

Sun Cobalt™ Control Station

Developer Guide



Copyright © 1997-2001 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, California 94303, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and other countries.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution and decompilation. No part of the product or of this document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Sun Cobalt, Sun Cobalt RaQ, Sun Cobalt CacheRaQ, Sun Cobalt Qube and the Sun Cobalt logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Netscape and Netscape Navigator are trademarks or registered trademarks of Netscape Communication Corporation in the United States and other countries.

Linux is a trademark of Linus Torvalds.

Federal Acquisitions: Commercial Software - Government Users Subject to Standard License Terms and Conditions.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright © 1997-2001 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, California 94303, U.S.A. Tous droits réservés.

Sun Microsystems, Inc. détient des droits de propriété intellectuelle sur la technologie réunie dans le produit qui est décrit par ce document. Ces droits de propriété intellectuelle peuvent s'appliquer en particulier, sans toutefois s'y limiter, à un ou plusieurs des brevets américains répertoriés à l'adresse <http://www.sun.com/patents> et à un ou plusieurs brevets supplémentaires ou brevets en instance aux Etats-Unis et dans d'autres pays.

Ce produit ou document est distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Sun Cobalt, Sun Cobalt RaQ, Sun Cobalt CacheRaQ, Sun Cobalt Qube et le logo Sun Cobalt sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

Netscape et Netscape Navigator sont des marques de fabrique ou des marques déposées de Netscape Communication Corporation aux Etats-Unis et dans d'autres pays.

Linux est une marque de fabrique de Linus Torvalds.

LA DOCUMENTATION EST FOURNIE "EN L'ETAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.

Part Number / Numéro de pièce : 816-3802-10

Date : 11-2001

Preface

This Developer Guide is for anyone who will create control modules for the Sun Cobalt™ Control Station.

Developers must be familiar with the procedure for creating Red Hat Package Manager (RPM) files and with the GNU `make` utility.

This guide consists of the following chapters and appendices:

Chapter 1 — “Introduction” on page 1” includes an overview of the Sun Cobalt Control Station and of this guide .

Chapter 2 — “Module Files” on page 5 describes the files that are found in a control module and the method for building a control module. The files specific to the Appliance Inventory control module are discussed.

Chapter 3 — “Appliance Inventory Module” on page 19 provides all of the software code for the Appliance Inventory control module.

Chapter 4 — “Network Tool Module” on page 55 provides all of the software code for the Network Tool control module.

Appendix A — “Licenses” on page 59 lists licensing information.

Contents

Preface	iii
1 Introduction	1
Overview of the Sun Cobalt Control Station	1
Purpose of this document	2
Intended audience	2
Independent software vendors and developers	2
IT departments and in-house design teams	3
Terminology note	3
Related documents	3
2 Module Files	5
Format of the control module	5
Components	8
packing_list.xml file	8
Components in the control station	9
GUI section	9
Back-end sections	10
Components on the target appliance	12
Making a control module	13
Information on the Makefile	15
Makefile variables	15
Directory structure	15
Module directories	17
Post-install directory structure	18
3 Appliance Inventory Module	19
packing_list.xml	19
Control Station	21
GUI section	21
Menu items	21
Web page items	22
Localization file	27
Back-end section	34
Target appliance	47
i386-based appliances	51
mips-based appliances	52

4 Network Tool Module	55
Control Station	55
A Licenses	59
The BSD Copyright	59
GNU General Public License	60
SSL License	63
Acme Public License	64

Introduction

The Sun Cobalt™ Control Station is the latest addition to the Sun Cobalt line of server-appliance solutions. The control station is more than just a server-appliance management tool—it is a new platform for pushing software and services to your appliance customers.

Through the Sun Cobalt Control Station, you can fully control the distribution of software payloads, offering customized and tailor-made services to downstream and end-user customers. By leveraging the Sun Cobalt BlueLinQ technology, all available software updates and patches can be accessed and distributed to Sun Cobalt Qube™ and RaQ™ server appliances, as you designate.

In addition to distributing your own server-appliance solutions through the control station, you can also create control modules by utilizing the Sun Cobalt Control Station Developer Essentials tools. These tools provide you with everything you will need either to incorporate your utilities or to build new capabilities into the module architecture of the control station.

The Sun Cobalt Control Station Developer Essentials tools are available through the Sun Cobalt Developer Network.

Overview of the Sun Cobalt Control Station

Managing appliances and delivering services to them is a challenge due to the number of systems involved as well as the types of tools that are used. The framework and applications must be extensible to allow businesses to tailor the solution to meet business needs and processes, as well as their evolution. These needs drove the design of the Sun Cobalt Control Station. The control station was built with a simple framework that can incorporate modular applications to meet these requirements. Further, the control modules use a module-file format that allows users or third parties to create and integrate their own tools to create a unified management and service-delivery platform in a very simple and easy fashion.

The Sun Cobalt Control Station consists of two parts: a core framework that is the engine for executing control modules and the control modules themselves. The framework for the control station was designed to deploy and run system control modules. These control modules can come from Sun Microsystems, Inc., from third-party vendors or from your own in-house design team.

The control station allows complete flexibility. Control modules can perform almost any operation, from querying and configuring target appliances to daemons running on the control station or target appliance. Apart from following the format of the module file, there are no restrictions on the control module. Once you have loaded a control module on the control station and it has been properly registered, the control station acts as a vehicle only, passing calls to the user-defined applications and handling any returns.

Figure 1 shows the Control Modules section of the control station user interface; the modules appear in the menu on the left side.

Figure 1. Control Modules Splash screen



Purpose of this document

The document explains:

1. how to turn an existing application or service into a control module, and
2. how to create a new control module that will provide a specific application or service.

Intended audience

Developers must be familiar with the procedure for creating Red Hat Package Manager (RPM) files and with the GNU make utility.

This Developer Guide is intended for two primary audiences:

1. Independent software vendors (ISVs) and developers.
2. Information Technology (IT) departments and in-house design teams of users of the control station.

Independent software vendors and developers

Independent software vendors or developers who have created applications or services for Sun Cobalt appliances now have a powerful tool for deploying those applications and services.

The Sun Cobalt Control Station can be both a vehicle to deploy software payloads as well as a platform to deliver new services. The control station can also act as an extension of an existing application (for example, a Web interface for an application on the network), while leveraging the existing capabilities such as group operations.

IT departments and in-house design teams

End customers can convert an internal, homegrown utility or script into a control module and apply it to a much larger number of server appliances through a control station. At the same time, they can leverage the existing capabilities of the control station itself. This offers a number of advantages:

1. You can retain legacy systems and procedures, thus eliminating the need to buy new software or pay outside consultants to create a unique solution for you.
2. In turn, you avoid the additional costs of having to train employees on a new piece of software, as well as the drop in efficiency due to the learning curve.

Terminology note

The control station and the managed appliances function in a client/server-style relation. However, given that the managed appliances are still *servers*, we thought it would be confusing to refer to them as *clients*. In this Developer Guide, we talk of the control station and of *target appliance(s)*. A target appliance is an appliance that has been imported into the control station framework.

You may see references or tags in the software code labelled `client`. This refers to a target appliance.

Related documents

- The Sun Cobalt Control Station user manuals are available on the control station itself or on our Web site at <http://www.cobalt.com/support/resources/manuals.html>.
- The Sun Cobalt Qube™ 3 Developer's Guide for developing on the Sun Cobalt Qube 3 software is available exclusively to Sun Cobalt Developer Network members.

The Sun Cobalt Control Station is based on the Sun Cobalt Qube 3 software architecture.

Module Files

This chapter describes the components of a control module and how to construct a control module.

Format of the control module

A control module is based on Red Hat Package Manager (RPM) files. Each module is a gzipped tar file containing three components; each component is a distinct entity and is handled separately by the installer. See Figure 2.

The three components are:

1. The `packing_list.xml` file.

This XML file defines the meta-data such as the name, the vendor and the version of the module. This file also defines the RPM files intended for the control station and those intended for the target appliances.

2. The RPM files to be installed on the control station.

The executables and libraries contained within these RPM files are responsible for generating the graphical-user-interface (GUI) components for the control module and for interfacing with the application programming interfaces (APIs) on the Sun Cobalt™ Control Station. They may also interface with any of the executables and libraries contained within the RPM files on the target appliance(s).

3. The RPM files to be installed on the target appliance.



Note: The control-module format is very flexible and none of the three components are mandatory. The components required depend on what you want your control module to do.

These components can be brought together using the GNU `make` utility to create the control module.

Figure 2 shows the components of a control-module file.

Figure 3 shows the “Managed Appliance Inventory” table.

Figure 4 shows the detailed inventory tables for a particular appliance.

Figure 2. Components of a control-module package file

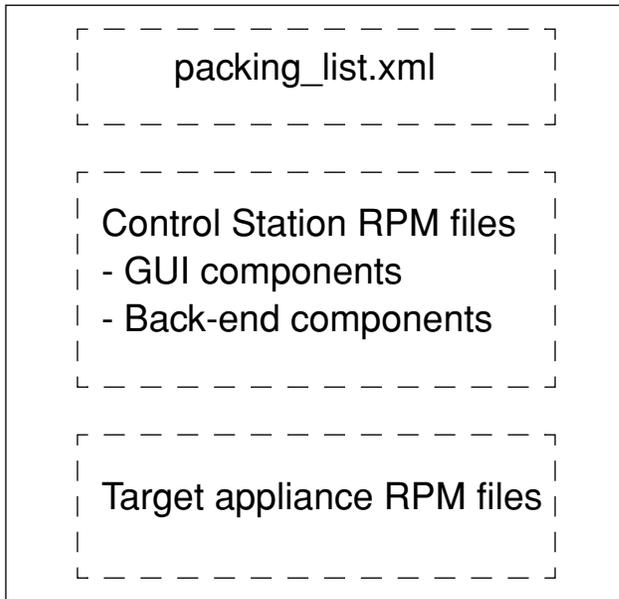


Figure 3. Managed Appliance Inventory table

Update

Managed Appliance Inventory									
IP Address	Host Name	Class	CPU's	MHz	RAM (MB)	NIC's	Drives	Capacity (GB)	Detail
10.9.24.118	10.9.24.118	CacheRaQ 4	1	448	256	2	1	18.52	
10.9.32.109	10.9.32.109	RaQ 4	1	448	512	2	1	27.36	
10.9.33.238	10.9.33.238	Qube 3	1	298	64	2	1	11.75	

3 Entries

Figure 4. Detailed Inventory tables

Back

Appliance Details	
IP Address	10.9.24.118
Host name	10.9.24.118
Build Class	CacheRaQ 4
Kernel	Linux
Version	2.2.16C27_III

System Memory	
Installed RAM (MB)	256
Swap space (MB)	128

CPU	
CPU Number	0
Name	AuthenticAMD
Model	AMD-K6(tm)-III Processor
CPU speed (MHz)	448

Network Interface Cards			
2 Entries			
Ethernet number ▼	IP Address ▼	Host Name ▼	MAC Address ▼
0	10.9.24.118	bmula8	00:10:E0:03:40:15
1	0.0.0.0		00:10:E0:03:3F:4E

Disks		
1 Entry		
Device name ▼	Partitions ▼	Capacity (MB) ▼
hda	3	18963

Filesystems			
3 Entries			
Device name ▼	Partition ▼	Capacity (MB) ▼	Mount Point ▼
hda	4	18013	/home
hda	1	750	/
hda	3	200	/var

Components

packing_list.xml file

The `packing_list.xml` file defines the meta-data for the control module, such as the name, the vendor and the version number. The configuration file also defines the RPM files intended for the control station and those intended for the target appliances.



Note: The string values shown in the table are sample values.

Table 1. XML file format

Tags	Description
<code><appPkg></code>	This opens the header tag <code>appPkg</code> .
<code><app></code>	This opens the header tag <code>app</code> .
<code><appName value = "Name of Application"/></code> <code><appVersion value = "Version string"/></code> <code><appVendor value = "String for name of vendor."/></code>	The tags define the application name, vendor and version of this control module. The name and vendor are used to determine if another instance of the application is installed. The version is a string that is used to allow updates.
<code><nameTag value = "[[base-mgmt-inventory.mappInventory]]"/></code> <code><versionTag value = "[[base-mgmt-inventory.mappVersion]]"/></code> <code><vendorTag value = "[[base-mgmt-inventory.mappVendor]]"/></code>	These names can be viewed from the Administer Modules screen on the control station UI. They are internationalization tags based on the internationalization modules in the Sun Cobalt Qube™ 3 software architecture.
<code><msRpm value = "RPMfile1.rpm"/></code> <code><msRpm value = "RPMfile2.rpm"/></code> --you can have any number of RPM files here -- <code><msRpm value = "RPMfile8.rpm"/></code> <code><msRpm value = "RPMfile9.rpm"/></code>	This is the list of RPM files to be installed on the control station.
<code><appUpdate boolean = "true"></code>	<p>This boolean attribute defines the type of error checking that should be performed when a control module is installed.</p> <p>If <code>true</code>, this module file is an update; if <code>false</code>, it is a complete module.</p> <p>When a control module is loaded into the control station, the installer verifies whether an existing module matches the name and vendor. If there is no match, the module is installed.</p> <p>However, if another module with the same name and vendor is already installed, this field is examined. If the module is a complete module, it is not installed and an error message is displayed.</p> <p>If the module is an update with a higher version number than one currently installed, the older module is uninstalled and the newer one installed.</p>

Table 1. XML file format

Tags	Description
<cleanupScript value= "Full path and name to an executable script"/>	The <code>uninstall</code> script is run when a control module is uninstalled. In this way, the control module can perform the operations necessary to clean itself up. This can include, for example, the removal of temp files and database tables. The control station passes on appliance-database IDs as command-line arguments to this script.
<initScript value ="full path and name of executable script"/>	The <code>init</code> script initializes the control module when a new target appliance is added. In this way, the module can get configuration data or preload a database. The control station passes on appliance-database IDs as command-line arguments to this script.
<device>	This specifies the product definitions for the RPM files that are to be loaded when a new target appliance is added.
<buildType value ="Build Number like 3001R"/> (must match exactly with the device) or <devClass value = "any string like RaQ3R"/> (must match exactly with the device)	<p><code>buildType</code> is a build number for Sun Cobalt products. This field is verified to determine whether the specified RPM file should be installed on the target appliance.</p> <p>The tag <code>devClass</code> can also be used rather than <code>buildType</code>. <code>devClass</code> is a class of Sun Cobalt product, such as the Sun Cobalt RaQ™ 4 server appliance.</p>
<rpm value = "RPMclient_file1.rpm"/> <rpm value = "RPMclient_file2.rpm"/> -- you can have any number of RPM files here -- <rpm value = "RPMclient_file8.rpm"/> <rpm value = "RPMclient_file9.rpm"/>	This is the list of RPM files to be installed on the target appliances that match the class or type of appliance specified in the row above.
</device >	<p>This tag closes off the <code>device</code> tag.</p> <p>The device can be repeated for each type of appliance supported by a control module.</p>
</app>	This tag closes off the header tag <code>app</code> .
</appPkg>	This tag closes off the header tag <code>appPkg</code> .

Components in the control station

GUI section

The GUI format is based on the Sun Cobalt Qube 3 software. For more details on this interface and its programming interfaces, refer to the Sun Cobalt Qube 3 Developer's Guide (available on the Sun Cobalt Developers Web site at <http://developer.cobalt.com>).

Back-end sections

The control station was designed for simplicity so the number of application programming interfaces (APIs) was kept to a minimum. A short list of APIs can achieve all the results needed by most control modules.

Currently, these APIs are supported in Perl. The Perl modules which define these APIs are located in the directory `/usr/mgmt/lib/perl5`.

The following Perl modules contain the public functions:

- **RawAgentInterface.pm**. The functions for accessing the target appliance(s).
- **Appliance.pm**. The functions for translating the target appliance id into an IP address and vice-versa.
- **Task.pm**. The functions for manipulating tasks, which are accessible through the task UI in the control station.
- **Event.pm**. The functions for creating events, which are accessible through the event UI in the control station.

The following tables explain each of the public functions in more detail.

Table 2. RawAgentInterface.pm Perl module

Function	Explanation
<code>ConnStart (ip_address)</code>	<ul style="list-style-type: none">• <code>ConnStart</code> starts a connection to the agent on the IP address.• It returns a connection handle.
<code>SendFile (connection, source, dest)</code>	<ul style="list-style-type: none">• <code>SendFile</code> sends the file with the pathname <code>local_path</code> to the target appliance <i>host</i>• <i>connection</i> is a connection handle.• If <code>remote_path</code> is specified, then the file is renamed to <code>remote_path</code> on the target appliance after the transfer completes. Otherwise, the file is placed in <code>/var/tmp</code>.• If the file transfer is successful, a value of “1” is returned.• If the file transfer fails, a value of “-1” is returned.
<code>GetFile (connection, source, dest)</code>	<ul style="list-style-type: none">• <code>GetFile</code> gets the file specified by <code>remote_path</code> from the remote system <i>host</i>.• <i>connection</i> is a connection handle.• If <code>local_path</code> is specified, the file is renamed to <code>local_path</code> on the control station after the transfer completes. Otherwise, the file is placed in <code>/var/tmp</code>.• If the file transfer is successful, a value of “1” is returned.• If the file transfer fails, a value of “-1” is returned.

Table 2. RawAgentInterface.pm Perl module

Function	Explanation
RunCmd (connection, cmd)	<ul style="list-style-type: none"> RunCmd executes the commands <code>cmd_arg0</code> through <code>cmd_argN</code> on the target appliance. <code>connection</code> is a connection handle. If the command is successful, the output from the command is returned. If the command fails, a value of “-1” is returned.
ConnEnd (connection)	<ul style="list-style-type: none"> ConnEnd terminates a connection to the agent. <code>connection</code> is a connection handle. If the termination is successful, a value of “1” is returned. If the termination fails, a value of “-1” is returned.

Table 3. Appliance.pm Perl module

Function	Explanation
getApplianceID (ip)	<ul style="list-style-type: none"> <code>getApplianceID</code> retrieves the internal appliance ID for the target appliance with the IP address <code>ip</code>. If the appliance ID is not found, a value less than 0 is returned.
getApplianceIPAddress (id)	<ul style="list-style-type: none"> <code>getApplianceIPAddress</code> retrieves the IP address of the appliance with the internal appliance ID <code>id</code>. If the IP address is not found, a value less than 0 is returned.

Table 4. Task.pm Perl module

Function	Explanation
addTask (name)	<ul style="list-style-type: none"> <code>addTask</code> creates a new task with the specified <code>name</code>. You can view tasks through the task UI in the control station. If the task is created, a positive integer is returned (the task id). If the task is not created, a negative integer is returned.
setTaskMessage (task, msg)	<ul style="list-style-type: none"> <code>setTaskMessage</code> sets the message corresponding to the specified task id <code>task</code> to the string <code>msg</code>.
setTaskFailed (task, msg)	<ul style="list-style-type: none"> <code>setTaskMessage</code> sets the task specified by the task id <code>task</code> to failed and updates the message for that task to <code>msg</code>.
setTaskComplete (task, msg)	<ul style="list-style-type: none"> <code>setTaskComplete</code> sets the task specified by the task id <code>task</code> to complete and updates the message for that task to <code>msg</code>.

Table 5. Event.pm Perl module

Function	Explanation
<code>addEvent (task, type, ip, proc, 0, msg)</code>	<ul style="list-style-type: none">• <code>addEvent</code> creates a new event that you can view through the event UI in the control station.• An event is associated with a task, specified by the task id <i>task</i>. The following information is stored for each task:<ul style="list-style-type: none">type: a string describing the type of the event (for example, information, warning or error).ip: the system that generated the event; events generated on the control station should specify 127.0.0.1 as the IP address.proc: the name of the process that generated the event.msg: a string containing event information• If the event is created, a positive integer is returned (the event id).• If the event is not created, a negative integer is returned.

Components on the target appliance

When a server appliance is imported into the control station, the agent is installed on the appliance and a new directory (`/usr/mgmt/`) is created.

When you select, in the control station UI, a target appliance to be managed by a control module, the target-appliance components of that module are saved within the `/usr/mgmt/` directory that was previously created on the appliance.

For example, the target-appliance components of the Appliance Inventory module are stored in `/usr/mgmt/`.

Making a control module

The general flow for creating a control module has the following steps.



Note: In each step, the relevant files in the Appliance Inventory control module are listed. You can view the software code for each of these files in Chapter 3, “Appliance Inventory Module”.

1. Create an XML file and name it `packing_list.xml`. This file must be named `packing_list.xml`.

You must use a unique, vendor-specific name for your control module to avoid name conflicts.



Important: We use the word *base* in the filenames for module files, for example, `base-mgmt-inventory-1.0-16` and recommend that you not use *base* in your filename.

Developers must differentiate their modules by naming the modules with a distinctive name, preferably a name that reflects their company or product (for example, `modulename_vendor_version`).

2. Create the RPM file(s) for the control station. There are two sections for the control station: the GUI section and the non-GUI section.

GUI section

The GUI section covers the RPM files for:

- the menu items on the left side of the GUI
- the Web page(s)
- the localization of the content

For the menu items in the Appliance Inventory module, there are three files:

- a. `mgmt-inventory.xml` generates the **Appliance Inventory** menu item on the left hand side.
- b. `summary.xml` and `update.xml` generate the **Summary** and **Update** sub-menu items, respectively, under the **Appliance Inventory** item.

The structure of these files is based on the Sun Cobalt Qube 3 software architecture. Refer to the Sun Cobalt Qube 3 Developer’s Guide.

For the Web page items in the Appliance Inventory module, there are three files:

- a. `inventoryDetail.php` displays the Details page for a given target appliance.
- b. `inventorySummary.php` displays the Summary page for a given set of target appliances running the Appliance Inventory module.
- c. `updateInventory.php` updates the Appliance Inventory database tables while displaying a progress bar.

These three files also work in conjunction with other components of the control station (for example, the library files).

For the localization of content in the Appliance Inventory module, there is one file:

- a. `mgmt-inventory.po` contains the list of internationalization tags and strings for localizing the GUI.

The structure of this file is based on the Sun Cobalt Qube 3 software architecture. Refer to the Sun Cobalt Qube 3 Developer’s Guide.

Back-end section

The files for the back-end section of the control-station component are:

- a. `mgmtInventory.php` contains the library functions for the GUI.
- b. `Inventory.pm` contains the library functions for the back-end.
- c. `cleanupInventory.pl` is a script specified in the `packing_list.xml` file that cleans up the Appliance Inventory database tables in the control station if you remove the Appliance Inventory module.
- d. `getInventory.pl` is a back-end script that updates the Appliance Inventory database tables with the most recent information for the target appliance(s).
- e. `140_inventory_tables.sql` is the file that defines the Appliance Inventory database tables.

3. Create the RPM file(s) for the target appliance.

For the Appliance Inventory module, there is one file to load on to the target appliance.

- `inventory_get.pl` is a script on the target appliance that returns the inventory information to the control station.
- the `Makefiles` and the `spec` files in the client directory on the target appliance(s) are used to create the client RPM files.

`base-mgmt-inventory-i386-script.spec` is the spec file for appliances based on the i386 architecture

`base-mgmt-inventory-mips-script.spec` is the spec file for appliances based on the MIPS architecture.

When you click **Update** in the GUI, the system calls the `getInventory.pl` file. This file uses the `RunCmd` function in `RawAgentInterface.pm` API to run the `inventory_get.pl` script on the target appliance.

4. Use the Appliance Inventory Makefile to run `make mapp`.

The command `make mapp` creates the destination directories, places the files in the correct location, tars the whole directory and gzips the resulting `tar` file. It also generates the filename for the module and gives it the `.mapp` extension.

See “Information on the Makefile” on page 15 for an explanation of the variables and module directories.

You have now built the control module.

On the control station, use the **Administer Modules** menu item to add the control module that you just created. From this screen, you can browse for your control-module file.

Information on the Makefile

Makefile variables

Table 6 provides information about the `Makefile` variables.

Table 6. Top-level `Makefile` variables

Makefile variable	Description
VENDOR	The vendor field for your module.
VENDORNAME	The actual vendor name; this name can be the same as VENDOR.
SERVICE	The name for the service.
VERSION	The version number.
RELEASE	The release number.
BUILDARCH	If you do not want to generate platform-specific RPM files, set this variable to <code>noarch</code>
XLOCALEPAT	If you want to exclude any locale patterns, set this variable to a space-separated list of those locale patterns.
BUILDUI	This variable creates a package file of all the files in <code>ui/web</code> and <code>ui/menu</code>
BUILDLOCALE	Set this variable to <code>yes</code> to build these components, create RPM files and create a capstone RPM file.
BUILDSRC	This variable builds the files is in the <code>src</code> directory.
BUILDGLUE	If <code>BUILDGLUE</code> is set to <code>yes</code> , it wraps all the handlers, object schemas, conf files for the event triggers and the configuration files into a package file. If <code>BUILDGLUE</code> is set to <code>no</code> , it does not create a package file.
DEFLOCALE	This locale is used for static HTML pages (for example, <code>en</code> or <code>ja</code>).
EXTRAEXCLUDES	This variable defines the directories to ignore during a server build.

The `BUILD` variables determine which directories to include when calling the `clean`, `install`, and `rpm` targets.

The default make rules take the `BUILD` variables and build up `ui`, `glue` and `locale` RPM files, if requested. If any of these RPM files are generated, a capstone RPM file is created as well. A *capstone* is a type of packing list for the RPM files.

Directory structure

Before you build a control module with the `make` utility, the files that constitute the control module must be stored in a certain directory structure so that the module is built correctly. The `makefile` expects this directory layout when processing the files.

Figure 5 shows the directory structure.

Figure 5. Pre-build directory structure



Module directories

Table 7 provides information about the module directories.

Table 7. Module directories

Directory	Description
glue	Handler and configuration modification code.
ui	User interface and user-interface menu code.
locale	Locale information and locale-specific UI pages.
lib	Library code.
client	Code for the target appliance.
templates	User-modifiable template files used in generating the packing-list file and RPM file(s).
src	src directory (optional)

If your module does not contain compiled code, the `make` rules shown above should be all that you need for building a control module.

If your module does contain compiled code, you need to know some additional `make` rules. First, `make` checks for Makefiles in the `glue`, `src` and `ui` directories and, if they are present, uses them. You must prepend the `PREFIX` environment variable on the install phase of the Makefile so that RPM files are properly generated.

In addition, the `make rpm` rule does not touch the `src` directory, so you must create the RPM files you want from separate specification files. Update the `templates/rpmdefs.tpl` file to reflect any of these RPM files without version numbers. Create a file with the same name as the RPM file in the `RPMS` directory so that the appropriate version is included in the packing-list file.

Finally, you might need to edit the `templates/rpmdefs.tpl` file to add additional build, install and file targets for any files that you have. The `<begin [%]VARIABLE>` sections in the `rpmdefs.tpl` file correspond to the same `[VARIABLE_SECTION]` sections in the `templates/spec.tpl` file. If you want to add something to the `spec.tpl` file that is not dependent upon a single RPM file, you can add it directly to the `spec.tpl` file.



Note: If you have specified a `VENDORNAME`, the `make` utility searches first in `{glue, locale, ui, src}/$(VENDORNAME)` for files before searching in the `glue`, `locale`, `ui` and `src` directories.

Post-install directory structure

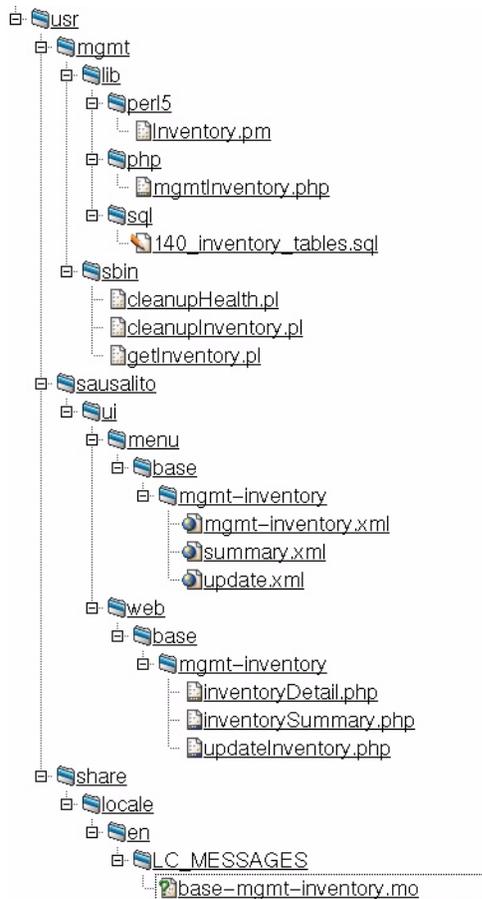
As part of the control-module installation process, the files are stored in the following directory structure.

The default make rules place these files in the following locations:

```
glue/conf/* -> $(CCEDIR)/conf/$(VENDOR)/$(SERVICE)/*
glue/handlers/* -> $(CCEDIR)/handlers/$(VENDOR)/$(SERVICE)/*
locale/localeX/$(SERVICE).po ->
  /usr/share/locale/localeX/LC_MESSAGES/$(VENDOR)-$(SERVICE).mo
ui/menu/* -> $(CCEDIR)/ui/menu/$(VENDOR)/$(SERVICE)/*
ui/web/* -> $(CCEDIR)/ui/web/$(VENDOR)/$(SERVICE)/*
bin/* -> /usr/mgmt/bin/*
sbin/* -> /usr/mgmt/sbin/*
lib/* -> /usr/mgmt/lib/*
```

Figure 6 shows the expanded post-install directory structure.

Figure 6. Post-install directory structure



Appliance Inventory Module

This chapter provides the files in the Appliance Inventory control module that ships with the Sun Cobalt™ Control Station. These files are described in Chapter 2 and are presented here in the same order as in Chapter 2.

Chapter 2, “Module Files”, explains how to build a control module for the control station.

packing_list.xml

```
<appPkg>
  <app>
    <name      value = "base-mgmt-inventory" />
    <version   value = "1.0-27" />
    <vendor    value = "Sun Microsystems" />
    <nameTag   value = "[[base-mgmt-inventory.mappInventory]]" />
    <versionTag value = "[[base-mgmt-inventory.mappVersion]]" />
    <vendorTag value = "[[base-mgmt-inventory.mappVendor]]" />
    <updateBool value = "Y" />

    <msRpm     value = "base-mgmt-inventory-capstone-1.0-21.noarch.rpm" />
    <msRpm     value = "base-mgmt-inventory-glue-1.0-21.noarch.rpm" />
    <msRpm     value = "base-mgmt-inventory-locale-en-1.0-21.noarch.rpm" />
    <msRpm     value = "base-mgmt-inventory-ui-1.0-21.noarch.rpm" />

    <initScript value = "/usr/mgmt/sbin/getInventory.pl" />
    <cleanupScript value = "/usr/mgmt/sbin/cleanupInventory.pl" />

    <device>
      <devClass value = "RaQ XTR" />
      <rpm       value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
    </device>

    <device>
      <devClass value = "RaQ XTRJ" />
      <rpm       value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
    </device>

    <device>
      <devClass value = "RaQ 4" />
      <rpm       value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
    </device>

    <device>
      <devClass value = "CacheRaQ 4" />
      <rpm       value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
    </device>
  </app>
</appPkg>
```

```

<device>
  <devClass value = "RaQ 4J" />
  <rpm value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "RaQ 3" />
  <rpm value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "RaQ 3J" />
  <rpm value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 3" />
  <rpm value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 3J" />
  <rpm value = "base-mgmt-inventory-i386-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 1" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 1J" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 2" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "Qube 2J" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "RaQ 1" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

<device>
  <devClass value = "RaQ 1J" />
  <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
</device>

```

```

    <device>
      <devClass value = "RaQ 2" />
      <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
    </device>

    <device>
      <devClass value = "RaQ 2J" />
      <rpm value = "base-mgmt-inventory-mips-scripts-1.0-6.noarch.rpm" />
    </device>

  </app>
</appPkg>

```

Control Station

GUI section

Menu items

mgmt-inventory.xml

```

<item
  id          = "base_inventory"
  label       = "[[base-mgmt-inventory.inventory]]"
  description = "[[base-mgmt-inventory.inventory_help]]"
>
  <parent id="base_applications" order="10" />
</item>

```

summary.xml

```

<item
  id          = "inventory_summary"
  label       = "[[base-mgmt-inventory.inventory_summary]]"
  description = "[[base-mgmt-inventory.inventory_summary_help]]"
  url        = "/base/appmgr/Selector/selektorInit.php?url=/base/mgmt-inventory/
inventorySummary.php&button_label=base-mgmt-inventory.viewSummaryButton&title=base-
mgmt-inventory.summarySelect&noItemsTag=base-bigdaddy.noAppliances"
>
  <parent id="base_inventory" order="62" />
</item>

```

update.xml

```

<item
  id          = "inventory_update"
  label       = "[[base-mgmt-inventory.inventory_update]]"
  description = "[[base-mgmt-inventory.inventory_update_help]]"
  url        = "/base/appmgr/Selector/selektorInit.php?url=/base/mgmt-inventory/
updateInventory.php&button_label=base-mgmt-inventory.inventory_update&title=base-
mgmt-inventory.updateSelect&noItemsTag=base-bigdaddy.noAppliances"
>
  <parent id="base_inventory" order="66" />
</item>

```

Web page items

inventoryDetail.php

```
<?php

/*
 * inventoryDetail.php
 * Copyright 2001, Sun Microsystems. All rights reserved.
 * $Id: inventoryDetail.php,v 1.22 2001/10/03 00:00:29 pploquin Exp $
 */

include('mgmtHelper.php');
include('mgmtInventory.php');
include('selectordata.php');

$helper    = new mgmtHelper('base-mgmt-inventory');
$factory   = $helper->getFactory('base-mgmt-inventory', $PHP_SELF);
$page     = $factory->getPage();
$i18n     = $helper->getI18n("base-mgmt-inventory");
$inventory = new Inventory();

print $page->toHeaderHtml();

$backButton = $factory->getBackButton('/base/mgmt-inventory/inventorySummary.php');
print $backButton->toHtml() . "<br>\n";

if ($applianceId)
{
    // We were called by the summary page to view details
    // for a single appliance
    $applianceList = array($applianceId);
}
else
{
    // We were called by the Appliance Manager with a list...
    $applianceList = getDataArray();
}
for ($i=0; $i < sizeof($applianceList); $i++)
{
    $applianceId = $applianceList[$i];

    if (($baseDetails = $inventory->getBaseAndMemoryDetails($applianceId))
    {
        printBlock('base_details',
            array('ip_address', 'host_name', 'build_class', 'product_serial_no',
'hw_serial_no', 'kernel', 'version'),
            $baseDetails);

        printBlock('memory_details',
            array('ram_mbytes', 'swap_mbytes'),
            $baseDetails);

        $inventory->getCPUDetails($applianceId);
        while ($inventory->next())
        {
            printBlock('cpu_details',
                array('cpu_number', 'name', 'model', 'cpu_mhz'), //, 'cache_kb'),
                $inventory->Record);
        }
    }
}
```

```

printList('nic_details',
    array('eth_number', 'nic_ip_address', 'nic_host_name', 'mac_address'),
    array('left', 'left', 'left', 'left'),
    $inventory->getNICdetails($applianceId));

printList('disk_details',
    array('device_name', 'num_partitions', 'device_capacity'),
    array('left', 'right', 'right'),
    $inventory->getDiskDetails($applianceId));

printList('fs_details',
    array('device_name', 'partition_num', 'capacity_mb', 'mount_point'),
    array('left', 'left', 'right', 'left'),
    $inventory->getFSdetails($applianceId));
}
else
{
    print $i18n->get('noInventory') . "<br>\n";
}
}
print $page->toFooterHtml();
$helper->destructor();

// -----
//
// printBlock
//
function printBlock($header, $fields, $values)
{
    global $factory;

    $block = $factory->getPagedBlock($header);
    for ($i=0; $i < sizeof($fields); $i++)
    {
        $block->addFormField($factory->getTextField($i, $values[$fields[$i]], 'r'),
            $factory->getLabel($fields[$i]));
    }
    print $block->toHtml() . "<br>\n";
}

//
// printList
//
function printList($header, $fields, $alignments, $values)
{
    global $factory;

```

```

$sortable = array();
for ($i=0; $i < sizeof($fields); $i++)
{
    $sortable[] = $i;
}
$list = $factory->getScrollList($header, $fields, $sortable);
$list->setAlignments($alignments);
for ($i=0; $i < sizeof($values); $i++)
{
    $textFieldList = array();
    for ($j=0; $j < sizeof($fields); $j++)
    {
        $textFieldList[] = $factory->getTextField('', $values[$i][$fields[$j]], 'r');
    }
    $list->addEntry($textFieldList);
}
print $list->toHtml() . "<br>\n";
}
?>

```

inventorySummary.php

```

<?php

/*
 * inventory.php
 * Copyright 2001, Sun Microsystems. All rights reserved.
 * $Id: inventorySummary.php,v 1.7 2001/08/14 00:26:08 pploquin Exp $
 */

include('mgmtHelper.php');
include('mgmtInventory.php');
include('selectordata.php');

$helper    = new mgmtHelper('base-mgmt-inventory');
$factory   = $helper->getFactory('base-mgmt-inventory', $PHP_SELF);
$page     = $factory->getPage();
$i18n     = $helper->getI18n('base-mgmt-inventory');
$inventory = new Inventory();

print $page->toHeaderHtml();

$updateButton = $factory->getButton('/base/mgmt-inventory/updateInventory.php',
$updateButton');
print $updateButton->toHtml() . "<br>\n";

$inventoryTable = $factory->getScrollList
(
    'inventoryHeader',
    array('ip', 'hostname', 'buildClass', 'numCPUs', 'cpuMHz', 'ramMBs', 'numNICs',
$numDrives', 'totalCapacity', 'detail'),
    array(0, 1, 2, 3, 4, 5, 6, 7, 8)
);
$inventoryTable->setAlignments(array('left', 'left', 'left', 'right', 'right', 'right',
'right', 'right', 'right'));

```

```

$appliances = $inventory->getApplianceSummaries(getdataArray());
for ($i=0; $i < sizeof($appliances); $i++)
{
    if ($appliances[$i]['num_cpus'] > 0)
    {
        $inventoryTable->addEntry(array
        (
            $factory->getTextField('', $appliances[$i]['ip_address'   ], 'r'),
            $factory->getTextField('', $appliances[$i]['host_name'   ], 'r'),
            $factory->getTextField('', $appliances[$i]['class'       ], 'r'),
            $factory->getTextField('', $appliances[$i]['num_cpus'    ], 'r'),
            $factory->getTextField('', $appliances[$i]['max_cpu_mhz' ], 'r'),
            $factory->getTextField('', $appliances[$i]['ram_mb'      ], 'r'),
            $factory->getTextField('', $appliances[$i]['num_nics'    ], 'r'),
            $factory->getTextField('', $appliances[$i]['num_drives'  ], 'r'),
            $factory->getTextField('', $appliances[$i]['total_capacity'], 'r'),
            $factory->getDetailButton("javascript: location='/base/mgmt-inventory/
inventoryDetail.php?applianceId=" . $appliances[$i]['appliance_id'] . "`;
top.code.flow_showNavigation(false)")
        ));
    }
    else
    {
        $inventoryTable->addEntry(array
        (
            $factory->getTextField('', $appliances[$i]['ip_address'   ], 'r'),
            $factory->getTextField('', $appliances[$i]['host_name'   ], 'r'),
            $factory->getTextField('', $appliances[$i]['class'       ], 'r'),
            $factory->getTextField('', 'N/A', 'r')
        ));
    }
}

print $inventoryTable->toHtml();

print $page->toFooterHtml();
$helper->destructor();
?>

```

updateInventory.php

```
<?php

/*
 * updateInventory.php
 * Copyright 2001, Sun Microsystems. All rights reserved.
 * $Id: updateInventory.php,v 1.23 2001/08/24 09:44:17 pploquin Exp $
 */

include('mgmtHelper.php');
$helper    = new mgmtHelper('base-mgmt-inventory');
$factory   = $helper->getFactory();
$page     = $factory->getPage();

print $page->toHeaderHtml();

$progress = $helper->makeProgress($task, 'base-mgmt-inventory');

if (! $progress->hasProgramStarted())
{
    $progress->spawnProgram('[[base-mgmt-inventory.updateInventory]]',
                          '/usr/mgmt/sbin/getInventory.pl',
                          $helper->getSelectorDataArray());

    $progress->setUltimateUrl('/base/mgmt-inventory/inventorySummary.php');
}

$rc = $progress->getProgress('[[base-mgmt-inventory.updateInventory]]', $PHP_SELF);
if($rc < 0)
{
    $helper->msgError('[[base-mgmt-inventory.updateInvDataFail]]');
}
elseif ($rc > 0)
{
    $p_info = $progress->getTaskRecord();
    if ($p_info['status'] == 'F')
    {
        $helper->msgError('[[base-mgmt-inventory.updateInvDataFail]]');
    }
    else if ($p_info['status'] == 'C')
    {
        $helper->msgSuccess('[[base-mgmt-inventory.updateInvDataSuccess]]');
    }
    else
    {
        $helper->msgWarn('[[base-mgmt-inventory.updateInvDataWarned]]');
    }
}

print $page->toFooterHtml();
$helper->destructor();
?>
```

Localization file

mgmt-inventory.po

```
# MAPP tags
#

msgid "mappInventory"
msgstr "Appliance Inventory"

msgid "mappVendor"
msgstr "Sun Microsystems, Inc."

msgid "mappVersion"
msgstr "1.0-27"

#
# Menu
#

msgid "inventory"
msgstr "Appliance Inventory"

msgid "inventory_help"
msgstr "View appliance inventory."

msgid "inventory_summary"
msgstr "Summary"

msgid "inventory_summary_help"
msgstr "Summary inventory data for managed appliances."

msgid "inventory_details"
msgstr "Details"

msgid "inventory_details_help"
msgstr "Inventory details for managed appliances."

msgid "inventory_update"
msgstr "Update"

msgid "inventory_update_help"
msgstr "Update inventory data for managed appliances."

#
# Manual
#

msgid "inventory_manual"
msgstr "Appliance Inventory"

msgid "inventory_manual_help"
msgstr "Click here to view the manual for the [[base-mgmt-inventory.inventory]] [[base-mgmtapp.moduleNameSingle]]."
```

```

# Name of manual

msgid "inventory_manualen"
msgstr "[[base-mgmt-inventory.inventory]] Manual (English)"

#
# Submit button labels
#

msgid "view"
msgstr "View"

#
# Summary page
#
# title of the page on which you select appl

msgid "summarySelect"
msgstr "Select appliances for the inventory summary."

msgid "viewSummaryButton"
msgstr "View summary"

msgid "updateSelectButton"
msgstr "Update inventory"

msgid "inventoryHeader"
msgstr "Managed Appliance Inventory"

msgid "ip"
msgstr "IP Address"

msgid "ip_help"
msgstr "The appliance's IP address"

msgid "hostname"
msgstr "Host Name"

msgid "hostname_help"
msgstr "The appliance's host name"

msgid "buildClass"
msgstr "Class"

msgid "buildClass_help"
msgstr "Appliance type"

msgid "numCPUs"
msgstr "CPU's"

msgid "numCPUs_help"
msgstr "Number of CPUs in the appliance"

msgid "cpuMHz"
msgstr "MHz"

msgid "cpuMHz_help"
msgstr "The CPU clockspeed measured in megahertz."

```

```
msgid "ramMBs"
msgstr "RAM (MB)"

msgid "ramMBs_help"
msgstr "The number of megabytes of memory installed in the appliance."

msgid "numNICs"
msgstr "NIC's"

msgid "numNICs_help"
msgstr "Number of network interface cards installed in the appliance."

msgid "numDrives"
msgstr "Drives"
msgid "numDrives_help"
msgstr "Number of disk drives installed in the appliance."

msgid "totalCapacity"
msgstr "Capacity (GB)"

msgid "totalCapacity_help"
msgstr "The total disk drive capacity measured in gigabytes."

msgid "detail"
msgstr "Detail"

msgid "detail_help"
msgstr "Buttons to view specific appliance details."

#
# Details page
#
# Base Details

msgid "base_details"
msgstr "Appliance Details"

msgid "ip_address"
msgstr "IP Address"

msgid "ip_address_help"
msgstr "The IP Address"

msgid "host_name"
msgstr "Host name"

msgid "host_name_help"
msgstr "The Host name"

msgid "build_class"
msgstr "Build Class"

msgid "build_class_help"
msgstr "The appliance's build class."

msgid "hw_serial_no"
msgstr "Hardware serial number"

msgid "hw_serial_no_help"
msgstr "The appliance's hardware serial number"
```

```
msgid "product_serial_no"
msgstr "Product serial number"

msgid "product_serial_no_help"
msgstr "The appliance's product serial number"

msgid "kernel"
msgstr "Kernel"

msgid "kernel_help"
msgstr "The appliance's operating system kernel name."

msgid "version"
msgstr "Version"

msgid "version_help"
msgstr "Kernel version"

# Memory

msgid "memory_details"
msgstr "System Memory"

msgid "ram_mbytes"
msgstr "Installed RAM (MB)"

msgid "ram_mbytes_help"
msgstr "The amount of RAM installed measured in megabytes"

msgid "swap_mbytes"
msgstr "Swap space (MB)"

msgid "swap_mbytes_help"
msgstr "The amount of swap space measured in megabytes"

# CPUs

msgid "cpu_details"
msgstr "CPU"

msgid "cpu_number"
msgstr "CPU Number"

msgid "cpu_number_help"
msgstr "CPU Number"

msgid "name"
msgstr "Name"

msgid "name_help"
msgstr "The CPU Name"

msgid "model"
msgstr "Model"

msgid "model_id"
msgstr "The CPU Model"
```

```

msgid "cpu_mhz"
msgstr "CPU speed (MHz)"

msgid "cpu_mhz_help"
msgstr "The CPU clock speed measured in megahertz"

msgid "cache_kb"
msgstr "Cache (KB)"

msgid "cache_kb_help"
msgstr "The amount of on-CPU cache measured in kilobytes"

# NICs

msgid "nic_details"
msgstr "Network Interface Cards"

msgid "eth_number"
msgstr "Ethernet number"

msgid "eth_number_help"
msgstr "Ethernet number"

msgid "nic_ip_address"
msgstr "IP Address"

msgid "nic_ip_address_help"
msgstr "IP Address"

msgid "nic_host_name"
msgstr "Host Name"

msgid "nic_host_name_help"
msgstr "The host name associated with this IP address."

msgid "mac_address"
msgstr "MAC Address"

msgid "mac_address_help"
msgstr "MAC Address"

# Disks

msgid "disk_details"
msgstr "Disks"

msgid "device_name"
msgstr "Device name"

msgid "device_name_help"
msgstr "Device name (as found in /dev)"

msgid "num_partitions"
msgstr "Partitions"

msgid "num_partitions_help"
msgstr "Number of partitions on disk."

```

```
msgid "device_capacity"
msgstr "Capacity (MB)"

msgid "device_capacity_help"
msgstr "Disk capacity measured in Megabytes."

# Filesystems

msgid "fs_details"
msgstr "Filesystems"

msgid "partition_num"
msgstr "Partition"
msgid "partition_num_help"
msgstr "The partition number"

msgid "capacity_mb"
msgstr "Capacity (MB)"

msgid "capacity_mb_help"
msgstr "Filesystem capacity measured in megabytes."

msgid "mount_point"
msgstr "Mount Point"

msgid "mount_point_help"
msgstr "Filesystem mount point"

#
# Update
#

msgid "updateButton"
msgstr "Update"

msgid "updateButton_help"
msgstr "Update inventory for these appliances."

msgid "progress"
msgstr "Update progress"

msgid "updateInventory"
msgstr "Inventory update"
```

```

# title of the page on which you select appl

msgid "updateSelect"
msgstr "Select appliances for which to update inventory data."

msgid "updateInventory_help"
msgstr "This is the progress status of the inventory update."

msgid "updateInvDataSuccess"
msgstr "The inventory data for the selected appliance(s) were successfully updated."

msgid "updateInvDataFail"
msgstr "The inventory data for the selected appliance(s) could not be updated be cause an error
occurred."

msgid "updateInvDataWarned"
msgstr "The inventory data for one or more selected appliance(s) could not be updated, see
event log."

#
# Errors
#

msgid "noInventory"
msgstr "No inventory information is available."

#
# Events
#

msgid "noAppliances"
msgstr "No appliances to work on"

msgid "getInventory"
msgstr "Update Inventory"

msgid "gettingInventory"
msgstr "Getting inventory information from [[VAR.ip]]"

msgid "inventoryGetError"
msgstr "Could not get inventory information from [[VAR.ip]] : [[VAR.reason]]"

msgid "inventoryGetSuccess"
msgstr "Successfully got inventory information from [[VAR.ip]]"

msgid "finishedSuccess"
msgstr "Successfully got inventory information"

msgid "finishedWarning"
msgstr "Failed to get inventory information on one or more appliances"

msgid "finishedError"
msgstr "Failed to get inventory information"

```

Back-end section

mgmtInventory.php

```
<?php
/*
 * Inventory DB Library
 * Copyright 2001, Sun Microsystems. All rights reserved.
 * $Id: mgmtInventory.php,v 1.21 2001/10/03 00:00:29 pploquin Exp $
 *
 */

include("Appliance.php");

class Inventory extends Appliance
{
    var $appliances;
    var $numAppliances;
    var $applianceIdArray;

    //
    // Constructor
    //
    function Inventory()
    {
        $this->Appliance();
    }

    //
    // mungeSummaryInfo : merge info into main appliance list
    //                    (for use by getApplianceSummaries())
    //
    function mungeSummaryInfo($selectFields, $fieldCaptions, $table)
    {
        // Build the sql query
        $sql = 'select appliance_id';
        for ($i=0; $i < sizeof($selectFields); $i++)
        {
            $sql .= ', ` . $selectFields[$i] . ` ` . $fieldCaptions[$i];
        }
        $sql .= ' from ` . $table
            . ` where ` . $this->sqlList($this->applianceIdArray, 'appliance_id')
            . ` group by appliance_id'
            . ` order by appliance_id';
    }
}
```

```

$this->query($sql);

$app = 0;
while ($this->next())
{
    while (($app < $this->numAppliances) &&
        ($this->appliances[$app]['appliance_id'] !=
            $this->Record['appliance_id']))
    {
        ++$app;
    }
    if ($app < $this->numAppliances)
    {
        for ($i=0; $i < sizeof($fieldCaptions); $i++)
        {
            $this->appliances[$app][$fieldCaptions[$i]] =
                $this->Record[$fieldCaptions[$i]];
        }
    }
}

//
// getApplianceSummaries
//
function getApplianceSummaries($idArray)
{
    $this->applianceIdArray = $idArray;

    $sql = 'select a.appliance_id'
        . ' , a.ip_address'
        . ' , a.host_name'
        . ' , b.build_class as class'
        . ' from mgmt_appliance as a'
        . ' , mgmt_build as b'
        . ' where a.build_id = b.build_id'
        . ' and a.appliance_id in `';
    $comma = '(';
    for ($i=0; $i < sizeof($this->applianceIdArray); $i++)
    {
        $sql .= $comma . $this->applianceIdArray[$i];
        $comma = ',';
    }
    $sql .= ') order by a.appliance_id';
    $this->query($sql);
    $this->appliances = array();
    while ($this->next())
    {
        $this->appliances[] = $this->Record;
    }
    $this->numAppliances = sizeof($this->appliances);

    $this->mungeSummaryInfo(array('count(*)',
        'max(cpu_mhz)'),
        array('num_cpus',
            'max_cpu_mhz'),
        'inventory_cpu');
}

```

```

$this->mungeSummaryInfo(array('ram_mbytes'),
                        array('ram_mb'),
                        'inventory_memory');

$this->mungeSummaryInfo(array('count(*)'),
                        array('num_nics'),
                        'inventory_nic');

$this->mungeSummaryInfo(array('count(distinct device_name)',
                              'sum(capacity_mb) / 1024'),
                        array('num_drives',
                              'total_capacity'),
                        'inventory_fs');

return $this->appliances;
}

//
// getBaseDetails
//
function getBaseAndMemoryDetails($applianceId)
{
    $sql = 'select a.ip_address'
          . ', a.host_name'
          . ', b.build_class'
          . ', o.kernel'
          . ', o.version'
          . ', if(o.hw_serial_no="",a.mac_address,o.hw_serial_no) as hw_serial_no'
          . ', if(o.product_serial_no="", "N/A",o.product_serial_no) as product_serial_no'
          . ', m.ram_mbytes'
          . ', m.swap_mbytes'
          . ' from mgmt_appliance as a'
          . ', mgmt_build as b'
          . ', inventory_os as o'
          . ', inventory_memory as m'
          . ' where a.build_id = b.build_id'
          . ' and a.appliance_id = o.appliance_id'
          . ' and a.appliance_id = m.appliance_id'
          . ' and a.appliance_id = ' . $applianceId;

    $this->query($sql);

    if ($this->next())
    {
        return $this->Record;
    }

    return 0;
}

//
// getCPUdetails
//
function getCPUdetails($applianceId)
{
    $sql = 'select *'
          . ' from inventory_cpu'
          . ' where appliance_id = ' . $applianceId
          . ' order by cpu_number';
    $this->query($sql);
}

```

```

//
// getListOfRecords
//
function getListOfRecords($sql)
{
    $this->query($sql);
    $recList = array();
    while ($this->next())
    {
        $recList[] = $this->Record;
    }
    return $recList;
}

//
// getNICdetails
//
function getNICdetails($applianceId)
{
    $sql = `select ip_address nic_ip_address
        . ` , host_name nic_host_name`
        . ` , mac_address`
        . ` , eth_number`
        . ` from inventory_nic`
        . ` where appliance_id = ` . $applianceId
        . ` order by eth_number`;

    return $this->getListOfRecords($sql);
}

//
// getDiskDetails
//
function getDiskDetails($applianceId)
{
    $sql = `select device_name`
        . ` , count(*) num_partitions`
        . ` , sum(capacity_mb) device_capacity`
        . ` from inventory_fs`
        . ` where appliance_id = ` . $applianceId
        . ` group by device_name`
        . ` order by device_name`;

    return $this->getListOfRecords($sql);
}

```

```

//
// getFSdetails
//
function getFSdetails($applianceId)
{
    $sql = 'select device_name'
        . ', partition_num'
        . ', capacity_mb'
        . ', mount_point'
        . ' from inventory_fs'
        . ' where appliance_id = ' . $applianceId
        . ' order by device_name, partition_num';

    return $this->getListOfRecords($sql);
}
}

?>

```

inventory.pm

```

#
# Inventory.pm
# Copyright (C) 2001 Sun Microsystems, Inc
# $Id: Inventory.pm,v 1.21 2001/09/04 07:39:45 pploquin Exp $
#
# Functions to call:
# -----
# $rc = Inventory::getInventory($applianceId, $ipAddress)
# $rc = Inventory::removeInventoryInfo($applianceId, [...])
#

package Inventory;

use lib '/usr/mgmt/lib/perl5';
use AgentInterface;
use BDDDB;

my $clientGetInventoryCmd = '/usr/mgmt/bin/inventory_get.pl';

#####
# setInventoryInfo
#
# In : Appliance ID, table, field list (ref), value list (ref)
#
# Out : Return code
#
#####

sub setInventoryInfo
{
    my ($applianceId, $table, $fields) = @_;

    my $sql = 'insert ' . $table
        . ' set appliance_id = ' . $applianceId;
    foreach (keys %{$fields})
    {
        $sql .= ', ' . $_ . ' = "' . $fields->{$_} . '"';
    }
}

```

```

return BDDB::errorCode(EDBERR, 'Error inserting inventory info')
    if (BDDB::runCommand($sql));

return BDDB::errorCode(OKAY);
}

#####
#
# removeInventoryInfo
#
# In : Appliance ID
#
# Out : Return code
#
# Remove all inventory records for specified appliance.
#
#####

sub removeInventoryInfo
{
    return BDDB::errorCode(EPARM, 'Appliance ID is required')
        unless (defined $_[0]);

    my $whereClause = ` where appliance_id in `;
    my $comma = `(`;
    foreach (@_)
    {
        $whereClause .= $comma . $_;
        $comma = `,`;
    }
    $whereClause .= `)`;

    my $stable;
    foreach (qw(os cpu memory nic fs))
    {
        $stable = `inventory_` . $_;
        return BDDB::errorCode(EDBERR, 'Error deleting from ` . $stable)
            if (BDDB::runCommand(`delete from ` . $stable . $whereClause));
    }

    return BDDB::errorCode(OKAY);
}

#####
#
# getInventory
#
# In : Appliance ID, IP address
#
# Out : Return code (0 for success, 1 for AgentInterface failure,
#           -1 for DB / other failure)
#
#####

```

```

sub getInventory
{
  my ($applianceId, $ip) = @_;

  my ($src, $outputStr) = AgentInterface::RunCmdToString($ip, $clientGetInventoryCmd);
  return 1 if ($src);
  if (!$outputStr)
  {
    MgmtUtil::printWarning('Unable to execute inventory fetch on client');
    return 1;
  }

  $src = removeInventoryInfo($applianceId);
  return -1 if ($src);

  my @lines = split(/\n/, $outputStr);
  my (@items, @toks, %fields, $1, $table);
  foreach $1 (@lines)
  {
    %fields = ();
    @items = split(/\|/, $1);
    $table = shift @items;
    foreach (@items)
    {
      @toks = split(/=/, $_);
      $fields{$toks[0]} = $toks[1] || '';
    }
    $src = setInventoryInfo($applianceId, $table, \%fields);
    return -1 if ($src);
  }

  return 0;
}

1;

```

cleanupInventory.pl

```
#!/usr/bin/perl -w

#
# cleanupInventory.pl : get rid of inventory info for specified
#                       appliance id(s)
# Copyright (C) 2001 Sun Microsystems, Inc
# $Id: cleanupInventory.pl,v 1.5 2001/09/04 07:39:45 pploquin Exp $
#

use strict;
use lib '/usr/mgmt/lib/perl5';

if (!defined($ARGV[0]))
{
    print STDERR "Usage: $0 <Appliance Id 1> [Appliance Id 2] ... [Appliance Id n]\n";
    exit 0;
}

use Inventory;
use MgmtUtil;

my $src = Inventory::removeInventoryInfo(@ARGV);
MgmtUtil::printError("Error code = $src " . BDDb::getErrorMessage())
    if ($src);
```

getInventory.pl

```
#!/usr/bin/perl -w

#
# getInventory.pl : Command line utility to get inventory information
#                  about one or more managed appliances.
# Copyright (C) 2001 Sun Microsystems, Inc
# $Id: getInventory.pl,v 1.29 2001/09/04 23:20:42 pploquin Exp $
#

use strict;
use lib '/usr/mgmt/lib/perl5';

if (!defined($ARGV[0]) || (($ARGV[0] eq '-i') ||
    ($ARGV[0] eq '-t')) && !defined($ARGV[1]))
{
    print STDERR<<EOD;
Usage: $0 -i <IP address 1> [IP address 2] ... [IP address n]
      or : $0 [-t <Task Id>] <Appliance Id 1> [Appliance Id 2] ... [Appliance Id n]
EOD
    exit 0;
}

use Inventory;
use Appliance;
use Progress;
use MgmtUtil;
use BDi18n;

BDi18n::setDomain('base-mgmt-inventory');
```

```

my ($src, $applianceList);
my $thisProgressIsAllMine = 0;

if ($ARGV[0] ne '-i')
{
    Progress::initProgress(\@ARGV);
    Progress::setProcessName(BDi18n::getMsg('getInventory'));
    $thisProgressIsAllMine = 1 if (!Progress::currentlyDone());

    ($src, $applianceList) = Appliance::getApplianceIPAddressList(@ARGV);
    if ($src || !$applianceList)
    {
        Progress::progressFailed(BDi18n::getMsg('noAppliances'))
            if ($thisProgressIsAllMine);
        exit -1;
    }
}
else
{
    shift @ARGV; # waste the -i

    ($src, $applianceList) = Appliance::getAppliancesByIpAddress(@ARGV);
    MgmtUtil::printError('Database error') if ($src);
    MgmtUtil::printError('No appliances under management in given set')
        if (!$applianceList);
}

my $success = 0;
my $total = 0;
my $msg = '';
foreach my $appliance (@{$applianceList})
{
    ++$total;
    Progress::setMessage(BDi18n::getMsg('gettingInventory',
        { 'ip' => $appliance->{'ip_address'} }));
    $src = Inventory::getInventory($appliance->{'appliance_id'}, $appliance->{'ip_address'});
    if ($src == 1)
    {
        $msg = BDi18n::getMsg('inventoryGetError',
            { 'ip' => $appliance->{'ip_address'},
              'reason' => MgmtUtil::getWarning() });
        Progress::warningEvent($msg);
    }
    elsif ($src == -1)
    {
        $msg = BDi18n::getMsg('inventoryGetError',
            { 'ip' => $appliance->{'ip_address'},
              'reason' => BDDb::getErrorMessage() });
        Progress::errorEvent($msg);
    }
    else
    {
        ++$success;
        $msg = BDi18n::getMsg('inventoryGetSuccess',
            { 'ip' => $appliance->{'ip_address'} });
        Progress::infoEvent($msg);
    }
    Progress::updateProgress($msg);
}

```

```

if ($thisProgressIsAllMine)
{
  if ($success == 0)
  {
    Progress::progressFailed(BDi18n::getMsg('finishedError'));
  }
  elseif ($success == $total)
  {
    Progress::progressComplete(BDi18n::getMsg('finishedSuccess'));
  }
  else
  {
    Progress::progressWarned(BDi18n::getMsg('finishedWarning'));
  }
}

```

140_inventory_tables.sql

```

#
# Create the inventory tables
# Copyright (C) 2001 Sun Microsystems, Inc
# $Id: 140_inventory_tables.sql,v 1.7 2001/10/03 00:00:29 pploquin Exp $
#
#
# OS
#
create table if not exists inventory_os
(
  os_id                mediumint          not null auto_increment,
  appliance_id         mediumint          not null,
  kernel               varchar(64)        not null,
  version              varchar(64)        not null,
  hw_serial_no         varchar(32)        not null,
  product_serial_no    varchar(32)        not null,
  index (os_id),
  index (appliance_id)
);
#
# CPU
#
create table if not exists inventory_cpu
(
  cpu_id               mediumint          not null auto_increment,
  appliance_id         mediumint          not null,
  cpu_number           tinyint           not null,
  name                 varchar(128)       not null,
  model                varchar(128)       not null,
  cpu_mhz              smallint          not null,
  cache_kb             smallint          not null,
  index (cpu_id),
  index (appliance_id)
);

```

```

#
# Memory
#
create table if not exists inventory_memory
(
    memory_id          mediumint          not null auto_increment,
    appliance_id       mediumint          not null,
    ram_mbytes         smallint           not null,
    swap_mbytes        smallint           not null,
    index (memory_id),
    index (appliance_id)
);

#
# NIC
#
create table if not exists inventory_nic
(
    nic_id             mediumint          not null auto_increment,
    appliance_id       mediumint          not null,
    eth_number         tinyint           not null,
    ip_address         char(16)          not null,
    mac_address        char(20)          not null,
    host_name          varchar(128)       not null,
    index (nic_id),
    index (appliance_id)
);

#
# Filesystem
#
create table if not exists inventory_fs
(
    fs_id              mediumint          not null auto_increment,
    appliance_id       mediumint          not null,
    device_name        varchar(16)        not null,
    partition_num      tinyint           not null,
    capacity_mb        mediumint          not null,
    mount_point        varchar(64)        not null,
    index (fs_id),
    index (appliance_id)
);

```

Makefile

```
# specify the following variables:
VENDOR=base

# if the VENDOR field is an alias, this should be the real name.
# otherwise, set it to VENDOR.
VENDORNAME = cobalt
SERVICE   = mgmt-inventory
VERSION    = 1.0
RELEASE    = 21

# add a buildarch if desired
BUILDDARCH=noarch

# locale exclude pattern
#XLOCALEPAT=
EXTRAEXCLUDES = --exclude client

BUILDUI=yes
BUILDGLUE=yes
BUILDDLOCALE=yes
BUILDSRC=no

#####
# some useful defines
INSTALL=install
INSTALL_BIN=$(INSTALL) -m 755
INSTALL_OTH=$(INSTALL) -m 644
TMPDIR=/tmp

include /usr/sausalito/devel/module.mk
MGMT_ROOT = /usr/mgmt

if [ -e devel/mgmt.mk ]; then \
include devel/mgmt.mk ; \
else \
echo "Using default location... "; \
MGMT_ROOT = /usr/mgmt ; \
fi; \

#
# install
#

install:
@echo "installing lib..."
mkdir -p $(MGMT_ROOT)/lib
find lib -type f -not -path "*CVS*" -exec install {} -D -o root -g root -m 644
$(PREFIX)$(MGMT_ROOT)/{} \;
@echo "installing sbin..."
mkdir -p $(MGMT_ROOT)/sbin
find sbin -type f -not -path "*CVS*" -exec install {} -D -o root -g root -m 755
$(PREFIX)$(MGMT_ROOT)/{} \;
/usr/mgmt/sbin/runsql 140_inventory_tables.sql
find manuals -type f -not -path "*CVS*" -exec install {} -D \
-o root -g root -m 644 $(CEDIR)/ui/web/base/$(SERVICE)/{} \;
```

clientrpms:

```
    cd client/i386 ; make rpm
cd client/mips ; make rpm

#
# clean
#

cleanit : clean
cd client/i386 ; make clean
cd client/mips ; make clean

#
# mapp
#

MAPP_NAME = base-mgmt-inventory
MAPP_TMP = $(TMPDIR)/$(MAPP_NAME)

mapp: rpm clientrpms
mkdir -p $(MAPP_TMP)/mgmtStation
mv client/i386/rpms/*.rpm $(MAPP_TMP)
mv client/mips/rpms/*.rpm $(MAPP_TMP)
cp as_rpms/*.rpm $(MAPP_TMP)/mgmtStation
cp packing_list.xml $(MAPP_TMP)
cd $(MAPP_TMP); \
tar zcvf $(MAPP_NAME)-`grep "<version " packing_list.xml | awk -F\" {'print $$2'}`.mapp * ; \
cd -
mv $(MAPP_TMP)/*.mapp .
rm -rf $(MAPP_TMP)
make clean
```

Target appliance

inventory_get.pl

```
#!/usr/bin/perl -w

#
# Copyright (C) 2001 Sun Microsystems, Inc
# $Id: inventory_get.pl,v 1.7 2001/10/03 00:00:29 pploquin Exp $
#

use strict;
use Socket;    # For host name lookups

my %inventoryInfo =
(
  'os' =>
  {
    'cmd'    => '/bin/uname -sr',
    'parser' => \&parseOSInfo,
    'table'  => 'inventory_os',
  },
  'cpu' =>
  {
    'cmd'    => '/bin/cat /proc/cpuinfo',
    'parser' => \&parseCPUInfo,
    'table'  => 'inventory_cpu',
  },
  'mem' =>
  {
    'cmd'    => '/bin/ls -l /proc/kcore ; /bin/cat /var/log/dmesg',
    'parser' => \&parseMemInfo,
    'table'  => 'inventory_memory',
  },
  'nic' =>
  {
    'cmd'    => '/sbin/ifconfig -a',
    'parser' => \&parseNICInfo,
    'table'  => 'inventory_nic',
  },
  'fs' =>
  {
    'cmd'    => '/bin/df -m',
    'parser' => \&parseFSInfo,
    'table'  => 'inventory_fs',
  }
);
```

```

#-----
#                               Parsing functions
#-----

my @lines;

#
# parseOSinfo
#
sub parseOSinfo
{
    $_ = $lines[0];
    my %osInfo;
    ($osInfo{'kernel'}, $osInfo{'version'}) = split /\s+//;

    return [ \%osInfo ];
}

#
# parseCPUinfo
#
sub parseCPUinfo
{
    my %useFields =
    (
        'cpu'          => 'name',
        'cpu model'    => 'model',
        'BogoMIPS'     => 'cpu_mhz'
    );

    my @toks;
    my %newCPU;
    $newCPU{'cpu_number'} = 0; # No such thing as a dual CPU MIPS product
    foreach (keys %useFields)
    {
        $newCPU{$useFields{$_}} = 0;
    }

    foreach (@lines)
    {
        @toks = split /:/:;
        foreach (@toks)
        {
            $_ =~ s/^\s+//;
            $_ =~ s/\s+$//;
        }
        if (exists($useFields{$toks[0]}))
        {
            $newCPU{$useFields{$toks[0]}} = $toks[1];
        }
    }

    # Round off CPU MHz
    $newCPU{'cpu_mhz'} = int($newCPU{'cpu_mhz'});
    $newCPU{'cache_kb'} = 0;

    return [ \%newCPU ];
}

```

```

#
# parseMemInfo
#
sub parseMemInfo
{
    my %memInfo;

    $_ = shift @lines;
    my @t = split(/\s+/, $_);
    $memInfo{'ram_mbytes'} = int($t[4] / 1024 / 1024);

    foreach (@lines)
    {
        if ($_ =~ /^Adding Swap:/)
        {
            $_ =~ /Adding Swap: ([0-9]+)k/;
            $memInfo{'swap_mbytes'} = int($1 / 1024);
        }
    }

    return [%memInfo];
}

#
# parseNICinfo
#
sub parseNICinfo
{
    my @nicList = ();

    my ($i, $ethNum);
    for ($i=0; $i < $#lines; $i++)
    {
        $_ = $lines[$i];
        if ($_ =~ /^eth([0-9]+)/)
        {
            $ethNum = $1;
            next if ($_ =~ /^eth[0-9]+:([0-9]+)/); # VSITE
            my %newNIC;
            $newNIC{'eth_number'} = $1;
            $_ = /\S+\s+Link encap:Ethernet\s+HWaddr\s+(\S+)/;
            $newNIC{'mac_address'} = $1;
            $_ = $lines[++$i];
            chomp;
            if ($_ =~ /inet addr/)
            {
                {
                    $_ = /\s+inet addr:(\S+)\s+Bcast:(\S+)\s+Mask:(\S+)/;
                    $newNIC{'ip_address'} = $1;
                    # $newNIC{'bcast'} = $2;
                    # $newNIC{'mask'} = $3;
                }
            }
            else
            {
                {
                    $newNIC{'ip_address'} = '0.0.0.0';
                }
            }
            $newNIC{'host_name'} = gethostbyaddr(inet_aton($newNIC{'ip_address'}), AF_INET) || '';
            push (@nicList, \%newNIC);
        }
    }
}

```

```

    return \@nicList;
}

#
# parseFSInfo
#
sub parseFSInfo
{
    my @fsList = ();
    my $i;
    for ($i=1; $i <= $#lines; $i++)
    {
        $_ = $lines[$i];
        if ($_ =~ /^\/dev/)
        {
            $_ = /\/dev\/([a-z]+)([0-9]+)\s+([0-9]+)\s+([0-9]+)\s+([0-9]+)\s+([0-9]+)\s+(\S+)/;
            my %fs;
            $fs{'device_name'} = $1;
            $fs{'partition_num'} = $2;
            $fs{'capacity_mb'} = $3;
            $fs{'mount_point'} = $7;
            push (@fsList, \%fs);
        }
    }

    return \@fsList;
}

#-----

#
# Do it
#
my ($inv, $outputStr, $resultList, $res);
foreach $inv (keys %inventoryInfo)
{
    $outputStr = `${inventoryInfo}{$inv}->{'cmd'}`;
    if ($outputStr)
    {
        @lines = split(/\n/, $outputStr);
        $resultList = &${inventoryInfo}{$inv}->{'parser'};
        foreach $res (@{$resultList})
        {
            print $inventoryInfo{$inv}->{'table'};
            foreach (keys %{$res})
            {
                print '|' . $_ . '=' . $res->{$_};
            }
            print "\n";
        }
    }
}

```

i386-based appliances

Makefile

```
#
# makefile for i386 inventory gathering scripts
#

ROOT      = $(PREFIX)
BINDIR    = $(ROOT)/usr/mgmt/bin
TARFILE   = base-mgmt-inventory-i386-scripts.tar.gz
SPECFILE  = base-mgmt-inventory-i386-scripts.spec

INSTALL   = install
SCRIPTS   = inventory_get.pl

all:
@echo Nothing to do.

install: install-all

install-all: $(SCRIPTS)
$(INSTALL) -d $(BINDIR)
$(INSTALL) -m 755 $(SCRIPTS) $(BINDIR)

rpm: rpm-base-mgmt-inventory-i386-scripts

rpm-base-mgmt-inventory-i386-scripts:
cd ../;tar -zcvf $(TARFILE) i386 &> /dev/null; \
cp $(TARFILE) /usr/src/redhat/SOURCES; \
cp i386/$(SPECFILE) /usr/src/redhat/SPECS; \
rpm -ba /usr/src/redhat/SPECS/$(SPECFILE);
mkdir -p rpms; cp /usr/src/redhat/RPMS/noarch/base-mgmt-inventory-i386* rpms

clean:
rm -f ../$(TARFILE)
rm -rf rpms
```

base-mgmt-inventory-i386-script.spec

```
Summary: Perl scripts to get inventory info on i386 based products
Name: base-mgmt-inventory-i386-scripts
version: 1.0
Release: 6
Copyright: Sun Microsystems
Group: Applications/System
Source: base-mgmt-inventory-i386-scripts.tar.gz
BuildRoot: /tmp/base-mgmt-inventory-i386
Requires: perl
BuildArchitectures: noarch

%prep
%setup -n i386

%build
rm -rf $RPM_BUILD_ROOT
make all

%install
make PREFIX=$RPM_BUILD_ROOT install
```

```
%files
/usr/mgmt/bin/*

%description
This rpm contains the client specific inventory gathering scripts for the
i386 based server appliances.

%changelog
* Sat Apr 14 2001 Phil Ploquin <phil.ploquin@sun.com>
- Inital Version
```

mips-based appliances

Makefile

```
#
# makefile for mips inventory gathering scripts
#

ROOT      = $(PREFIX)
BINDIR    = $(ROOT)/usr/mgmt/bin
TARFILE   = base-mgmt-inventory-mips-scripts.tar.gz
SPECFILE  = base-mgmt-inventory-mips-scripts.spec

INSTALL   = install
SCRIPTS   = inventory_get.pl

all:
@echo Nothing to do.

install: install-all

install-all: $(SCRIPTS)
$(INSTALL) -d $(BINDIR)
$(INSTALL) -m 755 $(SCRIPTS) $(BINDIR)

rpm: rpm-base-mgmt-inventory-mips-scripts

rpm-base-mgmt-inventory-mips-scripts:
cd ../;tar -zcvf $(TARFILE) mips &> /dev/null; \
cp $(TARFILE) /usr/src/redhat/SOURCES; \
cp mips/$(SPECFILE) /usr/src/redhat/SPECS; \
rpm -ba /usr/src/redhat/SPECS/$(SPECFILE);
mkdir -p rpms; cp /usr/src/redhat/RPMS/noarch/base-mgmt-inventory-mips* rpms

clean:
rm -f ../$(TARFILE)
rm -rf rpms
```

base-mgmt-inventory-mips-script.spec

Summary: Perl scripts to get inventory info on mips based products

Name: base-mgmt-inventory-mips-scripts

version: 1.0

Release: 6

Copyright: Sun Microsystems

Group: Applications/System

Source: base-mgmt-inventory-mips-scripts.tar.gz

BuildRoot: /tmp/base-mgmt-inventory-mips

Requires: perl

BuildArchitectures: noarch

%prep

%setup -n mips

%build

rm -rf \$RPM_BUILD_ROOT

make all

%install

make PREFIX=\$RPM_BUILD_ROOT install

%files

/usr/mgmt/bin/*

%description

This rpm contains the client specific inventory gathering scripts for the mips based server appliances.

%changelog

* Sun Apr 15 2001 Phil Ploquin <phil.ploquin@sun.com>

- Initial Version

Network Tool Module

This chapter provides the files in the Network Tool control module. These files are described in Chapter 2 and are presented here in the same order as in Chapter 2.

The Network Tool module consists of software code installed only on the Sun Cobalt™ Control Station. There is no software code to install on the target appliance.

Chapter 2, “Module Files”, explains how to build a control module for the control station.

Control Station

The code for both the back-end section and the graphical-user-interface (GUI) section is contained in one file.

networktool.php

```
<?php

/*
 * status.php
 * Copyright 2001, Sun Microsystems. All rights reserved.
 * $Id: status.php,v 1.6 2001/08/03 18:44:02 ahlwoon Exp $
 */

include('mgmtHelper.php');

$helper = new mgmtHelper('base-mgmt-import', $PHP_SELF);
$factory = $helper->getFactory();
$i18n = $helper->getMyi18n();
$page = $factory->getPage();

print $page->toHeaderHtml();

$backButton = $factory->getBackButton("$PHP_SELF");
$tmpfile = "/var/tmp/networklog.out";

switch ($action) {

    case "ping":

        // get helpful info about the appliances
        $sw = $helper->getApplianceDB(array("All"), $helper->getSelectorDataArray(), array());
        $appliances = $sw->getAppliances(); // "dataApplianceList";
        $title = "Ping " . count($appliances) . " Appliances";
        $block = $factory->getSimpleBlock($title);
        //do ping for each appliance, write to file, read file, output data more elegantly

        foreach($appliances as $record) {
            $host = $record['ip_address'];
            $name = $record['name'];
            $buffer = "\n";

            system("ping -c 5 $host > $tmpfile");
```

```

    $fp = fopen ($tmpfile, "r");
    while (!feof ($fp)) {
        $buffer .= fgets($fp, 4096);
    }
    $buffer .= "\n";
    fclose ($fp);
    unlink ($tmpfile);

    $block->addHtmlComponent( $factory->getTextBlock("", $buffer, "r" ) );
}

$block->addButton($backButton);
print $block->toHtml();

break;

case "traceroute":

// get helpful info about the appliances
$sw = $helper->getApplianceDB(array("All"), $helper->getSelectorDataArray(), array());
$appliances = $sw->getAppliances(); // "dataApplianceList";

$title = "Traceroute " . count($appliances) . " Appliances";
$block = $factory->getSimpleBlock($title);

// do traceroute for each appliance, write to file, read file, output data more elegantly
foreach($appliances as $record) {
    $host = $record['ip_address'];
    $name = $record['name'];
    $buffer = "\ntraceroute to $name ($host), 30 hops max, 38 byte packets\n";

    system("traceroute $host > $tmpfile");

    $fp = fopen ($tmpfile, "r");
    while (!feof ($fp)) {
        $buffer .= fgets($fp, 4096);
    }
    $buffer .= "\n";
    fclose ($fp);
    unlink ($tmpfile);
    $block->addHtmlComponent( $factory->getTextBlock("", $buffer, "r" ) );
}

$block->addButton($backButton);
print $block->toHtml();

break;

case "portscan":

$block = $factory->getSimpleBlock("Port Scan!");
$buffer = "Port scan feature coming soon. Please try again at a later time.";
$block->addHtmlComponent( $factory->getTextBlock("", $buffer, "r" ) );

$block->addButton($backButton);
print $block->toHtml();

break;

```

```

case "runcmd":

// get helpful info about the appliances
$sw = $helper->getApplianceDB(array("All"), $helper->getSelectorDataArray(), array());
$appliances = $sw->getAppliances(); //"dataApplianceList");

$title = "Run command on " . count($appliances) . " Appliances";
$block = $factory->getPagedBlock($title);
$block->setColumnWidths(array("25%", "75%"));

$buffer = "Caution! Please use this feature with extreme care.\n";
$buffer .= "Enter command to be run on the following appliances:\n";
$block->addFormField( $factory->getTextBlock("", $buffer, "r") );

// print out each appliance for which runcmd is applied
unset($buffer);
foreach($appliances as $record) {
    $name = "\t" . $record['name'];
    $buffer .= $name . "\n";
}

$block->addFormField( $factory->getTextBlock("", $buffer, "r") );

$cmdblock = $factory->getTextField("runcmd", $runcmd);
$cmdblock->setSize(50);
$cmdblock->setMaxLength(256);
$block->addFormField( $cmdblock, $factory->getLabel("Run Command") );

$block->addFormField( $factory->getTextField("action", $action="execute", "") );
// $block->addFormField( $factory->getTextField("appliances", $appliances, "") );

$block->addButton( $factory->getButton($page->getSubmitAction(), "Execute") );
$block->addButton( $factory->getCancelButton("$PHP_SELF") );
print $block->toHtml();

break;

case "execute":

// get helpful info about the appliances
$sw = $helper->getApplianceDB(array("All"), $helper->getSelectorDataArray(), array());
$appliances = $sw->getAppliances(); //"dataApplianceList");

$title = "Run command on " . count($appliances) . " Appliances Completed";
$block = $factory->getPagedBlock($title);
$block->setColumnWidths(array("25%", "75%"));

// print out each appliance for which runcmd is applied
unset($buffer);
foreach($appliances as $record) {
    $name = "\t" . $record['name'];
    $buffer .= $name . "\n";
}

$block->addFormField( $factory->getTextBlock("", $buffer, "r") );

unset($buffer);

system("$runcmd > $tmpfile");

```

```

$fp = fopen ($tmpfile, "r");
while (!feof ($fp)) {
    $buffer .= fgets($fp, 4096);
}

$buffer .= "\n";
fclose ($fp);
unlink ($tmpfile);

$block->addFormField( $factory->getTextBlock("", $buffer, "r") );

print $block->toHtml();

print "<P>" . $backButton->toHtml();

break;

default:
// urls for the 3 applet buttons
$url_ping = "$PHP_SELF?action=ping";
$url_traceroute = "$PHP_SELF?action=traceroute";
$url_portscan = "$PHP_SELF?action=portscan";
$url_runcmd = "$PHP_SELF?action=runcmd";

$block = $factory->getPageBlock("Check Network Host Info");
print $block->toHtml();

$helper->makeSelector(array($url_ping, $url_traceroute, $url_portscan, $url_runcmd),
array("ping", "traceroute", "portscan", "runcmd"), "3", "", "",
"base-appmgr", "getAppliancesToView", "Appliance", "",
array(array("All")));
}

print $page->toFooterHtml();
$helper->destructor();
?>

```

Licenses

The BSD Copyright

Copyright ©1991, 1992, 1993, 1994 The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by the University of California, Berkeley and its contributors.
4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program,” below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification.”) Each licensee is addressed as “you.”

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above, provided that you also do one of the following:

- a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated, so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING, THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT, UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING, WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

SSL License

Copyright (c) 1998-1999 Ralf S. Engelschall. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment:
“This product includes software developed by Ralf S. Engelschall <rse@engelschall.com> for use in the mod_ssl project (http://www.engelschall.com/sw/mod_ssl/).”
4. The name “mod_ssl” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact rse@engelschall.com.
5. Products derived from this software may not be called “mod_ssl” nor may “mod_ssl” appear in their names without prior written permission of Ralf S. Engelschall.
6. Redistributions of any form whatsoever must retain the following acknowledgment:
“This product includes software developed by Ralf S. Engelschall <rse@engelschall.com> for use in the mod_ssl project (http://www.engelschall.com/sw/mod_ssl/).”

THIS SOFTWARE IS PROVIDED BY RALF S. ENGELSCHALL “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL RALF S. ENGELSCHALL OR HIS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Acme Public License

Copyright (c) 2000 by Jef Poskanzer <jef@acme.com>. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.