that party as to subsequent enforcement of rights or subsequent actions in the event of future breaches.
14. **U.S. GOVERNMENT RESTRICTED RIGHTS.** The SOFTWARE PRODUCT and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software - Restricted Rights at 48 CFR 52.227-19, as applicable. The Manufacturer is NETAPHOR SOFTWARE, INC., Irvine, CA 92618.

YEAR 2000 STATEMENT

NETAPHOR SOFTWARE, INC. considers a product Year 2000 compliant if the product, when used in accordance with its associated documentation, is capable of correctly processing, providing, and/or receiving date data within and between the 20th and 21st centuries, provided that all products (for example, hardware, software and firmware) used with the product properly exchange accurate date data with it.

Table of Contents

Overview	1
MetaConsole Documentation	1
Chapter 1. MetaConsole Architecture.....	2
Client.....	4
Server	4
Service Providers	5
Chapter 2. Planning Your MetaConsole Deployment	6
Using Multiple MetaConsole Servers.....	6
Are you managing devices of multiple service providers?	7
Are you managing a large number of devices?.....	7
Do devices of the same service provider require different settings?	7
Are you running MetaConsole across a firewall?.....	7
Running MetaConsole as a Service	7
Monitoring the Network and Using Alarms.....	7
Alarms in OpenView.....	8
Alarms in MetaConsole	8
Choosing an Alarms Database	9
Chapter 3. Installation	10
Prerequisites	10
Server	10
Client	10
Database	11
Placement of Components	11
Installing MetaConsole Components	11
Starting the MetaConsole server after installation	13
Uninstalling MetaConsole Components.....	13
Installing MSSQL Database	14
Installing MySQL Database	14
Chapter 4. Managing Devices.....	16
Refreshing Displayed Information.....	16
Chapter 5. Configuring MetaConsole	17
Refreshing Displayed Information.....	18
MetaConsole client.....	18

Configuring the Server List	18
Specifying Alarm Notification Methods	19
Getting MetaConsole Component Version Information	21
Enabling and Disabling Service Providers	22
Viewing Alarms	22
Chapter 6. Configuration Text Files	24
Configuration.txt	24
Example file	24
Keywords	26
OVImp.properties	27
Example file	27
Keywords	27
Chapter 7. Frequently Asked Questions	28

Overview

A *management console* is a single tool used to manage entire networks. It is a framework within which device- or service-specific management modules can share a user interface, alarm monitoring, and other basic functions. Popular management consoles include Microsoft Management Console (MMC), HP OpenView, Tivoli Enterprise, and CA Unicenter.

MetaConsole technology provides a solution that works with multiple management consoles and multiple network protocols on multiple platforms.

This guide describes the architecture of MetaConsole and the installation of MetaConsole in an HP OpenView Network Node Manager (NNM) environment.

MetaConsole Documentation

For any type of device you want to manage using MetaConsole, there are two relevant documents in the suite of MetaConsole documentation:

- A *getting started* guide describing MetaConsole's components and explaining the requirements, installation steps, and other details specific to the management environment
- A *client user's guide* describing the MetaConsole client's management of a particular type of device

You are reading the getting started guide for MetaConsole in an HP OpenView environment.

- Chapter 1, a high-level introduction to the MetaConsole architecture, is useful if you desire a basic understanding of what's happening "under the hood," but you can use MetaConsole without reading this chapter.
- Chapter 2 helps you plan your MetaConsole installation to meet your specific needs; it includes important information about mixing MetaConsole versions.
- Chapter 3 provides instructions for installing and starting MetaConsole for OpenView.
- Chapter 4 provides introductory information about the MetaConsole client for OpenView
- Chapter 5 describes the use of a web browser to change MetaConsole server configuration options.
- Chapter 6 covers additional MetaConsole configuration that you perform through text files.
- Chapter 7 provides information to help you address common situations with your MetaConsole configuration.

Chapter 1. MetaConsole Architecture

The MetaConsole application architecture is three-tiered. The major components are:

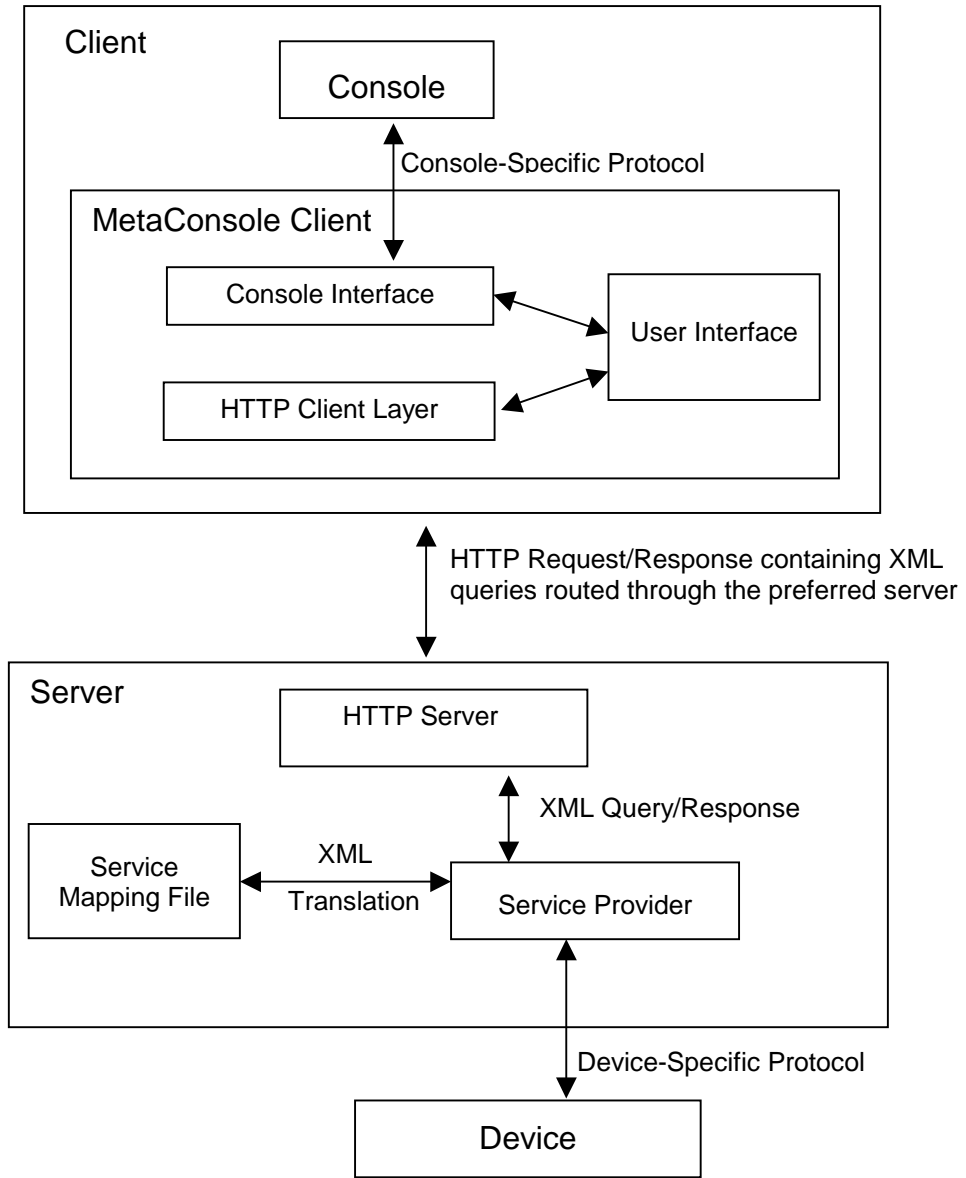
- MetaConsole client —this is the front end that integrates with the console
There is one client per console, managing all supported devices.
- MetaConsole server — this is the back-end component that communicates with MetaConsole clients and with the devices to be managed

A MetaConsole server acts as an agent collecting data for the MetaConsole clients. It can route queries and responses to and from other MetaConsole servers. A server that acts as a proxy in this way is the *preferred server* for the client.

- MetaConsole service providers — these contain the knowledge necessary to communicate with a managed device or service

Typically, a service provider exists for each vendor or class of devices.

The following diagram provides details.



Client

The client component includes three major sub-components:

- The console interface cooperates with the host console so the client functions in a manner native to that console. For example, the console interface in the MetaConsole client for OpenView NNM uses the OpenView API and ensures that the client performs all functions expected of OpenView applications. More specifically, MetaConsole integrates with the OpenView Alarm Browser. The client makes an entry in its own log when monitored events occur, but it also provides OpenView Alarm Browser with the information needed for a standard OpenView error message.
- The HTTP client layer communicates with the MetaConsole server using Hypertext Transfer Protocol (chosen because it is a well-defined protocol that works across firewalls). HTTP requests sent by the MetaConsole client contain queries encoded in XML; the responses returned by the MetaConsole server are also encoded in XML and contain style sheets with instructions for rendering the XML data.
- The user interface component is generally common among MetaConsole clients for different management consoles. This sharing provides the same “look and feel” regardless of whether the user has OpenView, Microsoft Management Console, another console, or a browser. The user interface is rendered by the client using the style sheets provided by the server.

The client might extend the common MetaConsole user interface in console-specific ways to use all the capabilities of the management console. For example, the navigation tree is usually adapted to reflect the native method used by the console to display devices.

Server

The MetaConsole server includes two sub-components:

- The HTTP server controls communication with the MetaConsole client.
- The Service Provider Interface controls the communication with the various service providers.

The MetaConsole server manages multiple service providers and takes care of routing client requests to the correct service providers.

Service Providers

Each MetaConsole server has one or more service providers that can be loaded on that server. Each service provider performs four main functions:

- Translates the HTTP server's XML queries into the protocol understood by the device and translates the device's responses into XML documents.
- Has a service mapping file that defines the translations from XML to the device-specific protocol.
- Discovers devices in the network.
- Polls and listens for alarms from the devices that it is communicating with for the purpose of asynchronous event notification.

Chapter 2. Planning Your MetaConsole Deployment

MetaConsole was designed to be deployed in various and diverse environments. MetaConsole runs well for the small workgroup and can be scaled up, with several MetaConsole servers being deployed in a large enterprise to handle very large networks. Due to this scalable design, it is important that you consider how you would like to deploy MetaConsole, making appropriate choices for your environment.

This chapter also covers considerations that are independent of network size and design but that will help you get the best use of MetaConsole.

Using Multiple MetaConsole Servers

You can install the MetaConsole server on as many computers as you like. A MetaConsole server can service multiple clients. These clients can be of different types (HP OpenView, MMC, and so on).

You can install the MetaConsole client on as many computers as you like. Each client must be able to communicate with at least one MetaConsole server. The client and server can be on the same computer. Each client must designate a specific MetaConsole server as the preferred server.

Given these options, it is important that you plan carefully for your MetaConsole deployment.

A MetaConsole client can communicate with multiple MetaConsole servers, each with its own set of service providers and devices. You should consider using multiple MetaConsole servers within a network when:

- Performance is an issue. To improve performance within a large network, multiple MetaConsole servers can be deployed such that each server has a reasonable set of devices to manage. Clients communicate with different servers depending on the devices that they will be managing.
- Logical groups of devices need to be created. This can be achieved by:
 - Providing each MetaConsole server with a different set of service providers to load, such that each such grouping reflects the different functions one might be seeking to segregate (such as printer management and server management).
 - Providing each service provider with a different set of subnet ranges over which to perform discovery. This allows physical segregation of a network. For example, devices on the first floor and devices on the second floor can be managed through different servers.

Segregation by function or by location is useful in large networks.

- Network traffic needs to be reduced. Dividing a network into logical components and assigning each component a MetaConsole server to manage it cuts down on network traffic.

The following will help you in deciding if multiple MetaConsole servers are appropriate for your environment.

Are you managing devices of multiple service providers?

To manage devices of multiple service providers, you might consider dividing the management load among multiple MetaConsole servers. This would also allow you to have multiple administrators with different job functions (such as managing servers and managing printers).

Are you managing a large number of devices?

To manage many devices, you might consider dividing the management load among multiple MetaConsole servers. By decreasing the load on each MetaConsole server, you allow the server to respond more quickly to client requests and improve client performance.

Do devices of the same service provider require different settings?

If different devices belonging to the same service provider require different settings, you can set up two MetaConsole servers, each with a different range for the service provider and with the appropriate alarm events selected.

Are you running MetaConsole across a firewall?

If you are using MetaConsole across a firewall, you might consider setting up a MetaConsole server with an address accessible outside the firewall and configure it to manage only the devices that you want the person accessing this server to be able to manage. As MetaConsole clients use HTTP to access the MetaConsole server, access to the server itself will be available; HTTP access is typically available across a firewall.

Running MetaConsole as a Service

If you are running MetaConsole on a computer running Windows NT or Windows 2000, you might consider running MetaConsole as a service. Services can be started automatically after the operating system boots and do not require a user to log on. This is especially useful when you are running MetaConsole on an unattended computer.

Monitoring the Network and Using Alarms

MetaConsole gives you flexibility in the monitoring of network activity. MetaConsole monitors managed devices for changes in state, warnings and errors. Monitoring is done by polling as well as through asynchronous event notification via trap reception.

Each service provider has a pre-decided set of alarms that it can generate. These alarms correspond to a change in state of one or more polled variables on a device or a trap that may be received from a device. For each service provider, you select the types of events that should trigger notifications when alarms occur, and select the poll rate for alarms. (For details, see the client user's guide for a particular service provider.) You can then view alarms in both OpenView and MetaConsole.

Alarms in OpenView

Configuring MetaConsole alarms in OpenView

When an event of a selected type occurs, alarms are sent in the form of a trap generated by the MetaConsole client. These alarms are received by Network Node Manager and added to the HP OpenView alarm log. The content of these traps is defined in the MetaConsole.Conf file installed with the MetaConsole client.

If you would like to change the format of the message that is written into the OpenView alarm log when a MetaConsole alarm occurs:

Select Options/Event Management

Select the metaConsole enterprise and valueChangedAlert event and choose Edit/Events/Modify

Change the Event Log Message in the Event Message tab

If you would like to change the category under which MetaConsole alarms are grouped in the OpenView alarm log:

Select Options/Event Management

Select the metaConsole enterprise and valueChangedAlert event and choose Edit/Alarm Categories

Add/remove from the category list into which the alarm will be written

Viewing MetaConsole alarms in OpenView

To view the MetaConsole alarms in OpenView, start the HP OpenView Alarm Browser and select the category into which you grouped the MetaConsole alarm (by default, you can always view it under **All Alarms** and **Application Alert Alarms**).

Alarms in MetaConsole

When an event of a selected type occurs, MetaConsole notifies you with a flashing message in the status bar of each MetaConsole client (browser) window launched from an OpenView terminal node. In addition, you can view lists of alarms on the **View Alarms** page within each MetaConsole server's **Configuration** node. (See *Viewing Alarms* on page 22.)

You can use the **Alarm Notification Methods** page at the top level of MetaConsole configuration to select any combination of alert boxes, system beeps, email messages, and pager text messages as additional alarm notification methods. (See *Specifying Alarm Notification Methods* on page 19.) The **Alert Box** option causes a message box to appear when a monitored event is detected. Only one alert box is visible at a time, so additional alarms that occur while an alert box is open will not generate additional alert boxes.

Note: If you are viewing a page that has an object selected for monitoring and a monitored event occurs for that object, one of two things happens. If the page is not editable, it is refreshed automatically. If the page can be edited, a dialog box appears, indicates that an alarm has occurred, and asks whether the page should be refreshed. In the latter case, choose **YES** to update the page immediately or **NO** to ignore the alarm.

Choosing an Alarms Database

MetaConsole ships with the HSQLDB database. This database stores the alarm (event) information. The HSQLDB database is loaded into memory when the MetaConsole server starts and records alarm information in a flat file in the MetaConsole directory.

The database adds alarm information at the rate of approximately 1000 events per megabyte. Acknowledging all events increases the file size to approximately 500 events per megabyte, but the database is optimized each time the MetaConsole server is started. The optimized database stores alarm events at the rate of 2000 events per megabyte.

Performance is based largely upon the amount of RAM on your system. However, serious degradation to server performance can occur once the database file size increases beyond 50 MB. The larger the database is, the longer it takes to load into memory when the MetaConsole server starts. A 10-MB file can take several minutes to load.

You can save the HSQLDB database by copying and storing the database files `MetaConsoleDB.data`, `MetaConsoleDB.properties`, and `MetaConsoleDB.script`. You can create a new database by deleting the old database files and running the `runcreator.exe` application in the MetaConsole directory.

If you prefer a more familiar and robust database, you can download and install one. The MetaConsole server interfaces with Microsoft's SQL database and MySQL database. The Microsoft SQL database must be purchased and licensed but the MySQL database can be downloaded free and used under a General Public License. For details about installation, see *Installing MSSQL Database* or *Installing MySQL Database*, beginning on page 14.

Chapter 3. Installation

This chapter describes the installation of MetaConsole. Both the MetaConsole server and the client for OpenView can be installed on a number of platforms.

The devices that your client will manage need not be present on your network when you install MetaConsole.

Prerequisites

Server

The following table describes the hardware and software requirements for the MetaConsole server.

Platform	Hardware	Software
Windows (NT 4.0 SP6; 2000, 2003, XP)	Pentium 700 MHz 256 MB available RAM 190 MB free hard disk space	
Solaris 8	Sun Ultra 5 512 MB available RAM 190 MB free hard disk space	CDE, XWindows, or OSF Motif
HP-UX 11.x	HP B2000 512 MB available RAM 190 MB free hard disk space	HP VUE, CDE, XWindows, or OSF Motif
Red Hat Linux 7.0, 7.2	Pentium 300 MHz 128 MB available RAM 190 MB free hard disk space	Gnome

Client

The following table describes the hardware and software requirements for the MetaConsole client.

Platform	Hardware	Software
Windows (2000, XP)	Pentium 700 MHz 256 MB available RAM 52 MB free hard disk space	HP OpenView NNM 6.4 or 7.01 One of these browsers: <ul style="list-style-type: none"> • Microsoft Internet Explorer 5.5 or 6.0 • Netscape Navigator 6.2 or 7.1 Sun's Java Plug-in 1.4 (http://java.sun.com/products/plugin/)
Solaris 8	Sun Ultra 5 512 MB available RAM 160 MB free hard disk space	HP OpenView NNM 6.3 or 6.4 CDE, XWindows, or OSF Motif Netscape Navigator 6.2 or 7.0 Sun's Java Plug-in 1.4 (http://java.sun.com/products/plugin/)

Platform	Hardware	Software
HP-UX 11.x	HP B2000 512 MB available RAM 65 MB free hard disk space	HP OpenView NNM 6.3 or 6.4 HP VUE, CDE, XWindows, or OSF Motif Netscape Navigator 6.2 or 7.0 Sun's Java Plug-in 1.4 (http://java.sun.com/products/plugin/)

Database

The following table describes the databases and corresponding drivers that are supported within MetaConsole.

Database	Notes
HSQldb HSQldb 1.7.0	This is the default database that ships with MetaConsole.
MSSQL 2000 (Version 8.0) MSSQL 2000 Driver for JDBC	For details about installation this database, see <i>Installing MSSQL Database</i> , beginning on page 14. The Microsoft MSSQL database must be purchased and licensed.
MySQL Admin 1.4 MySQL Connector/J 2.0.14 for JDBC	For details about installation, <i>Installing MySQL Database</i> , beginning on page 14. The MySQL software is free and licensed under the General Public License. The software can be downloaded at http://www.mysql.com

Placement of Components

You can install the client on as many computers as you like. Each client must be able to communicate with at least one MetaConsole server. The client and server can be on the same computer.

You can install the MetaConsole server on as many computers as you like. A MetaConsole server can service multiple clients. These clients can be of different types (browsers, MMC, HP OpenView, and so on). For more information, see *Using Multiple MetaConsole Servers* on page 6.

Installing MetaConsole Components

An installation program guides you through the process of installing the MetaConsole server, client, or both.

To use the installation program:

1. Insert the MetaConsole CD (for installers distributed via CD-ROM) or locate the `install.html` file (for installers distributed remotely via the Web).
2. Start the installation program:
 - CD-ROM distributions:

- [Windows] If the installation program does not start automatically, run `<INSTALLER_NAME>.exe` in the `Disk1\InstData\Windows` directory on the CD.
 - [HP-UX, Solaris, Linux]
 - a. Log in as `root`.
 - b. Execute `sh ./<INSTALLER_NAME>.bin` in the `Disk1\InstData\HPUX`, `Disk1\InstData\Solaris`, or `Disk1\InstData\Linux` directory on the CD.
 - Web-based distributions:
 - a. Open the `install.html` file in a web browser.
 - b. Click the **Start Installer** button.
The platform is detected and the platform-specific installer is saved locally and executed.
3. Accept the license agreement.
You cannot install and use MetaConsole if you do not accept the agreement.
 4. Select the component or components to install:
 - a. MetaConsole server
 - b. Client for MMC
For information, see *MetaConsole for MMC: Getting Started*.
 - c. Client for HP OpenView
 - d. Client for Tivoli NetView
For information, see *MetaConsole for NetView: Getting Started*.
 - e. Client for CA Unicenter
For information, see *MetaConsole for Unicenter: Getting Started*.
 5. Specify an installation directory.
The default directory is recommended.
On Windows, this is `C:\Program Files\MetaConsole`.
On HP-UX, Linux, and Solaris, this is `/opt/MetaConsole`.
 6. [MetaConsole server] Specify a TCP/IP port number.
The default port number, 80, is recommended, but if it is already in use, you can specify any unused port number.
 7. [Windows NT, 2000, and XP] [MetaConsole server] Indicate whether MetaConsole is to run as a Windows NT service.
 8. [Windows NT, 2000, and XP] [MetaConsole server] If you chose, in step 7, to run MetaConsole as a service, indicate whether it is to be started automatically at system startup or manually as needed.
 9. [client for HP OpenView or Tivoli NetView] Enter the name/address of the MetaConsole server host that the client will use. Optionally, specify a port number by following the name or address with a colon and the number.
(Example: `ServerMachineName:8080`)

10. [Windows] [client for HP OpenView] Select the folder where HP OpenView Network Node Manager is installed, and click **OK**.
11. [Windows] [client for Tivoli NetView] Select the drive where Tivoli NetView is installed, and click **OK**.
12. [client for Unicenter WorldView] Enter the name/address of the MetaConsole server host that the client for Unicenter will use. Optionally, specify a port number by following the name or address with a colon and the number.
(Example: *ServerMachineName* : 8080)
13. [Windows] [client for Unicenter WorldView] Select the drive where Unicenter TND is installed, and click **OK**.
14. [client for Unicenter WorldView] Enter the data repository name to be used.
15. [client for Unicenter WorldView] Indicate whether the Unicenter service is to be started automatically or manually by selecting **Manual** or **Automatic**.
16. Review the summary of installation options, and click **Install** to begin the installation.
17. Click **Done** to exit the installation program.

Starting the MetaConsole server after installation

Note: The MetaConsole installation procedure does not start the MetaConsole server. To use MetaConsole immediately after installation, you must manually start the MetaConsole server.

To start the MetaConsole server on Windows:

1. Click the **Start** button, point to **Programs**, and then point to **MetaConsole**.
2. Do one of the following:
 - To run the MetaConsole server as a program, click **RunMetaConsole**.
 - To run the MetaConsole server as a service, click **StartService**.

To start the MetaConsole server on HP-UX, Linux, and Solaris:

1. Change to the installation directory.
2. Execute `./RunMetaConsole`.

Uninstalling MetaConsole Components

On Windows platforms, you can uninstall MetaConsole components using Add/Remove Programs in Control Panel (they are listed under MetaConsole). Or you can use the **Uninstaller** command:

1. Click the **Start** button, point to **Programs**, and then point to **MetaConsole**.
2. Click **Uninstaller**.

On HP-UX, Solaris, and Linux, you uninstall MetaConsole components by executing the **Uninstaller** command:

1. Change to the `Uninstall` directory under the installation directory.
`Uninstall` is case sensitive.

2. Execute `./Uninstaller`.

Installing MSSQL Database

A simple database for storing alarm information is automatically installed with MetaConsole. The Microsoft SQL Server 2000 database can be used by the MetaConsole server to store alarm information. The following instructions describe how to configure the MetaConsole server to log alarms to a MSSQL database.

1. From <http://www.microsoft.com/sql/downloads>, download the JDBC driver for SQL Server 2000 for Windows.
2. Install the JDBC driver by executing the downloaded file.

The installation program create a `lib` subdirectory that contains the jar files `mssqldrivers.jar`, `msutil.jar`, and `msbase.jar`.

3. Copy the three jar files listed in step 2 into the `MetaConsole/lib` directory.
4. Modify the `local.xml` file so that the MetaConsole tables can be configured in the database.

The example below shows the contents of the `dataStore` element in the `local.xml` file. Modify the data shown in italics to connect to your MSSQL server.

```
<dataStore>
    DATABASE_NAME = "MSSQL"
    DATABASE_DRIVER="com.microsoft.jdbc.sqlserver.SQLServerDriver"
    CONNECTION_URL="jdbc:microsoft:sqlserver://10.0.0.200:1433;DatabaseName=MetaConsoleDB;SelectMethod=cursor"
    USER_NAME="root"
    PASSWORD="password">
</dataStore>
```

5. Save and close the `local.xml` file.
6. Make sure the database specified in the `local.xml` file is running and the MetaConsole database has been created.
7. Execute the `runcreator` file in the MetaConsole directory.
8. Modify the `configuration.txt` file (located in the MetaConsole directory) to indicate the alarm database.

This program creates the tables in your MetaConsole database.

Chapter 6 explains the `configuration.txt` file. The database parameter values entered in the `configuration.txt` file should be the same as those entered in the `local.xml` file.

9. Start the MetaConsole server.

Installing MySQL Database

A simple database for storing alarm information is automatically installed with MetaConsole. The MySQL database can be used by the MetaConsole server to store alarm information. The following instructions describe how to configure the MetaConsole server to log alarms to a MySQL database.

1. From <http://www.mysql.com>, download MySQL Connector/J.
2. Extract the `mysql-connector-java-version#.jar` file, and rename it `mysql.jar`.
3. Copy `mysql.jar` into the `MetaConsole/lib` directory.
4. Modify the `local.xml` file so that the MetaConsole tables can be configured in the database.

The example below shows the contents of the `dataStore` element in the `local.xml` file. Modify the data shown in italics to connect to your MSSQL server.

```
<dataStore>
  DATABASE_NAME = "MySQL"
  DATABASE_DRIVER="org.gjt.mm.mysql.Driver"
  CONNECTION_URL="jdbc:mysql://10.0.0.207/ks"
  USER_NAME="root"
  PASSWORD="password">
</dataStore>
```

5. Save and close the `local.xml` file.
6. Make sure that the database specified in the `local.xml` file is running and that the MetaConsole database has been created.
7. Execute the `runcreator` file in the MetaConsole directory.

This program creates the tables in your MetaConsole database.

8. Modify the `configuration.txt` file (located in the MetaConsole directory) to indicate the alarm database.

Chapter 6 explains the `configuration.txt` file. The database parameter values entered in the `configuration.txt` file should be the same as those entered in the `local.xml` file.

9. Start the MetaConsole server.

Chapter 4. Managing Devices

When OpenView program is started, the MetaConsole OpenView snapin runs automatically. When a new OpenView map is opened, the snapin checks each symbol on the map to see if it can be managed. If the symbol represents an object that can be managed by the MetaConsole OpenView snapin, the symbol changes to an executable icon with a three-dimensional appearance.

To view information or change settings for one of these devices, double-click its icon or right-click the icon and, on the menu that appears, click **Device Management**. A browser opens and displays management information related to that device.

The browser window is divided into two panes. You use the *navigation tree* in the *navigation pane* on the left to select the information you want displayed in the *details pane* on the right.

For details about using MetaConsole to manage a particular type of device, see the client user's guide for that device.

Refreshing Displayed Information

You can manually update the MetaConsole information that is displayed in the MetaConsole client (browser) window.

To update the information that is displayed:

1. In the navigation pane, right-click the node whose information you want to update.
2. In the menu that appears, click **Refresh**.
 - Refreshing a service provider gets a new list of discovered devices.
 - Refreshing a specific page of device information updates the information on that page.

Chapter 5. Configuring MetaConsole

You can change settings that apply to:

- All of MetaConsole
To change these settings, which include which other MetaConsole servers are visible and how alarms are presented: Right-click a map node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.
- A MetaConsole server
To change these settings, right-click a map node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **Server Configuration**.
- A service provider
To change these settings, right-click a map node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **Service Provider Configuration**. For information about configuration at the service provider level, see the MetaConsole client user's guide for a service provider.

The MetaConsole client window has two panes. You use the *navigation tree* in the *navigation pane* on the left to select the information you want displayed in the *details pane* on the right.

You can also perform MetaConsole configuration in a browser window that you open manually:

- Point your browser to the name or IP address of the computer running the MetaConsole server.

Examples:

```
http://MyMetaConsoleServer
http://190.190.0.0
```

If the MetaConsole server is running on a port other than 80, specify the port number:

```
http://MyMetaConsoleServer:8080
```

When you use the manual method, the navigation tree includes

- A **Help** node for access to MetaConsole online help
- A **Configuration** node for configuring the preferred server's settings
These are the settings you would see if you followed the instructions under the "All of MetaConsole" bullet item above.
- A node for each MetaConsole server, which in turn contains
 - A **Version Information** node for displaying component version numbers
 - A **Service Providers** node for enabling and disabling service providers
 - A **View Alarms** node for displaying and acknowledging alarm information

- A node for each enabled service provider (For details, see the client user's guide for a particular service provider.)

All configuration settings for the server or a service provider are maintained by the MetaConsole server and are not client specific. All clients use the same values; if any client changes a particular setting, the change affects all clients that use that setting.

Refreshing Displayed Information

You can manually update the MetaConsole information that is displayed in OpenView and in the MetaConsole client (browser) window.

MetaConsole client

To update displayed information:

1. In the navigation pane, right-click the node whose information you want to update.
2. In the menu that appears, click **Refresh**.
 - Refreshing a service provider gets a new list of discovered devices.
 - Refreshing a specific page of device information updates the information on that page.

Configuring the Server List

To add to the list of MetaConsole servers:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.
A MetaConsole client browser window opens.
2. In the navigation pane, expand the **Configuration** node and click **Server Discovery**.
3. In the **MC Server Address** boxes, type the IP address of the MetaConsole server you are adding to the list.
4. In the **Port** box, type the port number where the MetaConsole server is found.
5. Click **Add**.

The new server appears in the **Server List** box and in the navigation tree.

To remove a MetaConsole server from the list:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.
A MetaConsole client browser window opens.
2. In the navigation pane, expand the **Configuration** node and click **Server Discovery**.
3. In the **Servers** box, click the address of the MetaConsole server you are removing from the list.
4. Click **Remove**.

The server is removed from the list and from the navigation tree.

Specifying Alarm Notification Methods

The list of events you want to trigger alarm notifications is configured individually for each service provider at each MetaConsole server. How MetaConsole presents those alarms is determined at the highest level of MetaConsole configuration.

To specify how MetaConsole notifies you of alarms:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.

A MetaConsole client browser window opens.

2. In the navigation pane, expand the **Configuration** node and click **Alarm Notification Methods**.

3. Select the check box for each notification method you want MetaConsole to use.

If you select **Alert Box**, each alarm causes a message to appear on screen, indicating the device name or IP address where the alarm originated.

If you select **Audible Alarm**, each alarm causes a beep.

If you select **Email**, each alarm causes the sending of a formatted email message containing alarm time, device information, and alarm text to the recipients specified in the **Email Settings** form (see below).

If you select **Pager**, each alarm causes the sending of a formatted text page message containing alarm time, device information, and alarm text to the pager devices specified in the **Pager Settings** form (see page 20).

In any case, you can view details about the alarm; see *Viewing Alarms* on page 22.

4. Click **Apply**.

To specify settings for email notification:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.

A MetaConsole client browser window opens.

2. In the navigation pane, expand the **Configuration** node and click **Alarm Notification Methods**.

3. Click the **Settings...** button next to the **Email** check box.

Email notification settings appear at the bottom of the details pane.

4. Specify primary recipients by doing one or both of the following:

- To add a recipient to the **To: List**: In the **To: Email Address** box, type an email address; then click the adjacent **Add** button.
- To remove an address from the **To: List**: Click the address; then click the adjacent **Remove** button.

5. Specify “carbon copy” recipients by doing one or both of the following:
 - To add a recipient to the **Cc: List**: In the **Cc: Email Address** box, type an email address; then click the adjacent **Add** button.
 - To remove an address from the **Cc: List**: Click the address; then click the adjacent **Remove** button.
6. Specify “blind carbon copy” recipients by doing one or both of the following:
 - To add a recipient to the **Bcc: List**: In the **Bcc: Email Address** box, type an email address; then click the adjacent **Add** button.
 - To remove an address from the **Bcc: List**: Click the address; then click the adjacent **Remove** button.
7. In the **From** box, type the email address to be displayed as the sender of alarm email messages.

Typically, this is a network administrator’s email address.
8. In the **Subject** box, type text to be included in alarm email messages as the subject.

Typically, this text indicates the purpose of the message (for example, “Device Alert!”).
9. In the **Relay Server** box, type the name or IP address of the local network’s email (SMTP) server computer.

Note: Alarm notification email cannot be delivered if this information is missing or incorrect.
10. To send a test email message to the recipients specified in steps 4, 5, and 6, select the **Test Email on Apply** check box.
11. Click **Apply**.

To specify settings for pager notification:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **MetaConsole Configuration**.

A MetaConsole client browser window opens.
2. In the navigation pane, expand the **Configuration** node and click **Alarm Notification Methods**.
3. Click the **Settings...** button next to the **Pager** check box.

Email notification settings appear at the bottom of the details pane.
4. Specify primary recipients by doing one or both of the following:
 - To add a recipient to the **Pager List**:
 - a. In the **Pager ID** box, type a pager ID (typically a 10- or 11-digit phone number).
 - b. In the **Pager Service** list, click the name of the paging service to be used.
 - c. Click the adjacent **Add** button.

- To remove a recipient from the **Pager List**: Click the pager ID; then click the adjacent **Remove** button.
5. In the **From** box, type the email address to be displayed as the originator of alarm page messages.
Typically, this is a network administrator's email address.
 6. In the **Subject** box, type text to be included in page messages as the subject.
Typically, this text indicates the purpose of the message (for example, "Device Alert!").
 7. In the **Relay Server** box, type the name or IP address of the local network's email (SMTP) server computer.
Note: The SMTP server is used to send pages. Alarm notification pages cannot be delivered if this information is missing or incorrect.
 8. To send a test page to the recipients specified in step 4, select the **Test Page on Apply** check box.
 9. Click **Apply**.

Getting MetaConsole Component Version Information

To help you with support issues, technicians sometimes must know the version numbers of your MetaConsole components (server, service providers, and client).

To determine version numbers for the MetaConsole server and service providers:

→ In the navigation pane, expand the MetaConsole server node, and click **Version Information**.

The version number of the MetaConsole server software is shown at the top of the page. Below it are version numbers for all service providers, including

- Persistent Devices — a special service provider that manages persistent data
- ServerList Devices — a special service provider that manages the list of MetaConsole servers
- DataStore Devices — a special service provider that manages communication between the service provider and alarm database.

To get the version number of the MetaConsole client:

1. In the main OpenView window, on the **Help** menu, click **About HP OpenView**.
The **About HP OpenView** window opens.
2. Click **Applications...**
The **Application Index** window opens.
3. In the **Applications** list, click **MetaConsole**.
The MetaConsole client version number is displayed.

Enabling and Disabling Service Providers

To enable a service provider:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **Server Configuration**.
A MetaConsole client browser window opens.
2. In the navigation pane, click **Service Providers**.
3. In the **Disabled** list, click the name of the service provider you want to enable.
4. Click **Enable**.
5. To manually update discovered information in the map or the navigation pane, see *Refreshing Displayed Information* on page 18.

To disable a service provider:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **Server Configuration**.
A MetaConsole client browser window opens.
2. In the navigation pane, click **Service Providers**.
3. In the **Enabled** list, click the name of the service provider you want to disable.
4. Click **Disable**.
5. To manually update discovered information in the map or the navigation pane, see *Refreshing Displayed Information* on page 18.

Viewing Alarms

Note: The time that is displayed for each alarm is the time the MetaConsole server logged that alarm, and is based on the MetaConsole server's time.

To view alarms:

1. In the OpenView map, right-click a node that is managed by the MetaConsole OpenView snapin, and on the menu that appears, click **Server Configuration**.
A MetaConsole client browser window opens.
2. In the navigation pane, click **View Alarms**.
If the database is not configured, a message indicates that fact. Otherwise, alarm log information displays for up to 50 alarms.
3. To view more alarms, click one of the following in the **Action** list:
 - **Get Next 50 Alarms**
 - **Get Previous 50 Alarms**
 - **View All Alarms**

Viewing all alarms can be slow if there are many alarms in the database.

3. In the **Sort by** list, click **Most Recent Alarm** or **Device** to sort the alarms.

Most Recent Alarm shows the most recent alarms at the top of the page and older alarms at the bottom. **Device** shows alarms by device, with numbers in ascending order (for example, 10.0.0.1, 10.0.0.2) followed by letters in alphabetical order (for example, Alpha, Beta).

4. In the **Filter by** list, click an option to determine which alarms are shown.

New Alarms limits the display to log entries that have not been acknowledged.

Show All Alarms shows all acknowledged and unacknowledged log entries.

The name of a service provider shows all alarms, both acknowledged and unacknowledged, for that service provider.

To acknowledge an alarm as having been read:

- Select the **Acknowledge** check box for that alarm, and select the Update Current View from the **Action** list.

To acknowledge all alarms as having been read:

- Select the Acknowledge All Alarms from the **Action** list.

When you acknowledge an entry, its red text changes to gray and the alarm does not display when you click **New Alarms** in the **Filter by** list.

To delete all entries from the alarm log:

- Select Delete All Alarms from the **Action** list.

To acknowledge or delete specific entries:

- Select the **Acknowledge** or **Delete** check box for each alarm entry and select Update Current View from the **Action** list.

Chapter 6. Configuration Text Files

Configuration of MetaConsole server properties is governed by entries in a text file, `configuration.txt`. The MetaConsole client has a corresponding properties file, `OVImp.properties`.

Note: Changes in `OVImp.properties` do not take effect until you stop and restart Network Node Manager.

Configuration.txt

The `configuration.txt` file is created during installation and resides in the top-level installation directory. If the file is not present, MetaConsole uses the default values, originally set at installation time.

Configuration file entries have the form `keyword=value`. This chapter includes an example file and a description of each keyword (including the keyword's value when MetaConsole is shipped).

Example file

```
# HTTP server port
Port=80

# HTTP server root directory
Root=root

# SNMP Providers - disabling/enabling sets
AllowWrites=true

# Service providers folder
ServiceProviderDirectory=mcsp

# whether logging to console is needed
LogToConsole=true

# whether logging to a stream (file) is needed
LogToFile=false

# file to log to
LogFile=MetaConsole.log

# whether file logged to is appended, or rewritten
AppendToFile=false

# start page for the MetaConsole server, start page must
# be in the root for the server
StartPage=start.html

# will log HTTP query type and client IP address for all
# connections
LogClientIP=false

# ALARM DATABASE PROPERTIES

# The alarm database driver
DataBaseDriver=org.hsqldb.jdbcDriver
```

```
# The URL for the alarm database
URLConnection=jdbc:hsqldb:MetaConsoleDB

# The username for the alarm database
Username=sa

# The password for the alarm database
Password=#
```

Keywords

Keyword	Description	Comments
Port	HTTP server port number.	Original value is 80 .
Root	HTTP server root directory name.	Original value is root . Do not change this value.
AllowWrites	Indicates whether the server will allow sets.	Original value is true . Do not change this value.
ServiceProviderDirectory	Name of the service provider directory.	Original value is mcsp . Do not change this value.
LogToConsole	Indicates whether status and error information generated by the MetaConsole server is displayed on the user's screen.	Valid values are true and false .
LogToFile	Indicates whether status and error information generated by the MetaConsole server is written to a file.	Valid values are true and false . See next entry (LogFile).
LogFile	File to which status and error information is written if previous entry (LogToFile) is true .	
AppendToFile	Indicates whether new status and error information is appended to existing information in the log file, instead of overwriting it.	Valid values are true (append) and false (overwrite). Original value is false .
StartPage	File name of the server start page.	Original value is start.html . Do not change this value.
LogClientIP	Indicates whether HTTP query type and IP Address information are logged for each connection.	Valid values are true (log) and false (do not log). Original value is false .
DataBaseDriver	Indicates the driver used for the alarm database	Valid values are JDBC drivers
URLConnection	Indicates the information required to connect to the alarm database	Valid values are JDBC URL connections
Username	Indicates the user who is authorized to connect to the alarm database	Valid values are database user names
Password	Indicates the password for the authorized user	Valid values are dictated by the database

OVImp.properties

In this file, keywords and values are separated by white space.

Example file

```
LogToFile           true
LogToScreen        true
LogFile            OVImp.log
HostName           100.0.0.99
PortNumber         8080
MetaConsoleConfiguration MetaConsole Configuration

MetaConsoleConfigurationPage /start.html
ServerConfigurationPage     /start.html
ServiceProviderConfigurationPage /start.html
DeviceConfigurationPage     /start.html
```

Keywords

Keyword	Description	Comments
LogToFile	Indicates whether errors generated by the client for OpenView are sent to the file named in LogFile (see below).	True or False.
LogToScreen	Indicates whether errors generated by the client for OpenView are displayed on the user's screen.	
LogFile	File to which information is written if LogToFile (see above) is true .	
HostName	The host name of the MetaConsole server, specified as an IP address (such as 100.0.0.99).	
PortNumber	The port number (such as 8080).	If you do not specify a port, 80 is assumed.
MetaConsole-Configuration	The string used by the client for OpenView for MetaConsole configuration.	Do not change this value.
MetaConsole-ConfigurationPage	The start page for MetaConsole configuration.	Do not change this value.
ServerConfiguration-Page	The start page for server configuration.	Do not change this value.
ServiceProvider-ConfigurationPage	The start page for service provider configuration.	Do not change this value.
DeviceConfiguration-Page	The start page for device configuration.	Do not change this value.

Chapter 7. Frequently Asked Questions

With the HP OpenView client running, how do I know whether the MetaConsole snapin is installed and running?

Open the **Legend** option in the **Help** menu. In the **Legend** list, select **Icon Symbol Types**. If the snapin is installed, MetaConsole icon symbols are present.

The snapin runs automatically when the HP OpenView client runs. Icons for objects that the snapin manages have a 3D appearance. If you right-click these icons, MetaConsole menu options are displayed. If you double-click the icons, the default web browser is launched.

How can I manage the symbols in the HP OpenView client using an alternative MetaConsole server?

The `configuration.txt` file in the MetaConsole/HPOV directory indicates which server the HP OpenView client uses to manage its symbols. You should modify this file when the MetaConsole server address or port number changes. For details, see page 24.

When does MetaConsole manage symbols in the device map?

MetaConsole manages symbols for the devices it discovers. The snapin periodically scans all devices discovered by MetaConsole service providers and compares these with the HP OpenView symbols to determine whether to manage the symbols.

Why isn't MetaConsole managing a symbol in the HP OpenView map?

MetaConsole manages only those symbols corresponding to devices MetaConsole has discovered.

Why does the MetaConsole J (browser client) tree display devices that are not in the HP OpenView device maps?

HP OpenView and MetaConsole have separate discovery mechanisms and can discover different devices. Both can be configured to discover devices on remote networks, and both allow the manual addition of devices/symbols. You should configure MetaConsole service providers to discover devices in the same networks as the HP OpenView client. For details, see the client user's guide for a particular service provider.

MetaConsole allows me to manually add devices that are not discovered automatically. Why don't these devices appear in the HP OpenView device maps?

The MetaConsole snapin does not add symbols to the HP OpenView client maps. If you want to manage a device using HP OpenView and the OpenView client did not discover the device, you must manually add it to the map. Be sure that the added symbol has the same IP address as the device that you added manually to MetaConsole.

If multiple service providers discover a device, how can I control which one manages the corresponding symbol in the HP OpenView map?

MetaConsole uses an internal priority scheme to determine which service provider manages the OpenView symbols. To ensure that a certain provider manages a symbol, you can block other providers from discovering the device. For details, see the client user's guide for a particular service provider.

When does a provider on a manually added server manage a symbol in the HP OpenView device map?

In two cases:

- When a service provider on both the preferred server and the manually added server discover the same device, and the provider on the manually added server has a higher priority than the provider on the preferred server
- When no provider on the preferred server discovers the device

How can I manage devices on networks other than the local subnet?

The HP OpenView client must be configured to discover devices on the local subnet and the other networks. For details about adding a range of IP addresses for discovery, see the client user's guide for a particular service provider.

Another approach is to run an additional MetaConsole server on the remote network and add the server. For details about adding a server to the server list, see the client user's guide for a particular service provider.

When the MetaConsole snapin is installed and the HP OpenView client is running, but no symbols are managed by the snapin, how should I configure the MetaConsole environment?

Point a browser to the MetaConsole server (for example, `http://myserver:8080`). If the MetaConsole snapin manages symbols in the HP OpenView maps, right-clicking the symbol displays menu options for configuring the MetaConsole environment.

Why doesn't the browser launch on HPUX when I double-click on a managed device in HP OpenView?

The MetaConsole snapin for HP OpenView will launch the default browser when a managed device is double-clicked. You must make sure the browser is in your PATH.

Why doesn't HP OpenView receive traps from my device?

If the MetaConsole server is running on the same machine as HP OpenView, the traps will be intercepted by the MetaConsole server. If you want both HP OpenView and MetaConsole to receive traps, they should be installed on different machines.