

MetaConsole SDK: Getting Started

version 2.3



Copyright © 2001-2004 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.

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

LICENSE AGREEMENT FOR METACONSOLE™ SOFTWARE DEVELOPMENT KIT

1. BACKGROUND.

This License Agreement ("Agreement") is a legal agreement between you (either an individual or a single entity -- "Licensee") and NETAPHOR SOFTWARE, INC. ("Netaphor") for the version of the MetaConsole™ Software Development Kit provided to Licensee by Netaphor, which includes computer software and associated media and printed materials, and may include "online" or electronic documentation (collectively the "Software Product"). BY OPENING THIS PACKAGE, OR DOWNLOADING OR INSTALLING THE SOFTWARE PRODUCT TO YOUR COMPUTER, YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT OPEN THIS PACKAGE, OR DOWNLOAD OR INSTALL THIS SOFTWARE PRODUCT, AND PROMPTLY RETURN IT UNOPENED TO THE PLACE WHERE YOU OBTAINED IT. If you do not agree to the terms of this Agreement, you may not install, copy or use the SOFTWARE PRODUCT. Netaphor is located at 15520 Rockfield Boulevard, Suite E, Irvine, California.

2. OWNERSHIP OF SOFTWARE PRODUCT.

The Software Product is owned exclusively by Netaphor, and is protected by copyright laws and international copyright treaties, as well as other intellectual property laws. This Agreement does not convey to Licensee an interest in or to the Software Product, but only a limited and revocable right of use. The Software Product is licensed (not sold) to you by Netaphor in accordance with the terms of this Agreement.

3. GRANT OF LICENSE.

3.1 Definitions.

- 3.1.1 "Designated User" means and refers to Licensee if Licensee is an individual. If Licensee is an entity, "Designated User" means and refers to a single, designated employee or consultant of Licensee. Designation shall be made by submitting written notice to Netaphor (e-mail to licenses@netaphor.com is acceptable). At any give time, only a single individual may be the Designated User.
- 3.1.2 "Designated Computer" means and refers to the single computer (or computer system) designated by Licensee to run, access and operate the Software Product. Designation shall be made by submitting written notice to Netaphor (e-mail to licenses@netaphor.com is acceptable). At any give time, only a single computer (or computer system) may be the Designated Computer.

3.2 Regular License.

When a Regular License for the Software Product (as opposed to an Evaluation license) is purchased by Licensee, Licensee is bound by the Terms of the MetaConsole SDK License Agreement executed by and between Netaphor and Licensee.

3.3 Evaluation License.

Licensee is hereby granted the following non-exclusive rights:

- 3.3.1 Licensee may install the Software Product on the Designated Computer for use by the Designated User.
- 3.3.2 The Designated User may use the Software Product for the sole purpose of evaluating the functions and features of the Software Product. Licensee may not use or distribute any applications developed with the Software Product ("Applications"), and Licensee may not distribute the Software Product (or any portions thereof) in any form whatsoever.

3.3.3 Licensee may only use the Software Product for a period of ninety (90) days from the date on which Licensee first installed the Software Product, or until the Software Product ceases functioning, whichever occurs first (the "Termination Date"). Within five (5) days after the Termination Date, Licensee shall (i) delete (uninstall) the Software Product and all Applications from the computer (or network) they are installed on, and (ii) destroy all copies of the Software Product and the Applications.

4. **RESTRICTIONS AND OTHER PROVISIONS.**

4.1 Licensee will not and will not knowingly permit any third party to (a) remove, deface, bypass, over-ride or otherwise defeat any product identification, copyright notices, trademarks, restricted rights or other proprietary restrictions, or any license administration or enforcement mechanisms contained in or affixed to the Software Product, (b) use the Software Product for any purpose other than as expressly permitted in this Agreement, (c) reverse engineer, decompile, disassemble, trace or translate the Software Product, or do anything to attempt to discover the Software Product's source code, (d) prepare any derivative works based on the Software Product, (e) modify, adapt or alter the Software Product, (f) copy the Software Product or transfer, or (g) transfer, sell, assign, pledge, lease, rent or share the Software Product or Licensee's rights in the Software Product.

4.2 Netaphor reserves the right to alter, modify or change, without prior notice, any aspect or feature of the Software Product.

5. **TERMINATION.**

This Agreement is effective upon Licensee's (a) acceptance of the Software Product, or (b) the downloading of the Software Product from Netaphor's web site or other authorized electronic medium, and this Agreement shall continue in effect until terminated by Netaphor. Netaphor may terminate this Agreement upon the breach by Licensee of any material term hereof. Termination of this Agreement by Netaphor shall automatically, and without further action by Netaphor, terminate and extinguish all licenses granted by Netaphor to Licensee under this Agreement. Upon such termination by Netaphor, Licensee agrees to delete the Software Product from the hard drive of the computer that it is installed on and destroy all copies of the Software Product.

6. **SUPPORT SERVICES.**

At Netaphor's sole discretion, Netaphor may provide Licensee with support services for the Software Product ("Support Services") as described, from time to time, on Netaphor's world wide web site (<http://www.netaphor.com>). Licensee agrees and acknowledges that Netaphor has no obligation to provide Support Services.

7. **NO WARRANTIES.**

TO THE MAXIMUM EXTENT PERMITTED BY LAW, NETAPHOR EXPRESSLY DISCLAIMS ANY WARRANTY FOR THE SOFTWARE PRODUCT AND ANY SUPPORT SERVICES PROVIDED BY NETAPHOR. THE SOFTWARE PRODUCT AND SUPPORT SERVICES 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 SOLELY WITH LICENSEE.

8. **LIMITATION OF LIABILITY.**

LICENSEE AGREES AND ACKNOWLEDGES THAT NETAPHOR AND ITS' SUPPLIER'S SHALL HAVE NO LIABILITY, WHETHER IN CONTRACT, TORT (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE OF ANY TYPE AND STRICT LIABILITY) OR OTHERWISE, ARISING OUT OF OR RELATED TO LICENSEE'S OR ANY OTHER PARTY'S USE OF, OR INABILITY TO USE, THE SOFTWARE PRODUCT OR ANY ACCOMPANYING PRINTED MATERIALS OR DOCUMENTS, OR NETAPHOR'S PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES.

MOREOVER, AND WITHOUT IN ANY WAY LIMITING THE PRIOR PARAGRAPH, IN NO EVENT SHALL NETAPHOR OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, CONSEQUENTIAL, PUNITIVE OR MULTIPLE DAMAGES WHATSOEVER, WHETHER IN CONTRACT, TORT (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE OF ANY TYPE AND STRICT LIABILITY) OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF OR RELATED TO LICENSEE'S OR ANY OTHER PARTY'S USE OF, OR INABILITY TO USE, THE SOFTWARE PRODUCT OR ANY ACCOMPANYING PRINTED MATERIALS OR DOCUMENTS, OR NETAPHOR'S 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 FORESEEABLE.

9. **ENTIRE AGREEMENT AND AMENDMENTS.**

This Agreement constitutes the entire agreement between the parties concerning the subject matter hereof and supersedes all prior and contemplated agreements or representations, written or oral, of the parties pertaining to the subject matter. This Agreement may not be modified except in a written amendment signed by authorized representatives of both parties.

10. **SEVERABILITY.**

If any provision of this Agreement is declared to be illegal, unenforceable or void, the remainder of this Agreement shall be enforced to the extent permitted by law, and the illegal, unenforceable or void provision shall be replaced with a mutually acceptable provision which comes closest to the intention of the parties underlying the original provision.

11. **APPLICABLE LAW.**

This Agreement shall be interpreted, construed and governed by the laws of the State of California without regard to its conflict of law rules. The parties hereby irrevocably submit to the exclusive jurisdiction and venue of the state and federal courts sitting in Orange County, California for the purpose of all legal proceedings arising out of or relating to this Agreement.

12. **WAIVER.**

No delay or omission by either party to exercise any right or remedy hereunder shall be construed as a waiver of such right or remedy. Further, the waiver by either party of a particular breach of this Agreement by the other party shall not be construed as, or constitute, a continuing waiver of such breach, or of other breaches of the same or other provisions of the Agreement.

13. **PREVAILING PARTY.**

In the event a dispute arising under this Agreement results in litigation, the non-prevailing party shall pay the prevailing party's reasonable litigation costs, including, without limitation, reasonable attorneys' fees.

14. **COMPLIANCE WITH U.S. EXPORT LAWS.**

LICENSEE UNDERSTANDS, AGREES AND WARRANTS THAT IT SHALL NOT TRANSFER, DIVERT, EXPORT OR RE-EXPORT TO ANY THIRD PARTY ANY ITEM PROVIDED TO LICENSEE UNDER OR IN CONNECTION WITH THIS AGREEMENT, EXCEPT AS EXPRESSLY AUTHORIZED BY THE U.S. GOVERNMENT IN ACCORDANCE WITH U.S. EXPORT CONTROL LAWS.

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.

Contents

Overview	1
Audience and Purpose	1
Chapter 1. Prerequisites	2
Background Knowledge.....	2
Hardware	2
Software.....	2
Chapter 2. How MetaConsole Works	4
Component Overview	5
Communication Protocol	5
XML Queries	6
Persistence	6
Chapter 3. SDK Contents	7
Service Provider (Non-UI)	7
Service Provider Structure	7
Service Mapping File.....	7
Discovery.....	7
Alarms	7
Debugging	7
Building the Service Provider	8
Limitations	8
Server	8
Service Provider UI	8
Navigation Pane	8
Details Pane	9
Help	9
Host Utilities.....	9
Configuration Persistence	9
Installer	9
Samples	9
Chapter 4. Using the SDK	10
Understanding Your Management Data	10
UI Changes	10
Modifying the Service Provider	10
Integration and Testing.....	11
Glossary	12

Overview

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

The MetaConsole architecture includes three major components:

- MetaConsole client, the front end that integrates with the console
- MetaConsole server, the back end
- MetaConsole service providers, the components containing device- or service-specific knowledge

The MetaConsole SDK (software development kit) enables you to modify a MetaConsole service provider by

- Changing the user interface (for example, by modifying the UI of the details pane, adding nodes to the navigation tree, replacing the icons that are displayed, choosing the object used to trigger the icon status, or modifying the help files)
- (For SNMP service providers)
 - Adding or removing MIB objects
 - Changing the set of MIB objects used for discovery of devices
- Modifying the groups of objects for which changes in values can trigger alarm notifications
- Localize a service provider

Audience and Purpose

This guide and the SDK are intended for software developers who want to modify a MetaConsole service provider. This guide is simply an overview of the SDK's capabilities and use; its purpose is to show you how to find the information that you need and to help you plan your modifications using the MetaConsole SDK. For details, see the *MetaConsole SDK Manual*.

Chapter 1. Prerequisites

Background Knowledge

To use the MetaConsole SDK, you should be familiar with

- MetaConsole
- Java
- JavaScript
- XML, HTML, and XSLT
- Management protocols such as SNMP, HTTP, and LDAP

Hardware

- 400 MHz or faster Pentium class CPU
- At least 128 MB of RAM (at least 256 MB when used in conjunction with a management console such as HP OpenView or Tivoli NetView)
- At least 200 MB free disk space (500 MB recommended)

Software

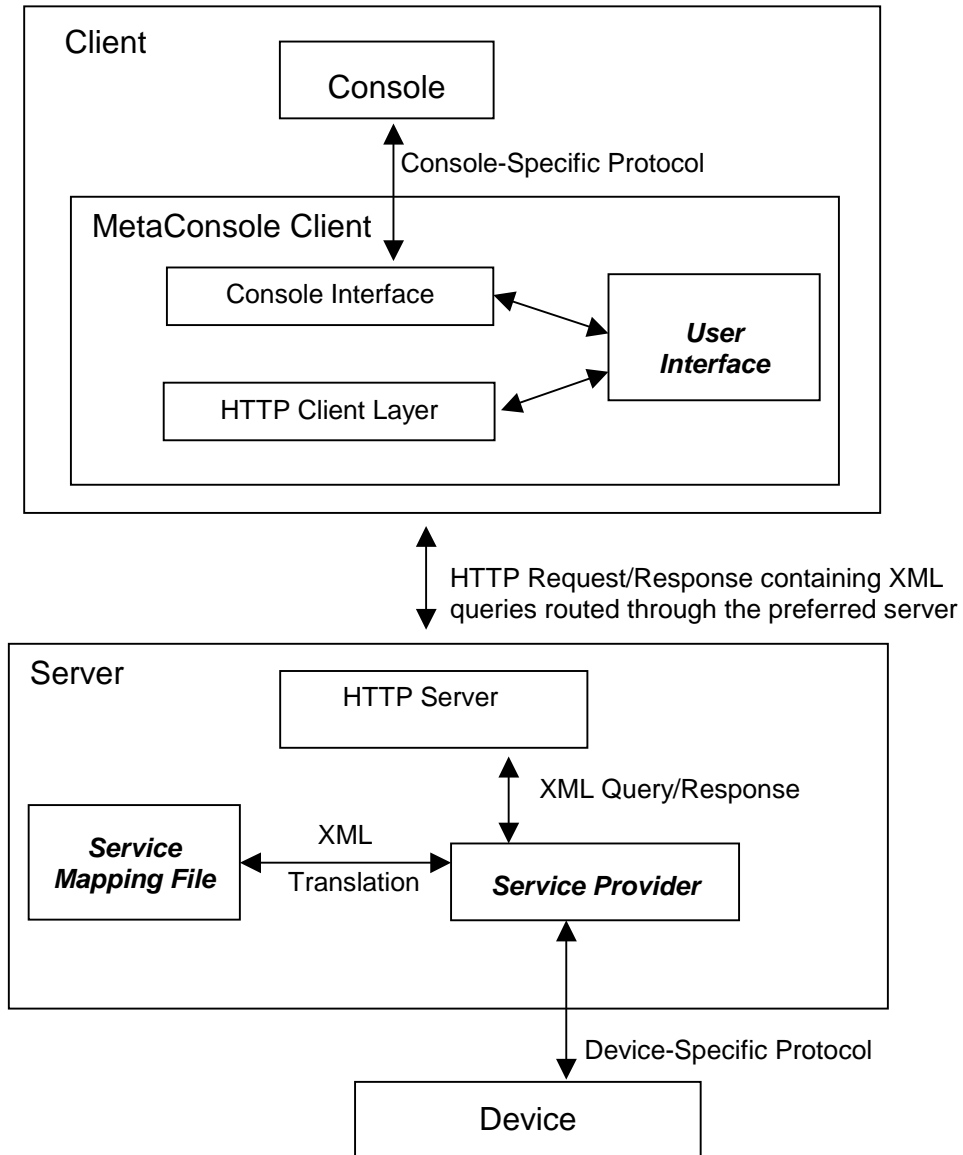
- One of the following operating systems:
 - Microsoft Windows NT 4.0 Workstation (SP6) or Server (SP6)
 - Microsoft Windows 2000 Professional (SP1 or later) or Server (SP1 or later)
 - Other MetaConsole-supported operating systems, such as HP UX and Sun Solaris, only as needed for development (when using platform-specific JNI calls) and testing
- J2SE SDK from Sun Microsystems, version 1.1.8, 1.2.2, or 1.3.0
- A text editor (an integrated development environment, such as Microsoft's Visual J++, Metrowerks' CodeWarrior, or IBM's Visual Age for Java strongly recommended)
- Internet Explorer or Netscape Navigator
- HP OpenView Network Node Manager, Tivoli NetView, MMC, or other platform software if you are developing MetaConsole solutions for that platform
- InstallAnywhere Enterprise Edition from ZeroG Software, to create installation programs for the developed software
- Other software as needed for management that is not SNMP- or LDAP-based
- C++ compilers if you are developing cross-platform code not written in Java

Additionally, you must have a good understanding of the management aspects of the device or software service that the service provider you are modifying is written for. As an example, if your device or software service is managed using SNMP, you need to understand the MIB that the device or software service implements. You also need an instance of your device or software service for testing your modified service provider.

Chapter 2. How MetaConsole Works

A MetaConsole client integrates with consoles, such as Microsoft Management Console (MMC), HP OpenView, and Tivoli NetView, presenting management options for network devices and services through the console's user interface. A service provider contains device- or service-specific knowledge needed for passing information between a client and a device or service.

The following diagram shows, in *italics>*, elements of MetaConsole that you can modify using the SDK.



Component Overview

MetaConsole consists of three major types of components – the MetaConsole client, the MetaConsole server, and the various service providers. MetaConsole clients implement the interface to the management console (such as HP OpenView) and are responsible for displaying the user interface. MetaConsole clients communicate with the MetaConsole server.

The MetaConsole server implements a protocol that allows MetaConsole clients to manage target devices and services. The protocol is implemented in the form of a query described in XML that is communicated to the MetaConsole server over HTTP. (Use of HTTP allows MetaConsole clients and servers to communicate over firewalls.) The MetaConsole server then directs the query to the appropriate MetaConsole service provider.

Each service provider contains the knowledge to communicate, for management purposes, with a particular device, service, or class of devices or services. Service providers may use a standard protocol or proprietary protocols to access the device. The standard protocol that is implemented with MetaConsole and available to a service provider to use is SNMP. When managing a class of devices that implement SNMP, a service provider must understand the syntax and semantics of the MIBs implemented by those devices. It must translate XML queries from a client into the device protocol and responses from the device into an XML response for the client.

Communication Protocol

MetaConsole clients do not know about either the syntax or the semantics of the management protocol for a particular target device or service and are completely device independent. Likewise, MetaConsole service providers have no client-specific knowledge.

The common ground is the interface between the two, communicated via the MetaConsole server in the form of the XML query. By using a query described in XML, MetaConsole abstracts the management protocols of the various target devices and services and allows any MetaConsole client to work with any MetaConsole service provider. Service providers can implement any protocol that is needed to communicate with a device; devices that use management protocols that are not natively supported by a management console can be supported in that console via the MetaConsole client.

Additionally, both clients and service providers can be developed independently. This gives developers creating and modifying MetaConsole service providers the big advantage of having to worry about their particular service provider only. Once the service provider is developed, it can be accessed by all MetaConsole clients.

XML Queries

Queries from MetaConsole clients are made for:

- Discovering servers and service providers that may be running at a server
- Discovering target devices and services that can be managed by a service provider
- Reading management data from a target device or service
- Setting management data for a target device or service
- Setting up monitoring of alarms
- Reading and setting persistent configuration information

Persistence

MetaConsole provides a mechanism for service providers and MetaConsole clients to persist data on a MetaConsole server. The following can be persisted:

- Server configuration data
- Alarm log and alarm notification mechanisms
- Service provider configuration data
- Device /service configuration data (like community names, which are device-specific)

Chapter 3. SDK Contents

The MetaConsole SDK consists of

- A *MetaConsole SDK Manual* that describes and explains key elements of MetaConsole and MetaConsole development. (For an overview of those elements, see below.)

The manual also includes high-level procedures you can follow to modify a MetaConsole service provider. (The manual cannot, of course, provide detailed instructions for every specific change you might want to make. The procedures provide a context for your understanding of details presented elsewhere.)

- An example service provider that shows clearly how a service provider is structured. This is a complete working example that contains the service mapping file, service provider layout file, device layout file, and other files described in the *MetaConsole SDK Manual*. It is localized to Japanese and English to illustrate MetaConsole's localization.

Service Provider (Non-UI)

Service Provider Structure

All components of a service provider and how they are packaged (for example, what directory structures appear inside the JAR file). It is a road map of the composition of a service provider, using the current model. It emphasizes the set of files, what they do, and where they go.

Service Mapping File

The structure of the SMF. The SMF is the core of a service provider. Particular attention is given to explaining how to add the MIB for a new (type of) device. Excerpts from the example service provider's SMF are used for illustration.

Discovery

How discovery works and how it is configured from the UI. References are made back to the SMF documentation for an explanation of the discovery section of the SMF and its relationship to the discovery process.

Alarms

The various ways alarms are reported in MetaConsole and any requirements of the client in this process. Alarms in MetaConsole are the result of a mechanism to detect changes in values of device or service objects and act upon them asynchronously.

Also documented is the alarm log. This includes how alarms can be added to the log, how the log is stored, and how the alarm log is accessed for viewing.

Debugging

The tools and techniques used to debug problems with service providers are part of the SDK.

Building the Service Provider

The mechanism for building the service provider. All builds must be done on a Windows platform.

Limitations

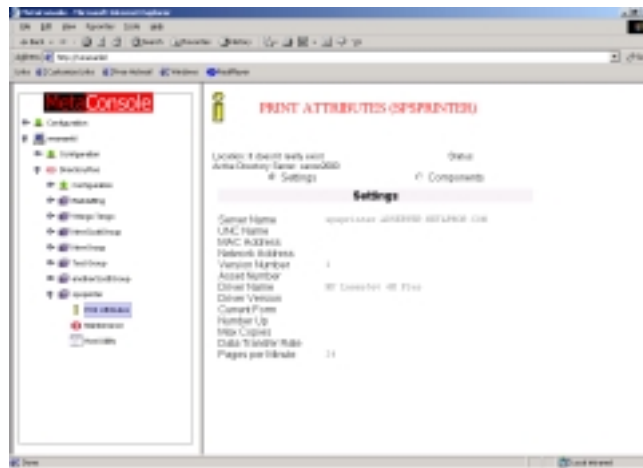
Limitations of MetaConsole that may impact the operation of service providers. Knowledge of these limitations enables a developer to correctly implement a solution.

Server

A MetaConsole server can route queries and responses to and from other MetaConsole servers. A multi-server configuration is useful under an assortment of circumstances (for example, in a large network, where traffic is reduced when each server has a relatively small set of devices to manage).

Service Provider UI

The MetaConsole window has two panes. Users make selections in the *navigation pane* on the left to display information in the *details pane* on the right.



Navigation Pane

Service Provider Layout, Device Layout

The Service Provider Layout and Device Layout are XML files that describe which elements are available for a service provider in the navigation pane in the form of configuration and device nodes. The structure of the file is documented in detail. Excerpts from the example service provider's Service Provider Layout file are used as illustrations.

Icons

The navigation pane includes icons. Which icon is used for a node is specified in the layout file that specifies that node. Consequently, each layout file deals with which icons are to be used under which circumstances. The different consoles handle icons differently; in some cases, the icon must be chosen from a fixed list.

Details Pane

JavaScript

JavaScript is used to build the XML requests sent to the server to request pages. Some scripts are specific to a service provider UI, and others are standard across service providers. The documentation describes which of the JavaScript scripts are intended to remain unchanged. The documentation describes how you can add pages to the service provider UI by creating new scripts. The documentation also describes how to modify existing scripts.

Style Sheets

Although documenting of style sheets and XSLT is well beyond the purview of the SDK (there are whole books on the subject), the role that style sheets play in MetaConsole and some excerpts from the current model examples are described.

Help

The MetaConsole client's help files are HTML files presented in a WebHelp format. WebHelp displays the files in a user-friendly manner in Internet Explorer, Netscape Navigator, and other browsers and on Windows, Linux, and UNIX platforms. Making substantial changes to the help requires RoboHelp Office from eHelp Corporation.

Host Utilities

Many devices include an embedded web server for configuration. Instead, of duplicating functions available in the embedded web server, MetaConsole clients link to it. The service provider UI provides a standard way of invoking the web server.

Configuration Persistence

The Persistent Service Provider is used to persist configuration information. Examples of how it's used within the service provider UI are described.

Installer

After modification, the components of a service provider must be packaged and deployed. Packaging instructions indicate the needed files and directory structures.

A third-party installer program called ZeroG Install Anywhere is required. The SDK includes instructions for using ZeroG Install Anywhere to prepare the installable package. ZeroG Install Anywhere has its own documentation as well.

Samples

A sample service provider with source is provided, along with the source to your service provider. Additionally, the sample service provider has been localized to Japanese.

Chapter 4. Using the SDK

You should plan your service provider changes before making them. This chapter contains suggestions that may be useful.

Understanding Your Management Data

The first step in working with the MetaConsole SDK should be to clearly understand the management data for your device or service. Specifically, when working with a service provider that uses SNMP as its management protocol, you should understand the MIBs for the target devices or services.

In addition to understanding the MIB (or schema), you must know any special details about the agent's (within the target device or service) implementation of the MIB (or schema). This information is typically not described within the MIB (or schema) but may be explained in supplementary agent implementation documentation for your target devices or services.

UI Changes

The next step is to design and make the UI changes. UI changes might include replacing graphics on a results page, adding new managed objects on a page, and adding new pages. Start by making the minor UI changes, and then proceed to making the complicated changes.

You may also make UI changes for localization purposes. This includes adding support for new languages that are currently not supported by your service provider, and updating localized versions with support for functionality added to your existing UI.

Modifying the Service Provider

The SDK includes all the source files for your service provider. This includes any pages, graphics, and help files that are provided by your service provider to a MetaConsole client requesting such, the SMF for your service provider, and any Java "classes" required by your service provider.

Modifications to the SMF include:

- Adding new discovery objects for discovery of additional target devices or services, or modifying existing discovery objects
- Adding or updating managed objects ("tags") for use by a MetaConsole client

Other modifications may be required to the service provider class for your service provider to:

- Add any special handling of management data not supported in the base providers for SNMP
- Add additional functionality and processing for other functions, such as compression

The "build" process for your service provider will create a new "jar" file. You may be required to modify this build process if it does not correctly handle all of your service provider's needs.

Integration and Testing

Your SDK includes all sources for your current service provider as created by Netaphor. Prior to starting any modifications or additions, it is a good idea to “build” your service provider and test it to make sure that you have an environment that correctly represents what your current service provider is supposed to do. This step will also help you in later stages of your development; if some modification does not work as it is supposed to, you can go back to the original environment and compare it with the updated one to see what differences exist. This is a useful step in eliminating bugs.

You should test each modification before proceeding. Install your new service provider, start MetaConsole, and check the added functionality. If it works, proceed to the next change. If it does not work as expected, use the debugging tools provided with the MetaConsole SDK release version. Make sure the query is valid, the SMF changes are correct, and the UI changes to the style sheet, JavaScript, or HTML are correct. Try using a network capture utility to ensure that the correct data is being communicated between your client and the MetaConsole server and then between your service provider and the target device or service.

Finally, you should test your service provider with additional service providers available from Netaphor’s web site. In this way, you can ensure that your modifications did not introduce conflicts.

Glossary

Following are definitions of MetaConsole terms found in this guide.

client

See MetaConsole client.

console

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.

details pane

In a MetaConsole window, the pane on the right, containing information relevant to the node selected in the navigation pane on the left. *Also called* results pane.

MetaConsole client

The component that integrates with consoles, such as Microsoft Management Console (MMC), HP OpenView, and Tivoli NetView, presenting management options for network devices and services through the console's user interface.

MetaConsole server

The back-end component that communicates with MetaConsole clients and with the devices to be managed.

navigation pane

In a MetaConsole client window, the pane on the left, containing the navigation tree.

navigation tree

The hierarchical structure displayed in the navigation pane of a MetaConsole client window. *Also called* console tree.

node

A location on the navigation tree. The tree typically contains nodes for configuration of a MetaConsole service provider and for each managed device or service. A node can itself contain nodes for specific aspects of configuration or management.

server

See MetaConsole server.

service mapping file

The core of a service provider. The service mapping file maps query element tags to service data items.

service provider

The component that translates XML queries into the protocols understood by the managed device or service. The service provider is also responsible for discovery of devices or services of the type that it manages.