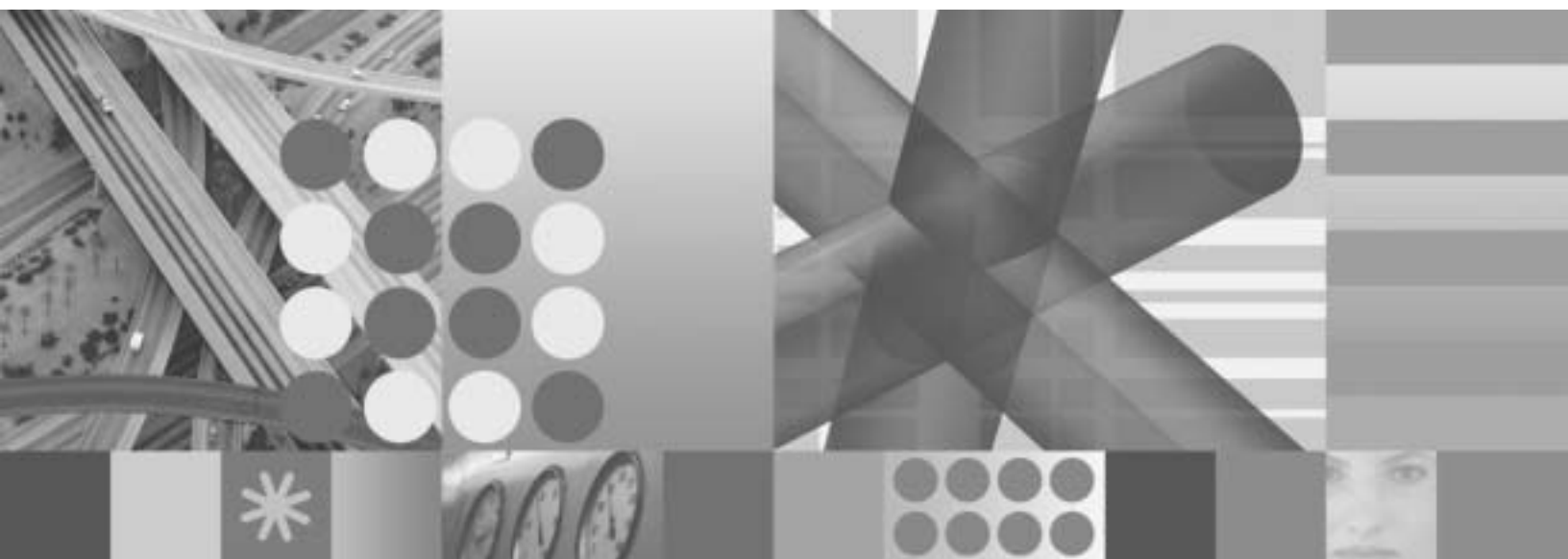


Version 4.3.1



User's Guide for Deployment Services

Version 4.3.1



User's Guide for Deployment Services

Note

Before using this information and the product it supports, read the information in "Notices" on page 449.

This edition applies to version 4, release 3, modification level 1 of IBM Tivoli Configuration Manager (program number 5724-C06) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC23-4710-04.

© **Copyright International Business Machines Corporation 2000, 2008.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	ix
--------------------------	-----------

Tables	xi
-------------------------	-----------

About This Guide	xiii
-----------------------------------	-------------

Who Should Read This Guide	xiii
What This Guide Contains	xiii
Publications	xv
IBM Tivoli Configuration Manager Library	xv
Related Publications	xvi
Accessing Publications Online.	xvii
Ordering Publications	xvii
Accessibility	xvii
Tivoli Technical Training	xviii
Support Information.	xviii
Conventions Used In This Guide	xviii
Typeface Conventions	xviii
Operating System-dependent Variables and Paths	xix
Using the Command Line	xix
Command Line Syntax	xix
Getting Help on Commands	xxi

Part 1. Planning, Scheduling, and Monitoring Activities 1

Chapter 1. Using Activity Planner. 5

Before You Start	5
Understanding the Activity Planner Environment	5
Activity Planner Concepts	6
Activity Planner Roles	6
Enabling IBM Tivoli Configuration Manager Components for the Activity Planner	7
The Activity Planner Configuration File	8
Defining the Activity Planner Engine Parameters	9
The apmJNL.ini File.	14
Activity Planner Server Variables	15
Activity Planner Command Line Variable	15
Using the Activity Plan Editor and the Activity Plan Monitor	16

Chapter 2. Performing Activity Planner Operations 19

Launching the Activity Planner GUIs.	19
Defining an Activity	20
Task Library Tasks	22
Defining a Task Library Task	22
Software Distribution Operations	22
Defining a Software Distribution Activity	23
Inventory Operations	24
Defining an Inventory Activity	24
Pristine Manager Operations	24
Defining a Pristine Manager Activity	25

Using Variables	25
Selecting Targets for Activities and Activity Plans.	25
Selecting Targets for an Activity	26
Excluding Targets from an Activity	27
Specifying a List of Target Names	27
Specifying a File Name Containing a List of Targets	28
Specifying an Inventory Subscriber as a Target	28
Specifying a Query Library Subscriber as a Target	29
Specifying a Directory Query Library Subscriber as a Target	29
Specifying a Profile Subscriber as a Target	29
Specifying a Pristine Target Subscriber	31
Using Variables to Specify Targets	32
Conditioning the Execution of Activities.	33
Sorting Activities in Order of Execution	36
Scheduling the Execution of Activities	36
Defining the General Properties of the Activity Plan	37
Scheduling the Activity Plan.	38
Scheduling Plans to Execute within a Time Interval.	38
Scheduling Activity Plans to Run Periodically	40
Interrupting the Recursion of an Activity Plan	41
Selecting Targets for an Activity Plan	42
Loading an Activity Plan	43
Saving the Activity Plan	43
Deleting Activity Plans or Activities	44
Submitting and Monitoring Activity Plans	44
Activity Plan Status	45
Filtering Activity Plans, Targets, or Activities	47
Submitting an Activity Plan	49
Defining the Variable Values.	50
Updating a Submitted Plan	50
Stopping the Recursion of a Submitted Activity Plan	51
Controlling the Execution of Activity Plans and Activities	51
Generating a Recovery Plan	52
Launching the Distribution Status Console	53
Updating the Status of an Activity Plan	53
Deleting Completed Activity Plans	53
Cleaning Up the Database	54

Chapter 3. Using the Command Line 57

Activity Plan Definition File	57
Setting Frequency Information	62
Defining Variables with the Variable Subelement	63
Defining Activities in the Activity Plan Definition File	63
The Execution Window	65
Using Conditions	66
Defining Targets.	68
Supported Task Library Activities	69
Parameters of Task Library Operations	69

Parameter Values for Task Library Operations	70
Supported Software Distribution Activities	70
Parameters of Software Distribution Operations	71
Parameter Values for Software Distribution Operations	80
Supported Inventory Activities	84
Parameters of Inventory Operations	84
Parameter Values for the Scan Operation	85
Supported Pristine Manager Activities	87
Parameter of Pristine Operations	87
Parameter Values for the Pristine Installation Operation	87
Managing Activity Plans	88
wapmfltr	91
wapmgui	94
wapmplugin	95
wapmrim	97
wcntpln	98
wdelpln	100
wdelstat	101
wexppln	102
wgenpln	103
wgeterrlev	106
wimppln	107
wlstpln	108
wmonpln	110
wresetlev	113
wsetapmpw	114
wseterrlev	115
wsfdpln	117
wstartapm	120
wstopapm	121
wsubpln	122
wunlockpln	126
wupdpn	127
Return Values	130

Chapter 4. Troubleshooting 131

Activity Planner Logs and Traces	131
wtrcapm	132
Main Activity Planner Log	135
Activity Plan Monitor and Editor Startup Trace Files	135
Activity Planner Core Trace	135
Activity Planner Main Trace	136
Activity Planner Executer Trace	136
Activity Planner Handler Trace	136
Activity Planner Command Line Interface Trace	136
Activity Plan Monitor Trace	137
Activity Planner Default Trace	137
Downloading Plug-ins	137
Managing DB2 Deadlock	138
Managing Different Domains	139
Activity Planner Installation Notes	139
Managing Linux Versions	140
Using X sessions	140
Checkpoint restart mechanism	140
Specific Problems and Workarounds	142

Part 2. Modeling the Enterprise Configuration 145

Chapter 5. Using Change Manager 147

Before You Start	147
Change Manager Concepts	148
Reference Model Overview	148
Configuration Element Overview	148
Subscribers	149
Differencing Mechanism	151
Understanding the Change Manager Environment	151
Change Manager Roles	152
Enabling Configuration Manager Components for Change Manager	153
The Change Manager Configuration File confccm.xml	153
The settings.xml file	155
The stable.xml File	155
The invtable.xml File	160
The ostable.xml File	160
Reference Models	161
Reference Model Example	161
Representing a Reference Model in XML Format	163

Chapter 6. Performing Change Manager Operations 171

Launching the Change Manager GUI	171
Using Change Manager	171
Creating the Reference Model Structure	172
Creating a Reference Model from a Target	173
Importing a Reference Model	175
Exporting a Reference Model in XML Format	175
Adding Elements to a Reference Model	176
Adding an Inventory Data Element	176
Adding an Inventory Scan Element	177
Adding an Operating System Element	177
Adding a Software Distribution Element	178
Assigning Subscribers to a Reference Model	180
Assigning an Inventory Subscriber	181
Assigning a Profile Manager Subscriber	182
Assigning a CSV Subscriber	182
Assigning a Static Subscriber	183
Assigning Query Library Subscribers	183
Assigning Directory Query Library Subscribers	184
Assigning Pristine Manager Subscribers	185
Modifying a Reference Model	186
Change Manager and Activity Plans	188
Previewing an Activity Plan	189
Assigning a Priority to Configuration Elements	189
Submitting an Activity Plan	191
Creating Change Manager Reports	192
Reference Model Report	193
Target Report	194

Chapter 7. Using the Command Line 195

Managing Reference Models	195
wccmgui	197

wccmplugin	198
wdelrmod	200
wexprmod	202
wimprmod	204
wlstrmod	206
wlstrep	208
wsyncrmod	210
Chapter 8. Troubleshooting	213
Change Manager Logs	213
Change Manager Traces	213
Checking the Change Manager Configuration File	215
Change Manager GUI Trace	216
wtrccm	217
Managing Different Domains	219
Distributing to a Large Number of Targets	219
Specific Problems and Workarounds	219
<hr/>	
Part 3. Installing an Operating System on Pristine Machines	221
<hr/>	
Chapter 9. Overview: Installing an Operating System on Pristine Machines	223
Advantages of Using Pristine Manager	223
Pristine Manager Concepts	223
Prerequisites	224
Configuring Pristine Manager	224
Enabling Pristine Manager Databases	224
Configuring Pristine Manager for Tivoli Enterprise Console	226
Enabling Integration with the Tivoli Enterprise Console	226
Installing on Pristine Machines in Interconnected Regions	230
Chapter 10. Installing on Pristine Machines	233
Setup Tasks	233
Defining Servers	233
Defining Machines	236
Filtering Machines in the List	242
Adding Machines to a Group	243
Updating Multiple Machines	243
Creating Groups	245
Creating an Operating System Element	248
Defining Environment Variables	252
Defining Your Own Environment Variables	254
Running Commands on Pristine Machines	256
Preparing and Submitting a Plan	256
Using Activity Planner	256
Using Change Manager	257
Chapter 11. Using the Command Line	259
woselement	260
wpristine	262
wpristineexport	264
wpristinegroup	265
wpristinemachine	267

wpristinesrv	270
------------------------	-----

Chapter 12. Troubleshooting 273

Gathering Trace Information	273
PRISTINE_TMA_SETUP_PATH and Sharing Access to Endpoint Setup Files	274
Installing from a RIS Server without Sharing the Folder	274
Installing from an ADS Server without Sharing the Folder	274
Adding Software Distribution or Inventory Activities to a Plan	276
Configuration Changes after Installing Pristine Manager	277
Failure of Installation	277
Avoiding Failure	277
ADS Certificate	277
Installing in Interconnected Regions	277
Installing Microsoft Windows 2003 from an ADS Server	278
Installation from RIS Server Successful but Activity Plan Fails	278
Specifying a Static IP Address	278

Part 4. Web Interface 279

Chapter 13. Administering the Web Interface 281

Setting up the Web Interface	281
Adding the WebUI_Admin Role	281
Registering Plug-ins	282
wwebplugin	283
Enabling Java Plug-in Automatic Download	285
Making Web Objects Available	285
Publishing and Unpublishing Web Objects	286
wweb	288
Unrestricted Access to Web Objects	292
Determining What Has Been Published	292
Example of PUBLISHED_PACKAGES_QUERY Output	293
Example of PUBLISHED_REFMODS_QUERY Output	294
Providing Information and Assistance for Web Interface Users	294
Configuring the Web Interface Window	295
Enabling User Context Switch	296
Monitoring the Results of Operations	298
Troubleshooting	300
Tivoli Web Gateway Logs and Traces for the Web Interface	300
Configuring Trace Settings	301
Configuring Trace Parameters	302
wwebcfg	303
Troubleshooting	305
Administrator on the Tivoli Server	305
Web Interface User	306

Chapter 14. Using the Web Interface 309

Starting the Web Interface	309
--------------------------------------	-----

Web Interface Window	311
Operations Console	311
Using Web Objects	312
Software Packages.	313
Installing Software Packages	313
Verifying Software Packages	314
Uninstalling Software Packages	315
Running Inventory Scans	316
Reference Models	317
Viewing Reference Models	317
Applying (Synchronizing) Reference Models	319
Troubleshooting	321

Part 5. Managing Resources . . . 323

Chapter 15. Overview 327

Resource Manager and Pervasive Devices	327
What Is New in This Version	328
Resource Manager and Users	328

Chapter 16. Installing and Configuring Devices for Resource Manager. . . . 331

Installing the Device Agent for Palm OS Devices	331
Using HotSync Manager.	331
Connecting to the Server with a Cradle.	331
Auto-connection to the Web Gateway Server	332
Conduit Communication with the Device Agent	332
Planning for Installation and Configuration	332
Installing the Files.	332
Configuring the Device Agent using the GUI	333
Preparing a Configuration File	334
Setting Up the PDBGenerator Utility	336
Administrative Tasks for Palm OS Devices	337
Changing a Configuration Parameter	337
Changing the Value for the Network Attachment Option	338
Manually Setting Up a Modem Connection	339
Creating a Network Interface (Manual Modem Setup)	339
Configuring the Modem (Manual Modem Setup)	340
Configuring the Device Agent (Manual Modem Setup)	340
SSL and the Device Agent	341
Enabling SSL	341
Disabling SSL	341
Disabling the Auto-connection When Using a Cradle.	341
Using a Security Manager	342
Installing the Device Agent for Windows CE Devices	342
Communicating with the Host PC	342
Planning for the Installation	342
Installing the Device Agent.	343
Configuring the Device Agent	344
Configuration File	346
Keeping Device Agent Files on Windows CE	348
Administrative Tasks for Windows CE Devices	349
Using SSL with the Device Agent.	349

Managing the List of Locally Valid Certificate Authority Certificates.	349
Using a Security Manager	349
Administrative Tasks for Nokia Devices	350
Enabling security for Nokia s60 devices	350
Configuring the DMS server	350
Configuring the IBM HTTP server	350
Downloading the certificate on the device	350
Enabling the WebSphere Application Server security	350
Creating a connection profile on the Nokia s60 device	350

Chapter 17. Managing Resources. . . 353

Sample Tivoli Environment.	354
Registering the Resource Types	354
Working with Pervasive Devices	355
Registering Endpoints as Resource Gateways	355
Enrolling Pervasive Devices Automatically	355
Discovering Devices	356
Creating Devices	357
Modifying Devices	360
Deleting Devices	361
Nokia Provisioning Scenario	362
Working with Users	368
Viewing Users	368
Grouping Resources	369
Using the Tivoli Policy Facility	369
Writing Scripts	370
Using the Tivoli Query Facility	370
Creating Resource Groups	370
Adding Resources to a Static Resource Group	371
Adding Dynamic Resource Groups	372
Listing and Finding Resources.	373
Customizing Resource Manager Actions	375
Script for wresource action Command	375

Chapter 18. Device Customization Parameters 377

Creating a Customization File for Palm OS Devices	377
Format Parameters	377
Preset Countries or Regions and Stored Values.	378
Time Formats and Stored Values	379
Date Formats and Stored Values	379
Long Date Formats and Stored Values	379
Number Formats and Stored Values.	379
General Parameters	380
Network Parameter	380
TCP/IP Parameters	381
Modem Parameter.	381
Agent Parameters	381
Proxy Parameters	382
Sample File for Windows CE Devices	382
PPP Parameters	383
TCP/IP Parameters	383
Browser Parameters	384
Mailer Parameters.	385
Management Parameters	385

Chapter 19. Using the Command Line 387

wresgrp	388
wresgw	391
wresource	394

Chapter 20. Sample Scripts 401

Script for Use with wresource action Command	401
Script for Default Policy	402
Script to Create Resource Group and Activate Default Policy	403

Chapter 21. Troubleshooting 405

Tivoli Resource Manager Traces	405
Tivoli Web Gateway Logs and Traces	405
Tivoli Web Gateway Configuration Files	407
TheTransaction.properties file	407
Managing the Transaction.properties File	408
The traceConfig.properties File	408
Tips for Performance Improvement	409
Tips to Improve Web Server Performance on AIX and Solaris Systems.	409
Improving Application Server Performance	410
Troubleshooting Connection Problems between Device Manager and Devices	411
Deleting a Tivoli Endpoint That Is a Resource Gateway	411
Deleting and Reconnecting an Endpoint with the Same Label	412
Deleting and Reconnecting an Endpoint with a New Label	412
Changing the User ID of Palm Devices	413
Resource Groups Do Not Exist	413
Grayed out Query Selection Box on Resource Group Members Dialog	413
Activity Planner login failure on Linux	413

Part 6. Managing an Enterprise Directory 415

Chapter 22. Using the GUI 417

Creating a Directory Query Library	417
Creating a Directory Query	418
Editing a Directory Query	420
Using Directory Queries	421
Running a Directory Query	421

Chapter 23. Using the Command Line 425

Directory Query Commands	425
wcrtdirquerylib.	426
wcrtdirquery	427
wsetdirquery	429
wgetdirquery	431
wrunidirquery	432
Directory Query Context Commands	433
wcrtdirectx	434
wgetdirectx	436
wsetdirectx	437
wsetdirectxpw	438
wmanagedir.	439

Chapter 24. Troubleshooting 441

Enabling Traces.	441
Registering Enterprise Directory Query Facility Resources	441
Values set for the LD_LIBRARY_PATH environment variable are lost	441

Appendix A. Accessibility 443

Navigating the Interface Using the Keyboard.	443
Activity Planner	443
Configuration Change Manager	444
Magnifying What Is Displayed on the Screen.	444
Enabling the IBM Voice Kit to Function with the Activity Planner and Configuration Change Manager GUIs	444

Appendix B. Support information 445

Searching knowledge bases.	445
Search the information center on your local system or network.	445
Search the Internet	445
Obtaining fixes	445
Contacting IBM Software Support	446
Determine the business impact of your problem	447
Describe your problem and gather background information	447
Submit your problem to IBM Software Support	447

Notices 449

Trademarks	450
----------------------	-----

Glossary 451

Index 455

Figures

1.	Conditioning activities	34	5.	Users Integrated into a Tivoli Environment by Resource Manager	329
2.	Example Reference Model	162	6.	Nokia Other Information dialog	366
3.	Schematic representation of a reference model	166			
4.	Pervasive Devices Integrated in a Tivoli Environment.	327			

Tables

1.	Activity Planner roles and operations	6
2.	Target statuses	46
3.	Activity statuses	46
4.	Activity plan statuses	46
5.	Subelements that define the activity plan	58
6.	Subelements that define frequency information	62
7.	Subelements that define activities in the activity plan definition file	64
8.	Subelements that define the execution window	66
9.	Supported built-in variables for specifying targets	69
10.	Supported task library operations and parameters.	70
11.	Parameter values.	70
12.	Parameters for the accept operation	72
13.	Parameters for the install operation	73
14.	Parameters for the remove operation	74
15.	Parameters for the commit operation	75
16.	Parameters for the undo operation.	76
17.	Parameters for the load operation	77
18.	Parameters for the unload operation	77
19.	Parameters for the send operation	78
20.	Parameters for the retrieve operation	79
21.	Parameters for the delete operation	80
22.	Parameter values.	80
23.	Parameters for the scan operation	84
24.	Parameter values.	85
25.	Parameter values.	87
26.	Commands for managing the activity plan database	88
27.	Commands for managing activity plans	89
28.	Commands for monitoring activity plans	89
29.	Commands for managing return values and error levels.	89
30.	Commands for performing administrative tasks.	90
31.	Commands for registering plug-ins	90
32.	Keys, operators and values for the -f option.	91
33.	Return values	130
34.	Sections of the confccm.xml file	154
35.	Subelements for <table_entry> element	156
36.	Subelements for <action_entry> element	156
37.	Subelements for <parameter> element	157
38.	Software states and actions	157
39.	Summary of Elements in the Example Reference Model	162
40.	Subscribers to the Example Reference Model	163
41.	Subelements for <model> elements	163
42.	Subelements for <element> elements	164
43.	Subelements for <dependency> elements	165
44.	Subelements for <subscriber> elements	165
45.	176
46.	Commands for managing reference models	195
47.	Pristine Manager Script Files	224
48.	Tivoli Enterprise Console event class for Pristine Manager	230
49.	Tivoli Enterprise Console event attributes for Pristine Manager	230
50.	Commands for working with Pristine Manager	259
51.	Web Interface Task Bar Buttons	311
52.	Columns in the Software Packages Window	314
53.	Columns in the Reference Models Window	318
54.	Palm OS configuration parameters	335
55.	Filtering Methods	369
56.	Commands for working with Resource Manager	387
57.	Enterprise Directory Query Facility Commands	425
58.	Keyboard shortcuts for the Activity Plan Editor service	443
59.	Keyboard shortcuts for the Activity Plan Monitor service.	444
60.	Keyboard shortcuts for Configuration Change Manager	444

About This Guide

IBM® Tivoli® Configuration Manager Version 4.3.1 provides remote system management facilities for your enterprise. The *IBM Tivoli Configuration Manager: User's Guide for Deployment Services* describes the different services you can use to facilitate the use of Software Distribution and Inventory components.

Who Should Read This Guide

This guide is intended for system administrators and users who perform software distribution and inventory operations. Readers should be familiar with the following topics:

- UNIX® and PC platforms
- Shell programming
- Database architecture and concepts
- Graphical user interfaces

What This Guide Contains

This guide contains the following sections:

Part 1. Planning, Scheduling, and Monitoring Activities

Part 1 contains the following chapters:

- Chapter 1, "Using Activity Planner," on page 5
Provides general information about Activity Planner.
- Chapter 2, "Performing Activity Planner Operations," on page 19
Describes how to use the Activity Planner service to define and schedule groups of activities.
- Chapter 3, "Using the Command Line," on page 57
Describes how to use Activity Planner commands.
- Chapter 4, "Troubleshooting," on page 131
Describes how to solve basic problems that might occur when using Activity Planner.

Part 2. Modelling the Enterprise Configuration

Part 2 contains the following chapters:

- Chapter 5, "Using Change Manager," on page 147
Provides general information about Change Manager.
- Chapter 6, "Performing Change Manager Operations," on page 171
Describes how to use the Change Manager service to model the enterprise and use reference models to maintain the environment.
- Chapter 7, "Using the Command Line," on page 195
Describes how to use Change Manager commands.
- Chapter 8, "Troubleshooting," on page 213
Describes how to solve basic problems that might occur when using Change Manager

Part 3. Installing an Operating System on Pristine Machines

Part 3 contains the following chapters:

- Chapter 9, “Overview: Installing an Operating System on Pristine Machines,” on page 223
Describes the Pristine Manager concepts and configuration steps.
- Chapter 10, “Installing on Pristine Machines,” on page 233
Describes the main tasks required to perform a pristine installation.
- Chapter 11, “Using the Command Line,” on page 259
Describes how to use Pristine Manager commands.
- Chapter 12, “Troubleshooting,” on page 273
Describes how to solve basic problems that might occur when using Pristine Manager.

Part 4. Web Interface

Part 4 contains the following chapters:

- Chapter 13, “Administering the Web Interface,” on page 281
Describes how to perform administration tasks to enable and support remote users to use the Web Interface service.
- Chapter 14, “Using the Web Interface,” on page 309
Describes how a remote user uses the Web Interface service to install software, run inventory scans, and view and synchronize reference models.

Part 5. Managing Resources

Part 5 contains the following chapters:

- Chapter 15, “Overview,” on page 327
Provides an introduction to Resource Manager.
- Chapter 16, “Installing and Configuring Devices for Resource Manager,” on page 331
Describes how to install and configure pervasive devices to work with the resource gateway and Resource Manager.
- Chapter 17, “Managing Resources,” on page 353
Describes to examine the computers in your enterprise to determine where resources are associated.
- Chapter 18, “Device Customization Parameters,” on page 377
Describes how to customize devices.
- Chapter 19, “Using the Command Line,” on page 387
Describes how to use Resource Manager commands.
- Chapter 20, “Sample Scripts,” on page 401
Provides sample device configuration scripts.
- Chapter 21, “Troubleshooting,” on page 405
Describes how to solve basic problems that might occur when using Resource Manager.

Part 6. Managing an Enterprise Directory

Part 6 contains the following chapters:

- Chapter 22, “Using the GUI,” on page 417
Describes how to use the Enterprise Directory Query Facility to create, edit, and run directory queries.
- Chapter 23, “Using the Command Line,” on page 425
Describes how to use Enterprise Directory Query Facility commands.
- Chapter 24, “Troubleshooting,” on page 441
Describes how to solve basic problems that might occur when using the Enterprise Directory Query Facility.

Appendixes

There are two appendixes:

- Appendix, Appendix A, “Accessibility,” on page 443
Accessibility features help users who have physical disabilities, such as restricted mobility or limited vision, to use software products successfully.
- Appendix, Appendix B, “Support information,” on page 445
Describes options for obtaining support for IBM products.

Publications

This section lists publications in the IBM Tivoli Configuration Manager library and related documents. It also describes how to access Tivoli publications online and how to order Tivoli publications.

IBM Tivoli Configuration Manager Library

The following documents are available in the IBM Tivoli Configuration Manager library:

- *IBM Tivoli Configuration Manager: Introducing IBM Tivoli Configuration Manager*, GC23-4703
Provides an overview of IBM Tivoli Configuration Manager and its components, as well as providing user scenarios to highlight various processes.
- *IBM Tivoli Configuration Manager: Planning and Installation Guide*, GC23-4702
Explains how to install, upgrade, and uninstall the product and its components in a Tivoli environment.
- *IBM Tivoli Configuration Manager: User's Guide for Software Distribution*, SC23-4711
Explains the concepts and procedures necessary for you to effectively use the Software Distribution component to distribute software over local area networks (LANs) and wide area networks (WANs).
- *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*, SC23-4712
Explains advanced features and concepts needed to use and tailor the Software Distribution component.
- *IBM Tivoli Configuration Manager: User's Guide for Deployment Services*, SC23-4710
Provides information about the Deployment Services of the product.
- *IBM Tivoli Configuration Manager: User's Guide for Inventory*, SC23-4713
Describes the Inventory component and the management tasks that you can perform.
- *IBM Tivoli Configuration Manager: Database Schema Reference*, SC23-4783
Provides information about the IBM Tivoli Configuration Manager repository.

- *IBM Tivoli Configuration Manager: Messages and Codes*, SC23-4706
Details all the error, warning messages and error codes issued by all the components and services of the product.
- *IBM Tivoli Configuration Manager: Patch Management Guide*, SC23-5263
Describes a solution that covers the distribution and management of security patches and software updates in a Tivoli environment.
- *IBM Tivoli Configuration Manager: Guide for Active Directory Integration*, SC32-2285
Describes the integration of Microsoft Active Directory with your Tivoli environment.
- *IBM Tivoli Configuration Manager: License Management Extension*, SC32-2260
Describes the license management facilities provided in your Configuration Manager environment.
- *IBM Tivoli Configuration Manager: User's Guide for Operating System Deployment Solution*, SC32-2578
Describes how you can implement an operating system deployment solution delivered with Configuration Manager.
- *IBM Tivoli Configuration Manager: Release Notes*, G111-0926
Contains late-breaking information about the product.

Related Publications

The following documents also provide useful information:

- *IBM Tivoli Enterprise Console: Installation Guide*, SC32-1233
Explains how to install and upgrade Tivoli Enterprise software within your Tivoli region using the available installation mechanisms provided by Tivoli Software Installation Service and Tivoli Management Framework.
- *Tivoli Management Framework: Planning for Deployment Guide*, GC32-0803
Explains how to plan for deploying your Tivoli environment. It also describes Tivoli Management Framework and its services.
- *Tivoli Management Framework: Maintenance and Troubleshooting Guide*, GC32-0807
Explains how to maintain a Tivoli environment and troubleshoot problems that can arise during normal operations.
- *Tivoli Management Framework: Reference Manual*, GC32-0806
Provides in-depth information about Tivoli Management Framework commands. This manual is helpful when writing scripts that are later run as Tivoli tasks. This manual also documents default and validation policy scripts used by Tivoli Management Framework.
- *Tivoli Management Framework: User's Guide*, GC32-0805
Describes the concepts and procedures for using Tivoli Management Framework services. It provides instructions for performing tasks from the Tivoli desktop and from the command line.
- *IBM Tivoli Configuration Manager Warehouse Enablement Pack: Implementation Guide*
Describes how to install and configure the warehouse enablement pack for the IBM Tivoli Configuration Manager product and describes the data flow and structures that are used by the warehouse pack.

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

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

Accessing Publications Online

The documentation CD contains the publications that are in the product library. The format of the publications is PDF, HTML, or both. Refer to the readme file on the CD for instructions on how to access the documentation.

The product CD contains the publications that are in the product library. The format of the publications is PDF, HTML, or both. To access the publications using a Web browser, open the infocenter.html file. The file is in the appropriate publications directory on the product CD.

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli software information center Web site. Access the Tivoli software information center by first going to the Tivoli software library at the following Web address:

<http://www.ibm.com/software/tivoli/library/>

Scroll down and click the **Product manuals** link. In the Tivoli Technical Product Documents Alphabetical Listing window, click the **IBM Tivoli Configuration Manager** link to access the product library at the Tivoli software information center.

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

Ordering Publications

You can order many Tivoli publications online at the following Web site:

<http://www.elink.ibm.link.ibm.com/public/applications/publications/cgibin/pbi.cgi>

You can also order by telephone by calling one of these numbers:

- In the United States: 800-879-2755
- In Canada: 800-426-4968

In other countries, see the following Web site for a list of telephone numbers:

<http://www.ibm.com/software/tivoli/order-lit/>

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully. With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For additional information, see Appendix A, “Accessibility,” on page 443.

Tivoli Technical Training

For Tivoli technical training information, refer to the following IBM Tivoli Education Web site:

<http://www.ibm.com/software/tivoli/education>

Support Information

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

- Searching knowledge bases: You can search across a large collection of known problems and workarounds, Technotes, and other information.
- Obtaining fixes: You can locate the latest fixes that are already available for your product.
- Contacting IBM Software Support: If you still cannot solve your problem, and you need to work with someone from IBM, you can use a variety of ways to contact IBM Software Support.

For more information about these three ways of resolving problems, see Appendix B, “Support information,” on page 445.

Conventions Used In This Guide

This guide uses several conventions for special terms and actions, operating system-dependent commands and paths, and margin graphics.

Typeface Conventions

This guide uses the following typeface conventions:

Bold

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

Italic

- Words defined in text
- Emphasis of words (words as words)
- New terms in text (except in a definition list)
- Variables and values you must provide

Monospace

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

Operating System-dependent Variables and Paths

This guide uses the UNIX convention for specifying environment variables and for directory notation.

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

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

Using the Command Line

Commands enable you to perform system operations from a UNIX or PC command line instead of using the Tivoli desktop. Most Tivoli commands begin with the letter W. Vowels are often omitted to shorten the name of a command. Commands are named using two conventions, depending on their provenance:

- Commands that are inherited from the previous versions of Software Distribution are named using the *w+verb+object* convention, which matches the way you might think of the action. For example, to import a reference model in Change Manager, you use the **wimprmod** command. To delete a reference model, you use the **wdelrmod** command.
- Other commands are named using the *w+service+activity* convention. For example, to work with resource gateways in Resource Manager you use the **wresgw** command, and to configure the Web Interface you use the **wwebcfg** command.

It is often necessary or convenient to invoke a Tivoli management application operation from the command line rather than from the desktop. For example:

- If you do not have access to a desktop, for example, if you are connected to the network by a modem.
- If you want to group several operations in a shell script or batch file.
- If an operation is not available using the desktop.
- If you prefer to invoke a command from a shell.

Command Line Syntax

The commands described in this book use the following special characters to define the syntax of commands:

- | | |
|-----|--|
| [] | Identifies optional attributes. Attributes not enclosed in brackets are required. |
| ... | Indicates that you can specify multiple values for the previous attribute. |
| | Indicates mutually exclusive information. You can use the attribute to the left of the separator or the attribute to its right. You cannot use both attributes in a single use of the command. |
| { } | Delimits a set of mutually exclusive attributes when one of the attributes is required. If the attributes are optional, they are enclosed in square brackets ([]). |
| \ | Indicates that the syntax in an example wraps to the next line; You should not break the command syntax at this point, it is just a documentation convention. |

Notes:

1. Keywords and values are separated from each other by one or more spaces, unless stated otherwise.
2. All commands described in this book are issued as a single command string, unless stated otherwise.

For example, the following is the syntax of the `wweb` command, when used to publish a Web object:

```
wweb -publish -p publicName -v version [-i all | [-i interp ] ... ] \  

-w app_server_ep_label [-w app_server_ep_label] ... [-c connspeed] \  

[-u all | [-u user ] ... ] [-U file] ... [-f | -n] @profile
```

Note: The first two lines end with the syntax wrap symbol, showing that the command comprises all three documented lines, written at the command line as one single command.

The same command is now shown with every option on a separate line to make it more readable:

```
wweb \  

  -publish \  

  -p publicName \  

  -v version \  

  [-i all | [-i interp ] ... ]\  

  -w app_server_ep_label [-w app_serv_ep_label] ... \  

  [-c connspeed] \  

  [-u all | [-u user ] ... ]\  

  [-U file] ... \  

  [-f | -n] \  

  @profile
```

For this command you *must* specify the following;

- The **wweb** command name
- The **-publish** keyword
- A **-p** keyword, followed by the *publicName*
- A **-v** keyword followed by the *version*
- At least one occurrence of the **-w** keyword followed by an *app_server*
 The first occurrence is not enclosed in brackets, and is therefore required; a subsequent occurrence is enclosed in brackets and is therefore optional; the ellipsis (...) indicates that there can be any number of the optional occurrences
- Either **-i all** or at least one occurrence of the combination of the **-i** keyword followed by an *interp*
 The options are enclosed in braces ({ }) and separated by the logical *or* character (|)
- A *profile*, preceded immediately (no space) by an atsign (@)

In addition, you can *optionally* specify;

- A **-c** keyword followed by a *connspeed*
- Either **-u all** or one or more **-u** keywords followed by a *user*
- A **-U** keyword followed by a *userfile*
- Either a **-f** keyword or an **-n** keyword (not both)

Getting Help on Commands

Full details of Deployment Services commands, with their syntax and descriptions of their functions, are given in the various chapters of this guide. On UNIX managed nodes, you can access online versions of these details by using the **man** command (for example, **man wweb**). On any platform, you can access summary help for any command by typing the command name, without specifying any options, on the command line.

Part 1. Planning, Scheduling, and Monitoring Activities

Chapter 1. Using Activity Planner.	5	Loading an Activity Plan	43
Before You Start	5	Saving the Activity Plan	43
Understanding the Activity Planner Environment	5	Deleting Activity Plans or Activities	44
Activity Planner Concepts	6	Submitting and Monitoring Activity Plans	44
Activity Planner Roles	6	Activity Plan Status	45
Enabling IBM Tivoli Configuration Manager		Filtering Activity Plans, Targets, or Activities	47
Components for the Activity Planner	7	Submitting an Activity Plan	49
The Activity Planner Configuration File	8	Defining the Variable Values.	50
Defining the Activity Planner Engine Parameters	9	Updating a Submitted Plan	50
The apmJNI.ini File.	14	Stopping the Recursion of a Submitted Activity	
Activity Planner Server Variables	15	Plan	51
Activity Planner Command Line Variable	15	Controlling the Execution of Activity Plans and	
Using the Activity Plan Editor and the Activity Plan		Activities	51
Monitor	16	Generating a Recovery Plan	52
 Chapter 2. Performing Activity Planner		Launching the Distribution Status Console	53
Operations	19	Updating the Status of an Activity Plan	53
Launching the Activity Planner GUIs.	19	Deleting Completed Activity Plans	53
Defining an Activity	20	Cleaning Up the Database	54
Task Library Tasks	22	 Chapter 3. Using the Command Line	57
Defining a Task Library Task	22	Activity Plan Definition File	57
Software Distribution Operations	22	Setting Frequency Information	62
Defining a Software Distribution Activity	23	Defining Variables with the Variable Subelement	63
Inventory Operations	24	Defining Activities in the Activity Plan Definition	
Defining an Inventory Activity	24	File	63
Pristine Manager Operations	24	The Execution Window	65
Defining a Pristine Manager Activity	25	Using Conditions	66
Using Variables	25	Defining Targets.	68
Selecting Targets for Activities and Activity Plans.	25	Supported Task Library Activities	69
Selecting Targets for an Activity	26	Parameters of Task Library Operations	69
Excluding Targets from an Activity	27	Parameter Values for Task Library Operations	70
Specifying a List of Target Names	27	Supported Software Distribution Activities	70
Specifying a File Name Containing a List of		Parameters of Software Distribution Operations	71
Targets	28	Parameter Values for Software Distribution	
Specifying an Inventory Subscriber as a Target	28	Operations	80
Specifying a Query Library Subscriber as a		Supported Inventory Activities	84
Target	29	Parameters of Inventory Operations	84
Specifying a Directory Query Library		Parameter Values for the Scan Operation	85
Subscriber as a Target	29	Supported Pristine Manager Activities	87
Specifying a Profile Subscriber as a Target	29	Parameter of Pristine Operations	87
Specifying a Resource Group of Pervasive		Parameter Values for the Pristine Installation	
Devices as a Target	30	Operation	87
Specifying a Resource Group of Users as a		Managing Activity Plans	88
Target	31	wapmfltr	91
Specifying a Pristine Target Subscriber	31	wapmgui	94
Using Variables to Specify Targets	32	wapmplugin	95
Conditioning the Execution of Activities.	33	wapmrin	97
Sorting Activities in Order of Execution	36	wcntpln	98
Scheduling the Execution of Activities	36	wdelpln	100
Defining the General Properties of the Activity Plan	37	wdelstat	101
Scheduling the Activity Plan.	38	wexppln	102
Scheduling Plans to Execute within a Time		wgenpln	103
Interval.	38	wgeterrlev	106
Scheduling Activity Plans to Run Periodically	40	wimppln	107
Interrupting the Recursion of an Activity Plan	41	wlstpln	108
Selecting Targets for an Activity Plan	42	wmonpln.	110

wresetlev	113	Activity Planner Core Trace	135
wsetapmpw	114	Activity Planner Main Trace	136
wseterrlev	115	Activity Planner Executer Trace	136
wsfdpln	117	Activity Planner Handler Trace	136
wstartapm	120	Activity Planner Command Line Interface Trace	136
wstopapm	121	Activity Plan Monitor Trace	137
wsubpln	122	Activity Planner Default Trace.	137
wunlockpln	126	Downloading Plug-ins	137
wupdpln	127	Managing DB2 Deadlock	138
Return Values	130	Managing Different Domains	139
Chapter 4. Troubleshooting	131	Activity Planner Installation Notes	139
Activity Planner Logs and Traces.	131	Managing Linux Versions	140
wtrcapm	132	Using X sessions	140
Main Activity Planner Log	135	Checkpoint restart mechanism.	140
Activity Plan Monitor and Editor Startup Trace Files	135	Specific Problems and Workarounds.	142

This part describes how you use the Activity Planner to create and control plans. It also describes how you use the Activity Planner with Software Distribution and Inventory.

Locating Related Information

Information related to this service is available from the following sources:

- *Introducing IBM Tivoli Configuration Manager*. This provides an overview of the service.
- *Planning and Installation Guide*. This includes information about how you install the service.
- *Messages and Codes*. This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 1. Using Activity Planner

Activity Planner enables you to define a group of activities in an activity plan, submit the plan to be executed, and monitor the execution of the plan. Activities are single operations that are performed on a set of targets at specified times. Operations include Tivoli Management Framework Task Library tasks, Software Distribution, Inventory, and Pristine Manager operations if the corresponding plug-ins are installed.

Activities contained in a plan can have dependencies associated with them which define circumstances under which the activity is executed. The execution of the operation defined in the activity is performed by the application to which the operation belongs. The group of activities forms the activity plan.

Before You Start

This section contains introductory information you must understand before you start using Activity Planner. It contains the following sections:

Before you start using Activity Planner, you must perform the following operations:

- Ensure that all installation and configuration tasks have been completed.
Ensure that the prerequisites for using the Java GUI have been installed on the system you are using, and the required RDBMS configurations have been made. See *Planning and Installation Guide* and *Database Schema Reference*.
- Assign an Activity Planner role to the user ID with which you are working.
The user ID with which you are working must have at least the following roles:
 - Framework user role.
 - RIM_view and RIM_update roles.
 - Appropriate Activity Planner role for operation, as described in “Activity Planner Roles” on page 6.

There are four Activity Planner roles:

- APM_View
- APM_Manage
- APM_Edit
- APM_Admin.

To complete all the tasks described in the sections that follow, you must assign at least the APM_Edit role. For more information about Activity Planner roles see the “Activity Planner Roles” on page 6.

Understanding the Activity Planner Environment

When you have completed the installation and configuration of Activity Planner, as described in the *Planning and Installation Guide* manual, the following Activity Planner components are present:

- The Activity Planner database and RIM object.
- Activity Planner Java GUI on the managed nodes where the Java GUI component has been installed.

The first time Activity Planner is run, the apm.ini file is created. This file defines the configurable components of Activity Planner, for example:

- Name of the Activity Planner user.
- Trace information.
- Log information.
- Plug-in information.

For more information, see “The Activity Planner Configuration File” on page 8.

In addition, Activity Planner uses the following components:

- The Activity Planner database, from which Activity Planner stores and retrieves plan and activity information, for example plan schedules, targets, and conditions.
- The Tivoli database, from which Activity Planner obtains information about software package objects and Tivoli endpoints.

Activity Planner Concepts

This section introduces the following Activity Planner concepts:

Activity

An operation in an activity plan that is performed on a set of targets on a specific schedule and which can depend upon the execution of other activities.

Activity Plan

A set of activities performed on a set of targets on a specified schedule.

Activity Planner Roles

The tasks you can perform using the Activity Planner are restricted by the Tivoli management region roles assigned to you. Depending on the operations you are required to perform, you must have one or more of the roles described below. To use Activity Planner, at least Tivoli Framework role “user” is required.

Table 1 lists the roles required to perform Activity Planner operations.

Table 1. Activity Planner roles and operations

Operation	Context	Required Role
Launch Activity Planner GUIs.	Activity Planner GUIs	APM_View RIM_view RIM_update
Display activity plans maintained in the activity plan database	Activity plans	
Export an activity plan in the activity plan definition file format	Activity plans	
Display the status of a submitted activity plan	Activity plans	
Display current error level mapping	Activity plans	

Table 1. Activity Planner roles and operations (continued)

Operation	Context	Required Role
Create an activity plan	Activity plans	APM_View APM_Edit RIM_view RIM_update
Import an activity plan into the activity plan database	Activity plans	
Delete an activity plan from the activity plan database	Activity plans	
Submit an activity plan for execution	Activity plans	APM_Manage APM_View APM_Manage RIM_view RIM_update
Update a submitted plan that has not yet started	Activity plans	
Cancel, pause, resume, and restart activities or the activity plan	Activities and activity plans	
Generate a recovery plan after a failure	Activity plans	
Clean up submitted activity plans from the activity plan database	Activity plans	
Remove the status of activity plans from the activity plan database	Activity plans	
Map error levels to return codes	Activity plans	APM_Admin RIM_view RIM_update
Delete mapping of error levels and reset default values	Activity plans	
Display a list of locked plans and unlock plans currently locked	Activity plans	
Start and stop the Activity Planner engine	Activity Planner feature	
Display and change the RIM object name	Activity Planner feature	APM_Admin senior admin super RIM_view RIM_update
Change the user password for the Activity Planner engine	Activity Planner feature	

Enabling IBM Tivoli Configuration Manager Components for the Activity Planner

You can use the Activity Planner in conjunction with the other IBM Tivoli Configuration Manager components, Inventory, Software Distribution, and Pristine Manager by registering the related plug-ins, as described in “wapmplugin” on page 95, or running the appropriate script as described in *Planning and Installation Guide*. If you installed Activity Planner using the installation program, the Task Library plug-in is automatically installed; if you installed Activity Planner using the Tivoli desktop, you need to register the plug-in using the wapmplugin command. For more information on registering plug-ins, refer to *Planning and Installation Guide*.

The Activity Planner Configuration File

The Activity Planner configuration file, called `apm.ini`, includes information such as the following:

- The name of the Activity Planner user (`TME_User`). This information is used for filling in the login panel at GUI startup.
- The host name of the machine where the Activity Planner server is running (`TME_Host`). This information is used for filling in the login panel at GUI startup.
- The log file name including all steps and operations performed by the Activity Planner (`log_file`).
- The directory where the log file is stored (`working_dir`).
- The maximum size of the log file (`log_max_file`).
- The log level for the log file (`log_level`). Possible values are as follows:
 - 0** This is the default value
 - 1** Logs informational messages
 - 2** Logs warning messages
 - 3** Logs error messages. If a value higher than three is entered, the log level is set to three.
- The settings and information concerning trace files. For more information about log and trace files, see “Activity Planner Logs and Traces” on page 131
- The engine tuning settings. For more information, see “Defining the Activity Planner Engine Parameters” on page 9.
- Information concerning plug-ins.

The `apm.ini` file is stored in one of the following locations:

Windows operating systems

`%SystemDrive%`

UNIX operating systems

`/etc/Tivoli`

To change configuration information for Activity Planner, you can edit the `apm.ini` file. The `apm.ini` file is created the first time the Activity Planner is started. To modify the settings for traces and logs, you can also use the **wtrcapm** command. For more information on this command, see “wtrcapm” on page 132. On Windows operating systems, it can have the following example configuration:

```
;APM configuration file
[ENGINE_TUNING]
load_paused_plans_at_startup=no
pre_loading_tmf_objects_threshold=0
cache_global_target_info=no
global_cache_max_size=500
global_cache_refresh_timeout=120
cache_local_target_info=yes
executer_max_threads=5
executer_min_threads=0
executer_max_idle_time=60
deep_gc_interval=-1
checkpoint_restart=enabled
db_retry_count=3
retry_wait_interval=15
cancel_unique_targets=no
retrieve_gateways_info=no
commit_interval=500
[DEFAULT]
trace_files_num=3
trace_level=0
working_dir=C:\Program Files\Tivoli\Desktop\..\apm
trace_size=1000000
```

Enabling Configuration Manager Components for the Activity Planner

```
log_max_file=100000
log_level=3
plugin_download=enabled
default_file_path=C:\Program Files\Tivoli\apm
log_file=apmlog
TME_Host=jupiter
TME_User=Administrator
TimeZone=Indian Time: *IST.-5:30*
[MAIN]
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
[HANDLER]
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
[EXECUTER]
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
[APMCLI]
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
[APMEDITOR]
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
plugin_download=enabled
[MONITOR]
enable_auto_update=true
auto_update_interval=180
trace_files_num=3
trace_level=0
working_dir=C:\Tivoli\bin\w32-ix86\..\apm
trace_size=1000000
plugin_download=enabled
```

Some of the keys listed in the example below are not present by default, but must be added manually.

If the apm.ini file is modified, the changes made are effective only after Activity Planner is stopped and restarted.

The information contained in the apm.ini file related to sections [DEFAULT], [MAIN], [HANDLER], [EXECUTER], [APMCLI], and [MONITOR] refers to the trace files. For more information on trace files, see “Activity Planner Logs and Traces” on page 131.

When the Activity Planner is removed, the apm.ini file is automatically deleted.

Defining the Activity Planner Engine Parameters

To improve product performance, you can define the engine tuning parameters described below according to the size and configuration of your environment.

You can set the **notify_ext_directly** key to **a** using the Software Distribution **wswdcfg** command. This key allows the user to skip the notification manager

Enabling Configuration Manager Components for the Activity Planner

queue and the Inventory database validation when notifying Activity Planner about Software Distribution activities, thus improving product performance. For more information on this key, refer to *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*.

You can also modify the MDist 2 **notify_interval** parameter of the gateway to return the reports less frequently. This parameter specifies the frequency (in minutes) of status reporting. When the **notify_interval** has elapsed or the distribution has completed on all targets, the results are flushed. Refer to the *Tivoli Management Framework: Reference Manual* for more information about modifying MDist 2 parameters using the **wmdist** command.

In the [ENGINE_TUNING] section of the apm.ini file, you can customize the following parameters:

load_paused_plans_at_startup

Specifies whether plans in paused state are loaded in the memory at engine startup. Possible values are **yes** and **no**. The default value is **yes**.

pre_loading_tmf_objects_threshold

Specifies the minimum number of targets for an activity required to load the endpoint and managed node list available in the Tivoli Management Region. This list is used to search for the object ID of the target based on its name. This cache has no effect on the **cache_local_target_info** and **cache_global_target_info** caches. The average RAM usage for this cache is several megabytes if you have an environment with tens of thousands of endpoints. The default value is 200 targets.

cache_local_target_info

Specifies whether the engine must cache information about the targets of a single plan. If enabled, targets are evaluated only once at plan submission. This parameter applies only to plans in which the targets are specified at plan level and are resolved at plan submission. If several plans with many targets are submitted, setting this parameter can improve performance during the submission phase, while increasing memory usage. The information contained in this cache is flushed when the plan is completed. Possible values are **yes** and **no**. The default value is **yes**.

cache_global_target_info

Specifies whether the engine must globally cache information about targets. This cache is unique in Activity Planner and its data is shared between all active plans. Enable this cache if you have many plans in which targets are specified at activity level or resolved at activity execution, because the local cache cannot be used in these cases. Setting this parameter can improve performance during the submission phase, while increasing memory usage. Possible values are **yes** and **no**. The default value is **no**.

Note: When you delete and recreate a target having the same name, the target OID changes in the Activity Planner cache memory. If you try to perform any action on that target, the action is performed on the OID of the deleted target, until the **cache_global_target_info** cache memory is updated.

global_cache_max_size

Specifies the maximum size for the global target cache. The default value is 5000 targets with the related information, such as gateway name, ID, and so on. The average RAM usage for this cache is several megabytes if you have an environment with tens of thousands of endpoints.

Enabling Configuration Manager Components for the Activity Planner

global_cache_refresh_timeout

Specifies the number of minutes after which the data in the **pre_loading_tmf_objects_threshold** cache and the **cache_global_target_info** cache are no longer valid and must be updated. The default value is 120 minutes (2 hours).

Note: The data stored in the caches described above is updated only when the value defined in the **global_cache_refresh_timeout** parameter expires or when the Activity Planner engine is stopped and restarted.

executer_max_threads

Specifies the maximum number of threads that can be generated to perform external activities. Possible values are 5, -1, and any finite number. The default value is 0, which means that thread pooling is disabled. In this case, if an operation hangs, Activity Planner also hangs because no other thread is available. If you specify -1, the number of threads is unlimited; if you specify a finite number, that number is applied.

executer_min_threads

Specifies the minimum number of threads that must be present in the thread pool. The default value is 0.

executer_max_idle_time

Specifies the maximum idle time after which a thread of the pool may be shut down. The default value is 120 seconds.

deep_gc_interval

Specifies the number of minutes the Activity Planner engine must wait before forcing a deep garbage collection. The default value is -1, which means that the garbage collection is never explicitly invoked. The deep garbage collection is used to free memory, but the process is slow and should not be enabled if not strictly necessary. For example, on Windows machines, the import operation uses a large amount of memory and in that case it could be useful to enable this feature if many import operations must be done sequentially.

checkpoint_restart

Specifies whether checkpoint information is kept about internal activities performed by Activity Planner. If this parameter is enabled and Activity Planner activities are interrupted for any reason (for example, a system power off, or loss of connection to the database), when Activity Planner restarts, the activities are automatically recovered from the last checkpoint. This parameter is enabled by default. Supported values are **enabled** and **disabled**. For more information on the recovery options available with this parameter, see “Checkpoint restart mechanism” on page 140.

db_retry_count

Specifies the maximum number of times Activity Planner tries to reconnect to the database after a failed attempt. The default value is 3.

retry_wait_interval

Specifies the interval in seconds between each connection attempt to the database. The interval is automatically increased each time a new attempt is performed based on the number of the attempt by the specified retry interval. For example, if the connection attempt fails and the **retry_wait_interval** parameter is set to 120 seconds, the first retry is performed after 120 seconds, the second retry after 240 seconds, the third retry after 360 and so on, until either the connection is established or the

Enabling Configuration Manager Components for the Activity Planner

value defined for the **db_retry_count** parameter is reached and no other retries are performed. After the last unsuccessful retry, the operation is canceled and a message is written to the log file. The default value is 15 seconds.

cancel_unique_targets

Unique targets are targets included in the target list of the conditioned activity but not included in the target list of the conditioning activity. This parameter specifies whether unique targets must be started when the conditioned activity starts. The default value is **no**, which means that unique targets are started when the conditioned activity starts. If you set this parameter to **yes**, unique targets are not started and their status is set to **canceled_by_condition**. This setting applies to all plans generated by the Activity Planner.

retrieve_gateways_info

Specifies whether information concerning the gateways must be retrieved when plan information is inserted in the Activity Planner database. The default value is **yes**. If you set this parameter to **no**, the information is not retrieved. In this case, conditioning by depot is not supported, but a significant performance improvement is achieved when submitting plans.

commit_interval

Specifies the number of operations after which the data inserted in the Activity Planner database is saved. This setting applies to the plan submission phase, which is the one creating more data to be stored in the database. The default value is 500 operations, which means that data is committed after 500 Activity Planner operations have been completed. You can specify any positive integer, depending on the size of your environment. Note that frequent commit operations can significantly slow down system performance.

delstat_commit_interval

You can modify the **delstat_commit_interval** parameter. The parameter is not normally listed in the apm.ini file, and when not listed, its default value of 10 is assumed. Using the default value, a commit is done for each recursion of the same plan_id for the defined interval of 10 minutes. The key can be customized, depending on how many recursions of the same plan are present in the database.

use_cached_targets_threshold

You can modify the **use_cached_targets_threshold** parameter. When the number of targets is bigger than the **use_cached_targets_threshold** value, the global cache is used to resolve those targets, if enabled. The default value of **use_cached_targets_threshold** is 0. By default the global cache is always used when enabled.

In the [DEFAULT] section of the apm.ini file, you can customize the parameters listed below.

Due to a Java Runtime Environment limitation, some time zones are not supported. Customize the following parameter in the apm.ini file to work around this limitation:

TimeZone

The value of the TimeZone parameter can be the ID of a Java supported time zone or a custom time zone specified in the following format:

Enabling Configuration Manager Components for the Activity Planner

id.offset

for time zones without Daylight Savings Time

***id.offset*<Daylight start><Daylight end>**

for time zones with Daylight Savings Time

where

id The name of the time zone.

offset The base offset in hours from GMT without applying GMT. Use a positive value for time zones that are west of GMT and negative values for time zones east of GMT. For an offset that is not a multiple of 1 hour, use the HH:mm format.

<Daylight start><Daylight end>

The start and end date and time at which Daylight Savings Time (DST) respectively starts and ends. They are expressed in the format:

month.day.dayofweek/time

where

time The DST starting time expressed in the format HH:mm:ss, and where HH (hour) ranges from 0-23.

month Is the number of the month in which the DST starts/ends (January=1, December=12).

day.dayofweek

The day of the month can be expressed in the following different ways:

Day of week

To specify a weekday, set day of month to the weekday number (1 for SUNDAY) and day, to the occurrence number (positive if counting from the beginning of the month and negative if counting from the end of the month). For example: <3.-1.1/2:00:00> represents 2 o'clock of the last (day=-1) Sunday (dayofweek=1) in March (month=3) <4.1.1/2:00:00> represents 2 o'clock of the first (day=1) Sunday of April.

Day of month

To specify a fixed day of a month, set dayofweek to 0, and day, to the day of the month. For example: <3.1.0/2:00:00> represents the 2 o'clock of March 1st.

Day of week after day of month

To specify the first day of the week occurring on or after an exact day of the month, make the day of the week negative and use day to specify the day of the month to start from. For example: <3.5.-1/2:00:00> represents 2 o'clock of the first Sunday (dayofweek=-1) on or after March 5th (day=5).

Day of week before day of month

To specify the last day of the week occurring on or before an exact day of the month, make the day of the week and the day of the month negative. For example: <3.-21.-1/2:00:00> represents 2 o'clock of the first Sunday (dayofweek=-1) on or before March 21st (day=-21).

Enabling Configuration Manager Components for the Activity Planner

Note: In all the methods, the value of dayofweek is from +/-1 for SUNDAY to +/-7 SATURDAY

Consider the following examples:

West Europe Time (without DST): *WET.0*

UK and Portugal: *WET.0*<3.-1.1/2:00:00><10.-1.1/03:00:00>

US Central States Time: *CST.6*<4.1.1/2:00:00><10.-1.1/02:00:00>

European Central Time: *ECT.-1*<3.-1.1/2:00:00><10.-1.1/03:00:00>

Indian Time: *IST.-5:30*

To force the Portugal time zone, use the following expression:

TimeZone=*WET.0*<3.-1.1/2:00:00><10.-1.1/03:00:00>

inventory_rim_name

Specifies the name of the RIM to be used for retrieving inventory information. Use this parameter only when the name of the RIM is different from the default one. This information is used by the Activity Planner when resolving the names of Inventory subscribers.

trace_files_num=3

Specifies the maximum number of trace files to be created. You can set this parameter to any positive integer. If this parameter is not specified, the maximum number of trace files is 3. For more information on Activity Planner traces, see “Activity Planner Logs and Traces” on page 131.

The apmJNI.ini File

You set the apmJNI.ini file to define the memory allocation pool, or heap size, for the Java Virtual Machine. The apmJNI.ini file is a plain text file consisting of a set of keywords with their related values. You must create this file using a plain text editor. In the file, add one or more of the keywords specified below.

The following is an example of an apmJNI.ini file:

```
minHeapSize=2000000
maxHeapSize=2000000000
other=Xms4000000
```

You can customize the following parameters:

minHeapSize

Specifies the initial size, in bytes, of the memory allocation pool. This value must be a multiple of 1024 greater than 1MB. The default value varies depending on the operating system being used.

maxHeapSize

Specifies the maximum size, in bytes, of the memory allocation pool. This value must be a multiple of 1024 greater than 2MB. The default value varies depending on the operating system being used.

other Specifies one or more Java parameters which are forwarded directly to Java Virtual Machine without Software Distribution intervention. Use the exact Java command line command. This example shows a sample record to set the initial heap size: other=-Xms4000000.

The apmJNI.ini file must be stored in one of the following locations:

Windows operating systems

%SystemDrive%

UNIX operating systems

/etc/Tivoli

The apmJNI.ini file is processed when the Activity Planner starts.

Activity Planner Server Variables

The list below details the environment variables used by Activity Planner. To set these variables, use the **odadmin environ set** command. For more information on this command, refer to *Tivoli Management Framework: Reference Manual*.

APM_HOSTNAME

Specifies the hostname and optionally the port that the Activity Planner engine uses to connect to the Framework Object Request Broker with the following syntax:

hostname:port

This variable is not usually necessary, because the Activity Planner engine can calculate these values automatically.

APM_PORT

Specifies the starting socket port to be used by the two Activity Planner engine internal processes. The first available port, starting from the one you specify, is used. If no port is specified, port 7070 is used by default.

_APM_DEBUG_

Enables the tracing function for the Activity Planner engine. For more information on traces, see “Activity Planner Logs and Traces” on page 131.

APM_THREADS_FLAG

Specifies the threads flag for Linux operating systems. For more information on this variable, see “Managing Linux Versions” on page 140.

APM_KERNEL_LINUX

Specifies the version of the Linux kernel installed on the workstation. For more information on this variable, see “Managing Linux Versions” on page 140.

APM_TIMEOUT

Modifies the default Activity Planner timeout by increments of 10 minutes. This variable specifies how many 10-minute intervals the Activity Planner engine must wait before shutting down when idle.

Activity Planner Command Line Variable

The following variable is one of the environment variables used by the Activity Planner command line. It must be set in the shell used to issue command line commands.

APM_CLI_TIMEOUT

Specifies the default Activity Planner command line timeout. It specifies how many seconds the command line must wait for a response to an Activity Planner engine request. The default value is 300 seconds. Only for the **wsubpln -f** command, the timeout is automatically extended to 900 seconds.

Using the Activity Plan Editor and the Activity Plan Monitor

The Activity Planner comprises the Activity Plan Editor, the Activity Plan Monitor, and the corresponding command line interfaces (CLI).

Use the Activity Plan Editor to define and manage activities, and create activity plans from the GUI or CLI. You can also define activities in a file in XML format using a text editor. The XML file in which the activity plan is defined is called the *activity plan definition file*. See Chapter 3, “Using the Command Line,” on page 57 for more information about the activity plan definition file.

Use the Activity Plan Editor to perform the following tasks:

- Manage a group of activities, originating from different applications, as a single activity from a single machine in the network.
- Schedule the activity plan to run on a specific day and time, or to be repeated at specific time intervals, days of the week, or days of the month. See “Scheduling the Activity Plan” on page 38 for detailed information.
- Specify whether you want a notification message concerning the status of the plan to be sent to the administrator. If you select this feature, a notification message is sent to a mailbox you specify, or logged in the Activity Planner Notice group.
- Schedule the plan to repeat indefinitely. See “Scheduling Activity Plans to Run Periodically” on page 40 for detailed information.
- Schedule activities to run at specific time intervals during the week. See “Scheduling the Execution of Activities” on page 36 for detailed information.
- Set conditions on activities so that the execution of one activity is dependent on the completion result of other activities. See “Conditioning the Execution of Activities” on page 33 for detailed information.
- Save activity plans in a database or to an XML file. See “Saving the Activity Plan” on page 43 for detailed information.

You can save activity plans that you prepared using the Activity Plan Editor as any of the following:

- Drafts, if they are not yet complete
- Templates, if they are complete
- XML files

Draft plans and templates are stored in their respective repositories in the activity plan database, but only templates are made available for submission.

Use the Activity Plan Monitor GUI or CLI to select a saved plan template and submit the plan for execution. You can submit, monitor, control, and delete submitted activity plans using the Activity Plan Monitor GUI or the command line interface (CLI). See Chapter 3, “Using the Command Line,” on page 57 for more information about performing operations on activity plans using the CLI.

You can use the Activity Plan Monitor to perform the following tasks:

- Submit activity plans to be executed. For more information, see “Submitting an Activity Plan” on page 49.
- View all submitted activity plans and retrieve information such as completion status, start time and completion time of a specific activity plan, activity, or of an activity for a specified target. For more information, see “Submitting and Monitoring Activity Plans” on page 44.
- Specify one or more filters to be applied to plans, targets, or activities so that only the objects matching certain criteria are displayed in the Activity Plan

Monitor graphical interface or command line. For more information, see “Filtering Activity Plans, Targets, or Activities” on page 47.

- View a graphical representation of the plan in the Activity Plan Editor window.
- View the list of activities contained in the plan.
- Perform operations on activities, and activity plans such as cancel, pause, resume, and restart on all targets or on selected targets only. For more information, see “Controlling the Execution of Activity Plans and Activities” on page 51.
- For each activity, view the targets (managers, depots) assigned to it.
- Update submitted plans that have not yet started. For more information, see “Updating a Submitted Plan” on page 50.
- Generate recovery plans. For more information, see “Generating a Recovery Plan” on page 52.
- Restart an activity on an endpoint where the operation was unsuccessful.
- Delete the status information of a plan from the activity plan database.
- Launch the Distribution Status Console to monitor and control software distributions that have been submitted using the Activity Planner. For more information, see “Launching the Distribution Status Console” on page 53.

Activity Planner operations are performed at the times set in the activity plan, but under the control of the Scheduler in the Tivoli Management Framework. For more information about the Scheduler, refer to *Tivoli Management Framework: Planning for Deployment Guide* and *Tivoli Management Framework: User's Guide*.

Chapter 2. Performing Activity Planner Operations

This chapter describes how to use the Activity Planner through the Graphical User Interface (GUI).

Launching the Activity Planner GUIs

You launch the Activity Plan Editor and the Activity Plan Monitor GUIs from the Tivoli desktop.



On UNIX machines, launching either GUI displays a login dialog box. Enter the following command before launching the GUIs to enable the graphic session:

```
xhost hostname
```

In the login dialog box, you specify the following information:

1. In the **Host Machine** text box, type the name of the host machine on which the Activity Planner server is installed.
2. In the **Login as** text box, type an identified user account name for the specified host machine. Ensure that the login user account is a valid Tivoli administrator with the necessary Tivoli management region roles assigned.
3. In the **Password** text box, type the password associated with the specified login user. To use the Activity Plan Editor and the Activity Plan Monitor GUIs from the Tivoli desktop, the user password can contain all the special characters from ASCII 32 to ASCII 127. The only special character that cannot be used is "double quotes".

If the Tivoli management region and the managed node are located on different domains, you can start the Activity Plan Editor and Monitor components using the command line only. For more information about starting the GUIs from the command line, see "wapmgui" on page 94.

Note: On Windows machines, if the Tivoli desktop is not installed when you install Activity Planner or if you remove the Tivoli desktop after installing Activity Planner, the login dialog box is displayed for both GUIs and must be filled in as explained above.

Defining an Activity

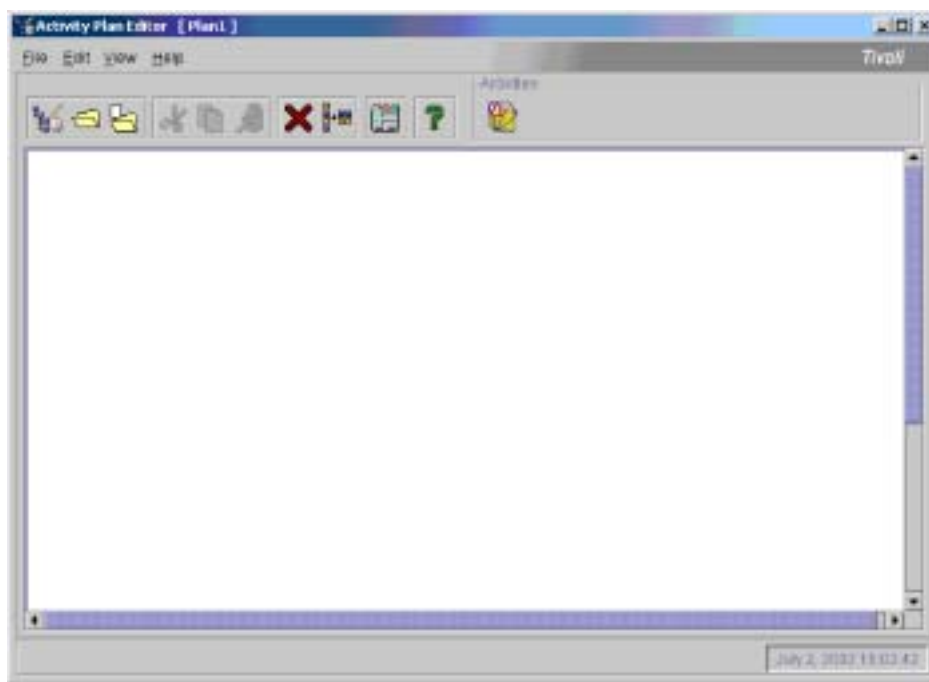
You define activities using the Activity Plan Editor GUI. When you launch the Activity Plan Editor from the Tivoli desktop the main window of the GUI displays an empty activity plan. You can define activities in the empty activity plan or you can open an existing plan from the **File** menu. The activities that you define are displayed as icons in the main window and arrows linking activities represent conditions among the activities. Depending on the plug-ins installed, one or more icons are displayed in the **Activities** toolbar.

To define an activity, you specify the following:

- The name of the activity
- A brief description of the activity (optional)
- The type of operation the activity will perform on a set of targets
- Properties related to the type of operation
- Targets of the activity (if not specified at the activity plan level)

For example, to create an activity that performs a specific operation, complete the following steps:

1. Double-click the **Activity Plan Editor** icon on the Tivoli desktop. The Activity Plan Editor main window is displayed.



2. Select the **Activities** icon. The Activity Properties dialog box is displayed.



3. If necessary, select the **Final Activity** check box. You select this to define the activity that is executed when all other activities have either completed or have been canceled.

Note: For final activities, conditioning is not permitted. For more information about conditioning, see “Conditioning the Execution of Activities” on page 33.

4. In the **Name of Activity** text box, type a name that identifies the activity in the activity plan, or use the default name provided. For activity names you can use alphanumeric characters plus the following special characters, but you cannot insert blank spaces:
 - number sign (#)
 - plus sign (+)
 - dash (-)
 - question mark (?)
 - asterisk (*)
 - open square bracket ([) and closed square bracket (])
 - period (.)
 - tilde (~)
 - underscore (_)
5. Optionally, enter a string of text that describes the activity in the **Description of Activity** text box.
6. From the **Operation** pull-down menu select the required operation. Supported operations may vary depending on the IBM Tivoli Configuration Manager component installed. For more information about supported operations, see the following sections:
 - “Software Distribution Operations” on page 22
 - “Inventory Operations” on page 24
 - “Pristine Manager Operations” on page 24

Tivoli Management Framework Task Library tasks are available if you installed IBM Tivoli Configuration Manager using the installation program, otherwise, you must register the plug-in as described in *Planning and Installation Guide*.

Activities that perform Tivoli Management Framework tasks are represented in the Activity Plan Editor main window by an icon that displays a timer, a schedule, and a pencil. For more information about Tivoli Management Framework Task Library tasks, refer to the *Tivoli Management Framework: User’s Guide*.

7. Select **Properties** to set the options for the activity.
8. Click **OK** to save the information and close the dialog box. An activity icon appears in the Activity Plan Editor window that represents the activity you have just defined.

Defining an Activity

Note: When adding new activities to a plan that are very similar to activities that you have already defined, you can copy and paste the activities into the same plan, or a different one.

Task Library Tasks

Tivoli Framework Task Library tasks are available if you installed IBM Tivoli Configuration Manager using the installation program, otherwise, you must register the plug-in as described in *Planning and Installation Guide*.

Defining a Task Library Task

This section explains how to define a Task Library task. For information about how to define an activity, see “Defining an Activity” on page 20. To define a Task Library task, complete the following steps:

1. In the Activity Properties dialog box, click the **Operation** drop-down list to display the list of supported operations. By default, the ExecuteTask operation is displayed.
2. Select **Properties** to set the task options for the activity. For more information about the settings in this dialog box, refer to the *Tivoli Management Framework: User's Guide*. Also, refer to the online help available from this dialog box.
3. In the **Task Name** text box, type the name of the task to be defined, or use the browse (...) button to browse your Tivoli environment and select a task from the list of available tasks in your profile manager.
4. Click **OK** to return to the Activity Properties dialog box.
5. In the Activity Properties dialog box, click **OK** to save the information and close the dialog box. An activity icon appears in the Activity Plan Editor window that represents the activity you have just defined.

Software Distribution Operations

After registering the Software Distribution plug-in, as described in *Planning and Installation Guide* you can create activities that perform the following Software Distribution operations:

- Install

Note: If an activity defined in the activity plan installs a software package with the roaming endpoints option selected, and you select to group the targets by gateway, the Activity Plan Monitor displays the original gateway of the endpoint and not the gateway to which it migrates.

- Remove
- Undo
- Accept
- Commit
- Load
- Unload

Note: When specifying the targets of an activity that performs a load or unload operation, you must use the managed node label and not the gateway label, or the operation will fail.

- **Data Moving operations**
- Send
- Retrieve
- Delete

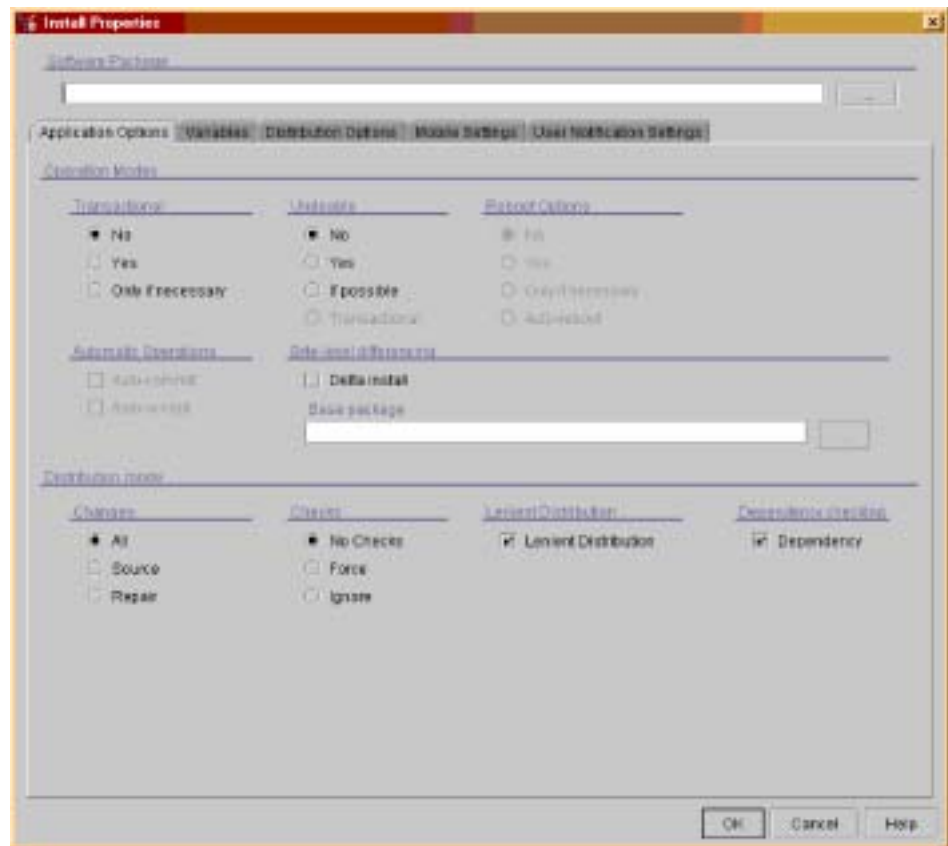
For information about the properties of these operations and their supported parameters, refer to the commands chapter in the *Reference Manual for Software Distribution* and to the *Reference Manual for Software Distribution*.

Defining a Software Distribution Activity

This section explains how to define an activity that performs an operation of a software package. For information about how to define an activity, see “Defining an Activity” on page 20.

For example, to define an install activity, complete the following steps:

1. In the Activity Properties dialog box, click the **Operation** drop-down list to display the supported Software Distribution operations, then select **Install**. See “Software Distribution Operations” on page 22 for a complete list of supported operations.
2. Select **Properties** to set the install options for the activity.



For more information about the settings in this dialog box, see *User's Guide for Software Distribution*. Also, refer to the *Reference Manual for Software Distribution* and the online help available from this dialog box.

3. In the **Software Package** text box, enter the name of the software package to be installed, or use the browse (...) button to browse your Tivoli environment and select a software package profile from the list of available software packages in your profile manager.



4. Click **OK** to return to the Activity Properties dialog box.
5. Click **OK** to save the information and close the dialog box. An activity icon appears in the Activity Plan Editor window that represents the activity you have just defined.

Inventory Operations

After registering the Inventory plug-in, as described in *Planning and Installation Guide*, you can create an Inventory scan operation.

Defining an Inventory Activity

This section explains how to define an activity that performs a scan operation. For information about how to define an activity, see “Defining an Activity” on page 20. To define a scan activity, complete the following steps:

1. In the Activity Properties dialog box, click the **Operation** drop-down list to display the list of supported operations. By default, the **Operation** drop-down list displays the Scan operation.
2. Select **Properties** to set the scan options for the activity. For more information about the settings in this dialog box, see *User's Guide for Inventory*. Also, refer to the online help available from this dialog box.
3. In the **Inventory Profile** text box, enter the name of the inventory profile to be used for the scan or use the browse (...) button to browse your Tivoli environment and select a profile from the list of available inventory profiles in your profile manager.
4. Click **OK** to return to the Activity Properties dialog box.
5. In the Activity Properties dialog box, click **OK** to save the information and close the dialog box. An activity icon appears in the Activity Plan Editor window that represents the activity you have just defined.

Pristine Manager Operations

After registering the Pristine Manager plug-in, as described in *Planning and Installation Guide*, and after doing the setup tasks, as described in “Setup Tasks” on page 233, you can create a Pristine Manager installation operation.

Defining a Pristine Manager Activity

1. In the Activity Properties dialog box, complete the fields. The **Operation** drop-down list displays the only supported operation: Install.
2. Click **Properties**. The Install Properties dialog is displayed. Enter the name of the operating system element to be included in the activity or click the browse (...) button. If you click the browse button, the Select Operating System dialog shows a list of operating system elements. Select an operating system element and click **OK**.
3. The Install Properties dialog displays your choice. Click **OK**.
4. In the Activity Properties dialog box, click **OK** to save the information and close the dialog box. An activity icon appears in the Activity Plan Editor window that represents the activity you have just defined.

Using Variables

You can use variables to express any attribute value of type *string* contained in the activity or activity plan. This makes them more generic for use on different target systems. For example, you can use the same plan more than once by modifying the relevant variables without modifying the plan itself. Otherwise, you can deploy an activity plan on different platforms by substituting the platform-specific information with variables.

Activity Planner supports both custom variables defined by the user, and built-in variables. You can use custom variables in all text fields when defining the Activity Properties. For more information about built-in variables, see “Using Variables to Specify Targets” on page 32.

To define a custom variable, perform the following steps:

1. Double-click the right button on one of the editable text boxes in any of the Activity Plan Editor dialog boxes to display the **Variables** dialog box.
2. Click **Add** to add an empty row to the **Variables** list.
3. Click the row under the **Name** column and type the name of the variable, such as `sp_list`.
4. Click the same row under the **Value** column and type, for example, a list of software packages, each separated by a comma: `sp1, sp2, sp3`. In this column you can also enter the string ‘Calculate at runtime’ and define the value for the variable afterwards in the **Plan Submission Parameters** notebook.
5. Click **OK**.

Note: When performing data moving operations, that is retrieve, delete, and send operations, you can use Activity Planner internal variables and data moving or Software Distribution variables. To distinguish between these variables, precede the Software Distribution variables with a \$ sign, for example `$(ep_label)`.

Selecting Targets for Activities and Activity Plans

You can specify targets using one or other of the following methods:

For each individual activity

If you assign targets for each individual activity, you cannot specify targets at the activity plan level.

Selecting Targets for Activities and Activity Plans

For the activity plan

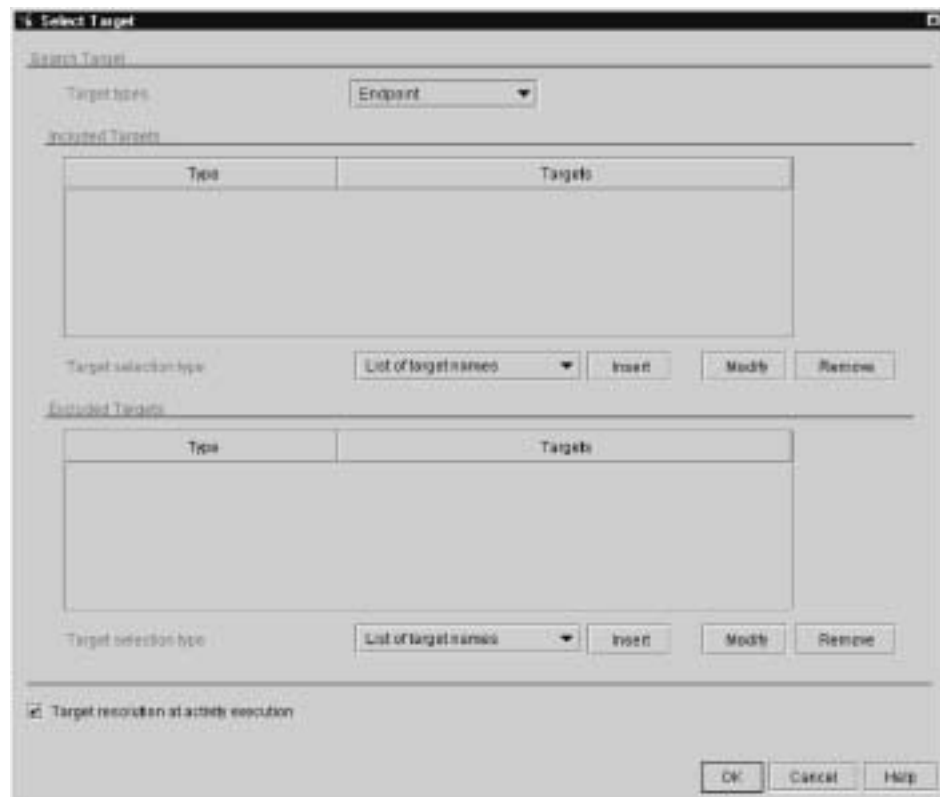
If you assign targets at the activity plan level, the targets apply to all of the contained activities and no other targets can be specified at the activity level.

For more information on selecting targets for activity plans, see “Selecting Targets for an Activity Plan” on page 42.

Selecting Targets for an Activity

To assign targets to an activity, perform the following steps:

1. Define an activity as explained in “Defining an Activity” on page 20.
2. Right-click the activity icon in the window and select **Targets** from the pop-up menu. The Select Target dialog box is displayed.



3. From the **Target types** drop-down list, select one of the mutually exclusive types of target the activity will be performed on.
 - Endpoint
 - User
 - Device
 - Managed node
 - Unknown

Note: The Unknown menu item is used only with plans and activities imported from the previous release, when no target type selection was available, while is not available for selection when creating plans.

4. From the **Target selection type** drop-down list, you can select one or more of the following selection types that are appropriate for the target type you selected in step 3:
 - A list of target names. Select this type if you define the targets using the \$(TARGET_LIST) variable.

- A file name containing a list of target names
- An inventory subscriber
- A query library subscriber
- A profile subscriber
- A directory query library subscriber
- A pristine target subscriber

If you select the **Target resolution at activity execution** check box, target names are resolved when the activity is performed, rather than when the plan is submitted. This means that the targets must be resolved when each activity contained in the plan is performed and this repeated process might decrease the Activity Planner performance. If this check box is cleared, the targets are resolved only once when the plan is submitted.

The **Target resolution at activity execution** check box should be selected only when necessary, for example if have a plan that comprises two activities, the first a task that creates a file containing a list of endpoints, the second an operation that addresses the targets defined in the previously generated file.

Note: If the **Target resolution at activity execution** check box is selected and variables are used to define targets, you can update targets after submitting the plan, as described in “Updating a Submitted Plan” on page 50.

Excluding Targets from an Activity

You can exclude targets at the activity level or at the activity plan level. However, if you exclude targets for each individual activity, you cannot exclude targets at the activity plan level. If you exclude targets at the activity plan level, the excluded targets apply to all of the contained activities, and no other targets can be excluded at the activity level.

To exclude targets from an activity, perform the following steps:

1. Define an activity as explained in “Defining an Activity” on page 20.
2. Right-click the activity icon in the main window and select **Targets** from the pop-up menu. The Select Target dialog box is displayed.
3. From the **Target selection type** drop-down list, you can select one or more of the following selection types that are appropriate for the target type you selected:
 - A list of target names
 - A file name containing a list of target names
 - An inventory subscriber
 - A query library subscriber
 - A profile subscriber
 - A directory query library subscriber
 - A pristine target subscriber

The specified targets are automatically checked against the targets specified in the **Included Targets** panel, and a differencing is performed. For example, if you specified a file name containing a list of targets, you can exclude some of the targets defined in the file without modifying the file itself.

Specifying a List of Target Names

You can specify a list of target names to be the targets of an activity. The available targets will vary according to the target type you selected in the **Target types** drop-down list. To specify the targets of an activity using a list of target names, perform the following steps:

Selecting Targets for an Activity

1. Select **List of target names** from the **Target selection type** drop-down list.
2. Click **Insert**. The Select Target dialog box is displayed.
3. From the Select Target dialog box, add the required targets in one of the following ways:
 - In the **Insert target** text box, type the name of the target you want to add in the **List of target names** box and click **Add**
 - Double-click the right button in the **Insert target** text box to display the Variables dialog box. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.

Notes:

- a. To select only those endpoints and users that are subscribers of the profile managers in your Tivoli environment, click the browse (...) button. This displays the Select target dialog box from which you can select those endpoints and users.
 - b. Click the **Target list** button to display all available endpoints and users in the List of Targets dialog box.
4. Click **OK** to save the changes and close the dialog box.

Specifying a File Name Containing a List of Targets

You can specify the targets of an activity plan by indicating the path to a file containing a list of target names. The target names must be either separated by a comma, or a space, or on a separated line. To specify a file name containing a list of targets, perform the following steps:

1. Select **File of target names** from the **Target selection type** drop-down list.
2. Click **Insert**. The Select Target dialog box is displayed.
3. From the Select Target dialog box, specify the managed node name where the file is stored in one of the following ways:
 - Type the name of the managed node in the **Managed node** text box. Browsing is enabled if the managed node you specify is on the local workstation and the name of the managed node is the same as the hostname.
 - Double-click the right button in the **Managed node** text field. The Variables dialog box is displayed from which you can specify targets. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.
4. From the Select Target dialog box, specify the file of target names in one of the following ways:
 - Browse for the file if the managed node you specified is the local workstation or type the fully qualified path of the file in the **Path to file** text box.
 - Double-click the right button in the **Path to file** text box to display the **Variables** dialog box. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.
5. You can optionally preview the contents of the selected file by clicking the **Preview** button.
6. Click **OK** to save the changes and close the dialog box.

Specifying an Inventory Subscriber as a Target

You can specify queries that access the configuration repository and select a list of targets as the query result. To specify an inventory subscriber as a target using a Structure Query Language (SQL) query, perform the following steps:

1. Select **Inventory Subscriber** from the **Target selection type** drop-down list.

2. Click **Insert**. The Select Target dialog box is displayed.
3. Click **SQL Queries** to display the Define SQL Queries dialog box.
4. Select the name of a table or view from the drop-down list to display a list of column names for the selected table.
5. Create an SQL search clause in the **where** box to specify what information the query should return. Use the table names, column names, and the operator buttons to create the SQL clause.
6. Click **Get result** to return the information defined in the SQL search clause.
7. Click **OK** to save the changes and close the dialog box.

Refer to the *Database Schema Reference* for more information about SQL queries.

Specifying a Query Library Subscriber as a Target

You can specify a query representing a query library to return a list of possible subscribers from the Inventory configuration database. To select a query library from the Tivoli Enterprise object database to retrieve subscribers, perform the following steps:

1. Select **Query Library Subscriber** from the **Target selection type** drop-down list.
2. Click **Insert**. The Select Target dialog box is displayed.
3. Click **SQL Queries** to display the Define SQL Queries dialog box.
4. Select a query library from the drop-down list. The contents of the query library are displayed in the list box.
5. Click **Get result** to retrieve the list of targets that satisfy the selected query library.
6. Click **OK** to save the changes and close the dialog box.

Note: If you create a custom query library, you must define at least the TME_OBJECT_LABEL and TME_OBJECT_ID fields, otherwise no result is returned.

Specifying a Directory Query Library Subscriber as a Target

You can specify a query representing a directory query library to return a list of possible subscribers from the Lightweight Directory Access Protocol (LDAP) database. To select a directory query library from the Tivoli object database to retrieve subscribers, perform the following steps:

1. Select **Directory Query Library Subscriber** from the **Target selection type** drop-down list.
2. Click **Insert**. The Select Target dialog box is displayed.
3. Click **LDAP Queries** to display the Define LDAP Queries dialog box.
4. Select a directory query library from the drop-down list and a query attribute, if necessary, in the Attributes text box. The selected attribute must contain data compatible with the target type selected in step 4 on page 26. The contents of the directory query library are displayed in the list box.
5. Click **Get result** to retrieve the list of targets that satisfy the selected directory query library.
6. Click **OK** to save the changes and close the dialog box.

Specifying a Profile Subscriber as a Target

You can specify profile subscribers as targets of an activity plan. Profile subscribers comprise profile managers, devices and users. The activities contained in the plan

Selecting Targets for an Activity

are performed on the subscribers associated with the specified profile subscriber. To specify a profile subscriber as the target of an activity plan, perform the following steps:

1. Select **Endpoint** or **Managed Node** from the **Target types** drop-down list
2. Select **Profile subscriber** from the **Target selection type** drop-down list.
3. Click **Insert**. The Select Target dialog box is displayed.
4. From the Select Target dialog list box, select the targets in one of the following ways:
 - Type the name of the profile subscriber in the **Profile subscriber** text box or click the browse (...) button to open the Select Profile Manager dialog box.
 - Double-click the right button in the **Profile Subscriber** text box to display the **Variables** dialog box. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.
5. In the Select Profile Subscriber dialog box, expand the policy regions in the left pane to display the profile subscribers.
6. From the **Search for Profile Subscriber** list box, add the required profile subscriber to the **List of Profile Subscriber** in one of the following ways:
 - Select a policy region and click the >> button to add all the profile subscribers contained in that policy region.
 - Click the switch to the left of the name of the required policy region. The policy region expands to show all the profile subscribers it contains.
 - Click the **Search** button to display the Search dialog box.
 - Select the required profile subscriber and click the > button.
7. To remove a profile subscriber, select the profile subscriber name and click **Remove**. The profile subscriber is removed from the list.
8. To clear the list, select **Clear**.
9. Click **OK** to return to the Select Target dialog box. The subscribers that you selected are added to the **Profile Subscriber** list.
10. In the Select Target dialog box, click **OK**.

Specifying a Resource Group of Pervasive Devices as a Target: You can specify resource group of pervasive devices as targets of an activity plan. The activities contained in the plan are performed on the pervasive devices associated with the specified resource group. To specify a resource group of pervasive devices as the target of an activity plan, perform the following steps:

1. Select **Device** from the **Target types** drop-down list
2. Select **Profile subscriber** from the **Target selection type** drop-down list.
3. Click **Insert**. The Select Target dialog box is displayed.
4. From the Select Target dialog box, select the targets in one of the following ways:
 - Type the name of the resource group of pervasive devices in the **Profile subscriber** text box or click the browse (...) button to open the Select Resource Group Device dialog box.
 - Double-click the right button in the **Profile subscriber** text box to display the **Variables** dialog box. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.
5. In the Select Resource Group Device dialog box, expand the policy regions in the left pane to display the profile subscribers.
6. From the **Search for Resource Group Device** list box, add the required profile manager to the **List of Resource Group Device** in one of the following ways:

- Select a policy region and click the >> button to add all the resource group of pervasive devices contained in that policy region.
 - Click the switch to the left of the name of the required policy region. From the expanded list, select the required group of pervasive devices and click the > button.
 - Click the **Search** button to display the Search dialog box.
7. To remove a resource group of pervasive devices, select the resource group name and click **Remove**.
 8. To clear the list, select **Clear**.
 9. Click **OK** to return to the Select Target dialog box. The resource group of pervasive devices that you selected are shown in the **Profile Subscriber** list.
 10. In the Select Target dialog box, click **OK** to save the changes and close the dialog box.

Specifying a Resource Group of Users as a Target: You can specify resource group of users as targets of an activity plan. The activities contained in the plan are performed on the subscribers associated with the specified resource group. To specify a resource group of users as the target of an activity plan, perform the following steps:

1. Select **User** from the **Target types** drop-down list
2. Select **Profile subscriber** from the **Target selection type** drop-down list.
3. Click **Insert**. The Select Target dialog box is displayed.
4. From the Select Target dialog box, select the targets in one of the following ways:
 - Type the name of the resource group of users in the **Profile subscriber** text box or click the browse (...) button to open the Select Resource Group User dialog box.
 - Double-click the right button in the **Profile subscriber** text box to display the **Variables** dialog box. See “Using Variables to Specify Targets” on page 32 for more information about defining targets using variables.
5. In the Select Resource Group User dialog box, expand the policy regions in the left pane to display the profile subscribers.
6. From the **Search for Resource Group User** list box, add the required profile manager to the **List of Resource Group User** in one of the following ways:
 - Select a policy region and click the >> button to add all the resource group of users contained in that policy region.
 - Click the switch to the left of the name of the required policy region. From the expanded list, select the required group of users and click the > button.
 - Click the **Search** button to display the Search dialog box.
7. To remove a resource group of users, select the resource group name and click **Remove**. The resource group is removed from the list.
8. To clear the list, select **Clear**.
9. Click **OK** to save the changes and return to the Select Target dialog box. The resource group of users that you selected are shown in the **Profile Subscriber** list.
10. Click **OK** to save the changes and close the dialog box.

Specifying a Pristine Target Subscriber

You can specify pristine target subscribers as targets of an activity plan. The activities contained in the plan are performed on the pristine machines; however,

Selecting Targets for an Activity

the first activity must be a pristine installation. After the target machines have operating systems installed, you can make them the target of other activities.

To specify a pristine target subscriber as the target of an activity plan, perform the following steps:

1. Select **Endpoint** from the **Target types** drop-down list
2. Select **Pristine Target Subscriber** from the **Target selection type** drop-down list.
3. Click **Insert**. The Select Pristine Target dialog box is displayed.
4. From the Select Pristine Target dialog list box, select the targets in one of the following ways:
 - Type the name of the pristine target in the **Insert target** text box.
 - Click the browse (...) button to open the Machine Manager dialog box:
 - a. Click **Refresh** to display all of the machines or filter them either by group or server. See “Filtering Machines in the List” on page 242 for more information.
 - b. Select one or more machines from the list and click **OK**.
5. On the Select Pristine Target dialog box, click **Add** for each target. The target appears in the list. To remove a pristine target, select it and click **Remove**.
6. Click **OK** to return to the Select Target dialog box. The subscribers that you selected are added to the **Included Targets** list.
7. In the Select Target dialog box, click **OK**.

Using Variables to Specify Targets

You can specify targets using variables. The list of targets is automatically evaluated at execution time. To display the **Variables** dialog box, double-click the right button in the target text field available in each **Select target** dialog box, as described above. The following list describes the available built-in variables:

\$(TARGET_LIST)

The target names associated with the variable are specified at submission time.

\$(DEPOT_LIST)

The targets are the managed nodes configured as depots to which the endpoints, specified in the \$(TARGET_LIST) variable, are assigned. This variable is calculated automatically when the \$(TARGET_LIST) variable is defined.

\$(ORIGINATOR)

The hostname of the machine from which the activity plan is submitted. This variable is calculated automatically.

You can also define any custom variables as necessary. For more information about defining custom variables, see “Using Variables” on page 25.

If you are performing operations that are not addressed to depots, you can skip the resolution of the variable \$(DEPOT_LIST) by creating a file containing a list of target names and specifying the file name as a custom variable. After saving the plan as a template, you can submit it using the Activity Plan Monitor GUI by specifying the same variable in the **Variables** text field in the **Plan Submission Parameters** dialog box, as described in “Defining the Variable Values” on page 50.

You can also submit the plan using the **wsubpln** command specifying the file name with the **-V** option. For more information on the **wsubpln** command, see “wsubpln” on page 122.

Conditioning the Execution of Activities

The execution of an activity can depend on the result of the execution of another activity or activities in the same activity plan.

You can condition an activity based on the result of the conditioning activity (Completion, Success, Failure), by the targets on which the specified result must occur (All, Target, Depot) and you can use Boolean operators (AND | OR) to add further flexibility to the condition.

Assume that an activity plan consists of two activities, Activity A and Activity B. A condition can be set on Activity B, the conditioned activity, related to the outcome of the execution of Activity A, the conditioning activity, that dictates when Activity B is executed on its targets. You can condition the result of the execution of Activity A on all its targets using one of the following classifications:

Completion

Activity B is performed when the operation defined in Activity A completes, regardless of whether it completes successfully or with errors.

Success

Activity B is performed only if the operation defined in Activity A completes successfully with no errors.

Failure

The execution of Activity B is performed if the operation defined in Activity A fails with an error.

In addition to conditioning the execution of Activity B based on the result of Activity A, you can specify the targets on which the specified result must occur. You can specify one of the following:

All The execution of Activity B depends on the execution of Activity A on all of the targets defined in Activity A. The operation specified in Activity B is executed when Activity A has completed execution on all its targets. A completion percentage of targets can be specified. For example, you can specify to execute Activity B when Activity A has completed executing on 50% of its targets.

Target The execution of Activity B on target X depends on the execution of Activity A on target X. The operation specified in Activity B is executed on target X when Activity A has completed execution on target X, without waiting for the remaining targets to complete. This kind of conditioning can decrease product performance; in this case, it is advisable to manage conditions by creating more efficient software packages.

Notes:

1. If you have the Software Distribution plug-in installed, and define a send operation from endpoint to endpoint in an activity, then conditioning by target capability is not available.
2. Conditioning by target is not supported for users and devices.
3. Inventory does not support conditioning by target.

Depot The execution of Activity B on a set of targets sharing the same depot depends on the execution of Activity A on the specified depot. The

Conditioning the Execution of Activities

operation specified in Activity B is executed on target X when Activity A has completed execution on the specified depot. Conditioning by depot is available only if the conditioning activity is a Load, Unload, or Task Library activity, and if the conditioned activity is addressed to endpoints sharing the specified depot.

Note: Conditioning by depot is supported only if the gateway also serves as a depot.

The following example depicts the sequence of events that take place during the execution of Activity A and Activity B, where Activity A is the conditioning activity and Activity B is the conditioned activity, as described in the diagram and in the following text.

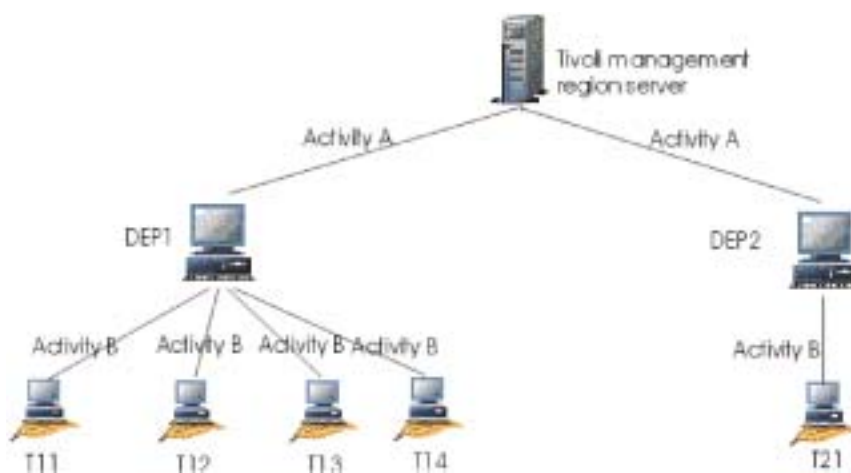


Figure 1. Conditioning activities

Activity B is conditioned by Activity A by completion and by depot so that the condition attribute in the activity stanza for Activity B is expressed as follows:

```
<condition> CD(Activity A)</condition>
```

Activity A has the following managed nodes as targets: DEP1 and DEP2. Activity B has the following endpoints as targets: T11, T12, T13, T14 and T21. Endpoints T11, T12, T13, T14 are assigned to depot DEP1, and endpoint T21 is assigned to depot DEP2.

Assuming Activity A has completed on all its managed nodes, Activity B can start on all endpoints. If Activity A does not complete on DEP2, Activity B cannot start on T21. This type of conditioning enhances network performance by isolating targets associated with a particular depot that experience difficulties, and enabling other targets, with different depots, to continue.

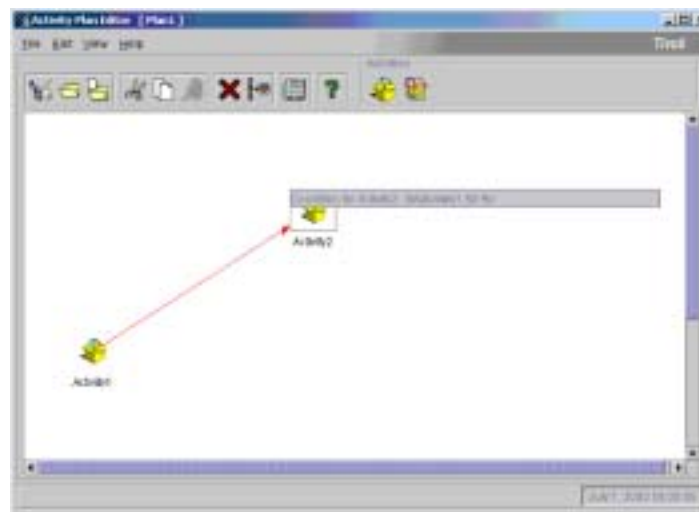
To condition the execution of an activity on the successful outcome of another activity on 50% of all its targets, perform the following steps:

1. From the Activity Plan Editor window, create two activities as explained in “Defining an Activity” on page 20.
2. Right-click the activity on which you want to set the condition and select **Condition** from the pop-up menu. The Condition dialog box is displayed.
3. Select **Successful-All** from the drop-down list.

4. Double-click the conditioning activity from the **Select an Activity** list in the plan. The selected activity is displayed in the Condition for text field.
5. Click the down arrow and scroll to select **50%**.



6. Click **OK** to save the information and close the dialog box. The condition is represented by a red line connecting the activities in the Activity Plan Editor main window.



When specifying a condition using a percentage, the conditioned activity might not start when that percentage is reached. This might occur where a single report for the completion of an operation groups many targets together. Even if the number of targets required to satisfy the percentage has been reached, if the report is not sent to the Activity Planner engine promptly, the conditioned activity will not start.

For Software Distribution activities, you can modify the MDist 2 **notify_interval** parameter of the gateway to return the reports more frequently. Refer to the *Tivoli Management Framework: Reference Manual* for more information about modifying MDist 2 parameters using the **wmdist** command. Note that setting the **notify_interval** parameter to a small interval can decrease product performance.

In addition to the result and target parameters, you can specify the AND and OR Boolean operators to add further flexibility to the condition, as described below:

- AND** If one or more of the conditions is not met, the activity is set to **Canceled by condition**.
- OR** If at least one of the conditions is met, the activity is successful.

Conditioning the Execution of Activities

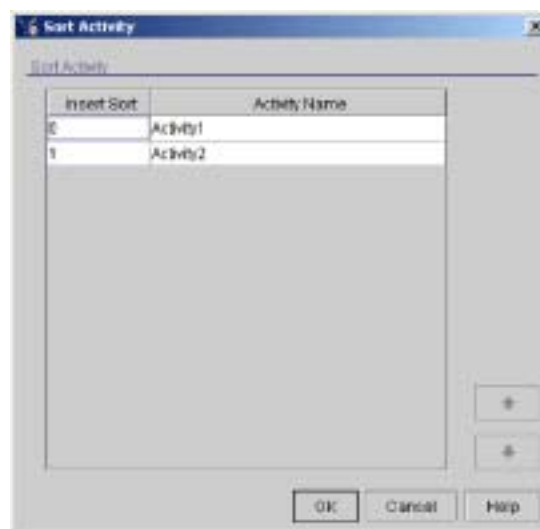
If some targets are included in the target list of the conditioned activity but are not included in the target list of the conditioning activity, by default they are started when the conditioned activity starts. To change this behavior, modify the **cancel_unique_targets** parameter in the apm.ini file. For more information on this parameter, see “Defining the Activity Planner Engine Parameters” on page 9.

Sorting Activities in Order of Execution

You can sort activities to specify the order in which they are submitted by the Activity Planner engine. Only activities without conditions can be sorted. Conditioned activities have a predefined order of execution and cannot be modified using the Activity Sort dialog box. For more information on applying conditions to activities, see “Using Conditions” on page 66.

To sort the activities contained in an activity plan in order of execution, perform the following steps:

1. From the Activity Plan Editor main window, select **Edit»Sort Activities**. The Sort Activity dialog box is displayed.



2. Select an activity in the list and click the up or down arrow to rearrange the order.
3. Click **OK** to save the changes and close the dialog box.

Scheduling the Execution of Activities

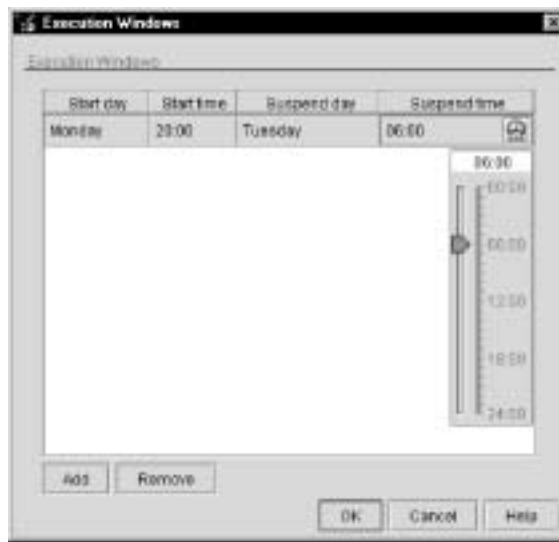
You can schedule activities to run on specific days of the week and within a specified time frame, such as, only during the day, during the night, on weekdays, or weekends. The execution window enables you to specify a time frame, at the activity level, within which the activity is to be executed. You can specify more than one execution window for each activity. The time refers to the time on the managed node where the plan is created. This function is available only for the plug-ins that support pausing and restarting activities.

Notes:

1. An execution window must have a duration of at least five minutes.
2. The interval between execution windows must not be less than five minutes.

For example, to limit the execution of an activity to start on Monday evenings at 8 p.m. until Tuesday morning at 6 a.m., perform the following steps:

1. Right-click an activity in the Activity Plan Editor window and select **Execution Windows**. The Execution Windows dialog box is displayed.
2. Click **Add** to enter the execution time within which the selected activity must be executed. A row is added in the dialog box.
3. Double-click the row under the **Start day** column and select **Monday** from the drop-down list.
4. Double-click the same row under the **Start time** column and a clock icon appears. Click the clock icon to display a 24-hour slide bar. Slide the slider to the **20:00** mark to set the start time of the execution for 8 p.m. Click above or below the slider to decrement or increment the time in minutes.
5. Double-click the same row under the **Suspend day** column and select **Tuesday** from the drop-down list.
6. Double-click the same row under the **Suspend time** column and a clock icon appears. Click the clock icon and using the slide bar, set **06:00** as the suspend time.



7. Click **OK** to save the information and close the dialog box.

The activity can be executed only within the specified execution time, Monday from 8 p.m. through Tuesday at 6 a.m. If the activity is running and has not completed by the suspend day and time, the activity is paused and is resumed at the next specified execution time, that is, the following Monday at 8 p.m.

Defining the General Properties of the Activity Plan

An activity plan contains one or more activities to be executed on specified target systems. Activity plans are executed consecutively according to their execution time and priority. The activities of all plans in submission that are ready to be executed are queued up and performed in order of priority.

To define general information about the activity plan, perform the following steps:

1. From the main window of the Activity Plan Editor, select the **Properties** icon from the toolbar. The Plan Properties notebook is displayed. The General page is displayed by default.

Defining the General Properties of the Activity Plan

2. Replace the default name in the **Name** text box and optionally enter a description in the **Description** text box.
3. Set a priority on the activity plan by clicking the **Priority** drop-down list and selecting from the **High**, **Medium**, and **Low** options. The priority setting is the same for all activities contained in the same plan.
4. Specify which of the notification options to use when the execution of the activity plan is complete. The notification options are:
 - E-mail
 - Posting a notice to the Activity planner Version 4.3.1 notice group, created during the installation of the product
5. Enter an e-mail address in the **Mail-ID** text box. You can specify more than one address by separating the addresses with a comma. For information about configuring a mail server to correctly route e-mail, refer to the *IBM Tivoli Enterprise Console: Installation Guide*.
The **Post notice** check box is selected by default.
6. Select the **Submit paused** check box to submit the plan in paused state. The plan is submitted when it is released from the paused state.
7. Select the **Notification date** check box to have Activity Planner send the administrator a message on the status of the plan. If you select this check box, a notification message is sent to the e-mail address specified in the **Mail ID** text field, or logged in the Activity Planner Notice group at the specified date and time.
8. Click **OK** to save the information and close the dialog box, or click on a tab to define more settings.

Scheduling the Activity Plan

You can schedule to execute an activity plan one time only, or you can specify that the activity plan be executed on certain days of the week, month, or at a date or time interval. The timed operations handled by the Scheduler are as follows:

- The start not before time
- The complete not after time
- The generation of new instances in a recursive plan
- The start at date and time of the execution window for the activity
- The suspend at date and time of the activity execution window for the activity

For recursive plans, the Scheduler times the generation of each instance of the plan. However, deleting a job from the Scheduler that represents an instance of a recursive plan, discontinues the recursion of the plan and no further instances are generated. For more information about the Scheduler, refer to *Tivoli Management Framework: Planning for Deployment Guide* and *Tivoli Management Framework: User's Guide*.

If an execution time is not specified for the activity plan, the activity plan is performed at the time it is submitted.

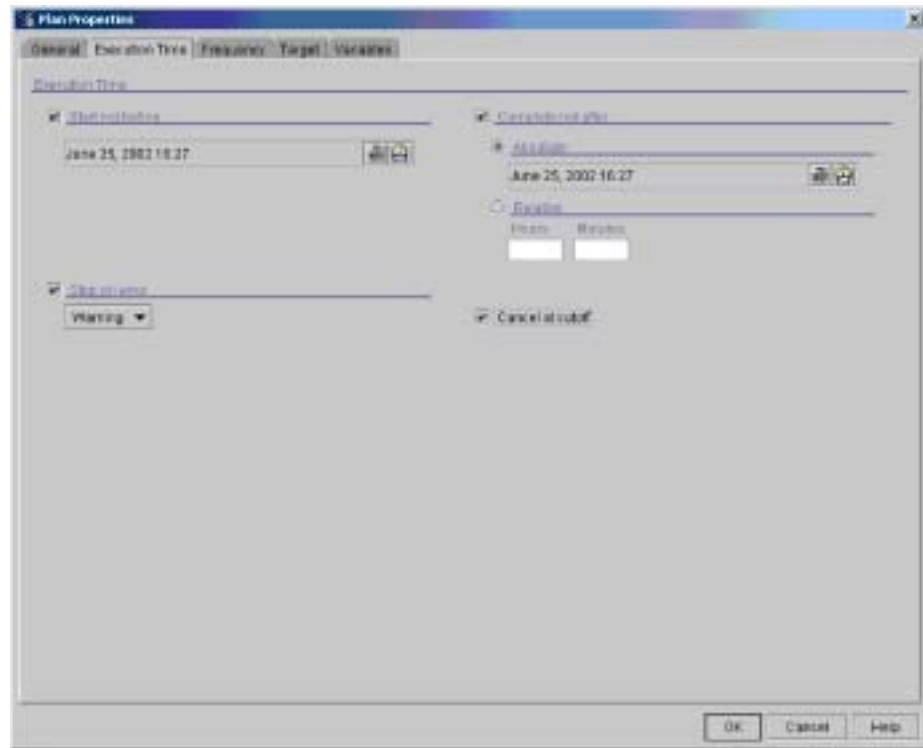
Scheduling Plans to Execute within a Time Interval

You can schedule an activity plan to start not before and complete not after a specified time. If the execution of the plan has not completed by the specified complete not after time, you can optionally cancel any outstanding activities and stop the execution of the plan. The time reference refers to the time on the local machine where the plan is created.

Defining the General Properties of the Activity Plan

For example, to schedule an activity plan to be executed not before October 20, 2002 at 7:00 a.m. and to complete by December 31, 2002 at 12 midnight, perform the following steps:

1. From the Plan Properties notebook, select the **Execution Time** tab.
2. Select the **Start not before** check box. The current date and time are displayed.
3. Click the Calendar icon and set the date in the calendar displayed.



4. To modify the time, click the clock icon then use the slider to set the time. Click above or below the slider to decrement or increment the time in minutes.
5. Select the **Complete not after** check box and set the date and time to December 31, 2002 at 12 midnight. You can also set a relative time, stating the **Complete not after** time based on the number of hours since the plan started.

Note: If the plan is recursive, you can only set a relative time.

If the activity plan does not complete within the complete not after time you specified, any outstanding activities not yet executed are canceled. To cancel all activities in execution, select the **Cancel at cutoff** check box. In this case, activities in execution when the **Complete not after** time is reached are canceled.

If the plan contains activities which cannot be canceled, that is Tivoli Framework tasks or Software Distribution operations addressed to devices or users, a warning message is displayed. If you choose to submit the plan, if the **Complete not after** time expires, all activities are canceled with the exception of Tivoli Framework tasks and Software Distribution operations addressed to devices or users.

If you do not select this check box, the **Complete not after** time specified is ignored for those activities already being executed.

6. To cause an activity plan to stop if an error occurs during the execution of an activity, select the **Stop on error** drop-down list and select **Warning**, **Error**, or

Defining the General Properties of the Activity Plan

Fatal as appropriate from the drop-down list. If the activity completes with an error level equal to or greater than that specified, the activity plan is stopped.

Note: If you register the plug-in for Software Distribution, set the **Stop on error** option to **warning** to ensure that the plan stops if an error occurs, as the Software Distribution return codes differ from those generated internally by Activity Planner. Alternatively, use the **wseterrlev** command to map a return code of 1 to either **error** or **fatal**.

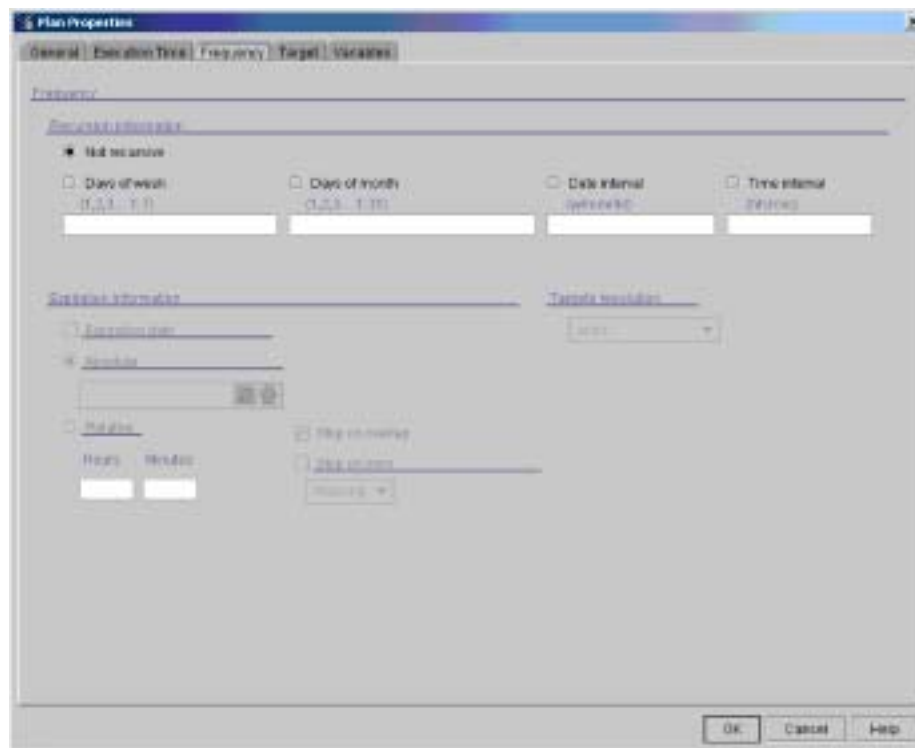
See “wseterrlev” on page 115 for information about mapping return codes with error levels.

Scheduling Activity Plans to Run Periodically

An activity plan can be scheduled to run more than once. You can specify to repeat the execution of a plan in days, months, at a specific date interval, or a time interval. When you specify to execute an activity plan recursively, each time the plan is executed, an instance of the plan is submitted for execution. The reference point from which the recursion begins is the *start not before* time specified on the Execution Time page. See “Scheduling Plans to Execute within a Time Interval” on page 38 for information about setting the start not before time. If the start not before time is not specified, the point of reference is the time the plan is submitted. The time reference is the time on the local machine where the plan is created.

For example, to schedule an activity plan to run every Tuesday, perform the following steps:

1. From the main window of the Activity Plan Editor, select the Properties icon from the toolbar. The Plan Properties notebook is displayed.
2. Select the **Frequency** tab. The Frequency page is displayed.



3. Select the **Days of the week** radio button and type **3** in the text box. The days of the week are expressed by the numbers 1 through 7, where 1 represents Sunday.

4. To dynamically recalculate the list of targets each time an instance of the activity plan is submitted, select **Dynamic** from the **Targets resolution** drop-down list. If you select **Static**, the list of targets calculated when the first instance of the plan is submitted is used for each successive instance submitted.

Note: If you select the **Days of month** radio button and specify a day that is not available in the current month, for example April 31, the plan instance will be executed on the last day of the current month, that is April 30.

Interrupting the Recursion of an Activity Plan

You can interrupt the recursion of an activity plan in any the following ways:

- Specifying an expiration date on which to stop the recursion.
- Specifying to stop the recursion when an instance completes in error.
- Specifying to stop the recursion if instances overlap.
- Deleting the job from the Tivoli Management Framework Scheduler.
- Using the **wcntpln** command. For more information on this command, see “wcntpln” on page 98.

You can discontinue the recursion mechanism by checking the **Expiration date** check box and specify a date on which to stop the recursive submission of an activity plan instance by performing the following steps:

1. Select the **Frequency** tab.
2. In the Frequency page, select the **Expiration date** check box.
3. Click the calendar and clock icons and set the date and time respectively, to the required expiration date. You can also set a relative time, giving an expiration date based on the number of hours since the start of the plan. If an expiration date is not specified, from the command line interface, use the **wcntpln** command to stop the recursion mechanism. For more information on this command, see “wcntpln” on page 98.

To specify an error level to stop the recursion when an error occurs, perform the following steps:

1. Select the **Frequency** tab.
2. In the Frequency page, from the **Stop on error** drop-down list, select the required error level. The recursion stops if the plan executes with an error equal to or greater than the value you specify.

Note: If you register the plug-in for Software Distribution, set the **Stop on error** option to **warning** to ensure that the plan stops if an error occurs, as the Software Distribution return codes differ from those generated internally by Activity Planner. Alternatively, use the **wseterrlev** command to map a return code of **1** to either **error** or **fatal**.

See “wseterrlev” on page 115 for information about mapping return codes with error levels.

No two instances of the same recursive plan can be active at the same time. For example, if Instance A of the recursive plan is currently in execution and Instance B is due to be executed, you can choose to do one of the following:

- Leave the **Stop on overlap** check box cleared to continue with Instance A and skip Instance B and all successive instances (Instance C, D, E, and so on) until Instance A has completed. Instance B is executed at the next scheduled repeat following the completion of Instance A.

Defining the General Properties of the Activity Plan

- Select the **Stop on overlap** check box to allow Instance A (the instance actively in execution) to complete but to discontinue the recursion mechanism.

Selecting Targets for an Activity Plan

You can specify targets at the activity plan level or at the activity level. If the targets are specified at the activity plan level, then the operations defined in the activities are executed on these targets. Targets cannot be specified at the activity level if they have been specified at the activity plan level. See “Selecting Targets for an Activity” on page 26 for more information.

To assign targets to the activity plan, perform the following steps:

1. From the main window of the Activity Plan Editor, select the Properties icon from the toolbar. The Plan Properties dialog box is displayed.
2. Select the **Target** tab. The Target page is displayed.
3. Select how to specify the targets of an activity plan. You can choose one or more of the following ways:
 - A list of target names. See “Specifying a List of Target Names” on page 27.
 - A file name containing a list of target names. See “Specifying a File Name Containing a List of Targets” on page 28.
 - An inventory subscriber. See “Specifying an Inventory Subscriber as a Target” on page 28.
 - A query library subscriber. See “Specifying a Query Library Subscriber as a Target” on page 29.
 - A profile subscriber. See “Specifying a Profile Subscriber as a Target” on page 29.
 - A pristine target subscriber. See “Specifying a Pristine Target Subscriber” on page 31.
 - A variable. See “Using Variables to Specify Targets” on page 32.

If you have a large number of targets (50,000 or more), you can improve network utilization by creating a number of different activity plans for specific target groups. For example, if you have 50,000 targets, you can create 10 activity plans for separate groups of 5,000 targets. The plan submission time increases linearly with the number of activities contained in the plan.

You can then submit these activity plans with a time interval which is less than or equal to the database insertion time for each plan. The maximum number of targets for one activity plan should not exceed 25,000.

You can also split the Activity Planner workload on several spoke Tivoli regions. For more information on hub and spoke Tivoli regions, refer to *Tivoli Management Framework: Planning for Deployment Guide*.

Note: If you install Activity Planner on more than one region, you need to create a separate Activity Planner database for each installation. These databases cannot communicate with each other and cannot share information on activity plans. To work around this problem, you should write a script to extract data from each Activity Planner database and collect it at a central location, typically, the Hub region.”

If you select the **Target resolution at activity execution** check box, target names are resolved when the activity is performed, which means one of the following cases:

- If the activity is conditioned, when the first report generated by one of the conditioning activities is managed by Activity Planner.
- If the activity is not conditioned, when the activity starts.

rather than when the plan is submitted.

This means that the targets must be resolved when each activity contained in the plan is performed and this repeated process might decrease the Activity Planner performance. If this check box is cleared, the targets are resolved only once when the plan is submitted.

The **Target resolution at activity execution** check box should be selected only when necessary, for example if have a plan that comprises two activities, the first a task that creates a file containing a list of endpoints, the second an operation that addresses the targets defined in the previously generated file.

Note: If the **Target resolution at activity execution** check box is selected and variables are used to define targets, you can update targets after submitting the plan, as described in “Updating a Submitted Plan” on page 50.

Loading an Activity Plan

You can load an activity plan template or draft from the activity plan database or open a file in XML format. When loading an activity plan in XML format, you can change the encoding type to one of the encodings supported by the Java Runtime Environment version installed. For more information about supported encoding types, see the Sun Microsystems web site. To open an activity plan, select the **Open** option from the **File** menu.

When activity plans (drafts, templates) are in use, they are automatically locked to avoid other users accessing the same plan and overwriting it. The plan unlocks when you open a new plan or exit the Activity Plan Editor GUI.

Saving the Activity Plan

Saving an activity plan saves it directly to the activity plan database or to an XML file. To save an activity plan prepared using the Activity Plan Editor, select the **Save as** option from the **File** menu. Select the required option:

Draft To save option save the activity plan as a draft in the activity plan database.

Template

To save the activity plan as a template in the activity plan database.

XML file

To open the Save as XML file dialog box.

Note: You can also export an activity plan into XML format from the CLI. For information about this, see “wexppln” on page 102.

Only activity plans that you store as templates can be accessed from the Activity Plan Monitor GUI and submitted for execution. Using the Activity Plan Editor, you can reopen saved activity plan templates and drafts and modify them and save them back to the database.

Deleting Activity Plans or Activities

You can delete an activity plan draft or template that has been saved to the activity plan database by performing the following steps.

1. In the Activity Plan Editor, select **Edit» Delete Plan**. The Delete Plan dialog box displays a list of draft plans and plan templates saved to the activity plan database.
2. Select a draft plan or template plan from the Delete Plan dialog box.
3. Click **Delete Plan**.
4. Click **OK** to save the changes and close the dialog box.

Notes:

1. This procedure deletes the template from the activity plan database, but not submitted plans and their status. To delete submitted plans and their status, use the Activity Plan Monitor GUI or the **wdelstat** command explained in the *Reference Manual for Software Distribution*. See “Deleting Completed Activity Plans” on page 53.
2. You can also delete an activity plan from the CLI. For information about this, see “wdepln” on page 100.

To delete an activity from an activity plan that is currently in preparation, perform the following steps:

1. From the Activity Plan Editor window, select the activity to be deleted.
2. From the **Edit** menu, select **Delete Activity**.

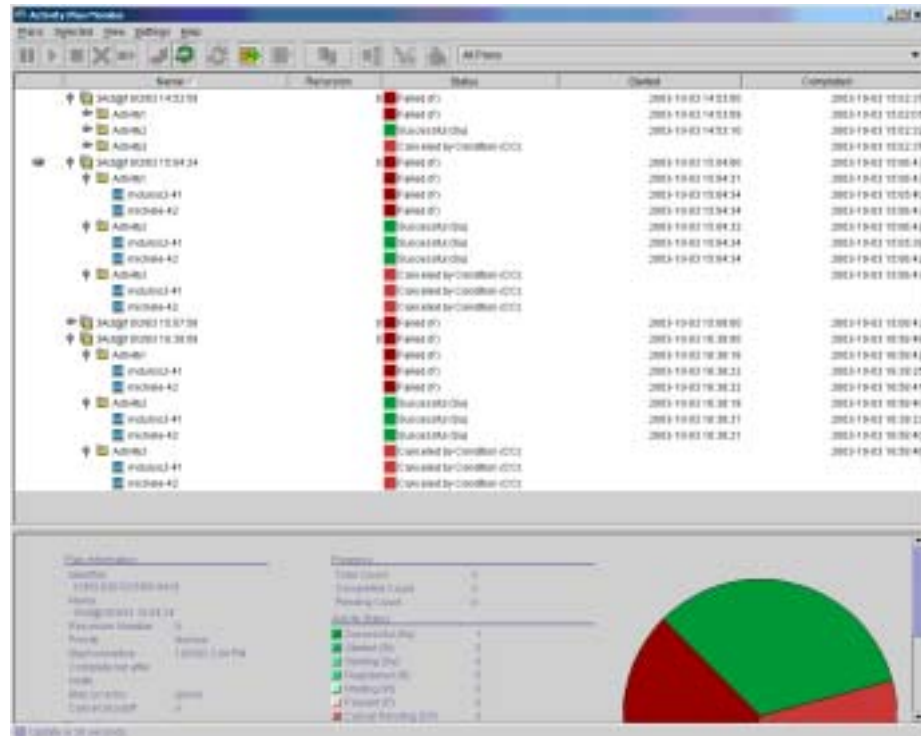
Submitting and Monitoring Activity Plans

The Activity Plan Monitor window displays a list of all submitted activity plans and their current status in table format. Each table row contains an expandable activity plan and each column displays a particular activity plan attribute.

The upper pane of the window displays a list of submitted activity plans. In this view, the following information is displayed:

- A list of submitted plans
- The activities contained within each plan
- Target information
- Recursion information
- Status of the plan
- Status of each activity in the plan
- Status of an activity on a particular target
- Start date and time of the plan and of the activity
- Complete date and time of the plan and of the activity completed

To view the activities contained within an activity plan, click the expand icon to the left of the plan name. To view the targets assigned to the activity and the status of the activity on that particular target, click the expand icon to the left of the activity. You can also apply filters to the Activity Plan Monitor view, and optionally select a filter to be used as default. For more information on defining filters, see “Filtering Activity Plans, Targets, or Activities” on page 47.



To view a graphical representation of the activity plan in the lower pane of the Activity Plan Monitor window, perform the following steps:

1. Select a plan from the Activity Plan Monitor window.
2. Select **View >> Activity Plan** from the menu.

Notes:

1. If an activity defined in the activity plan installs, for example, a software package with the roaming endpoints option selected, and you select to group the targets by manager, the Activity Plan Monitor displays the original manager of the endpoint and not the manager to which it migrates.
2. If the **Ignore** option is selected in a change management operation and the operation is performed successfully, the distribution returns a success code for all targets. This includes targets where the distribution has not taken place because they did not pass the verification phase. To view the targets that received the distribution, select a target and choose **Show Details** in the **Selected** menu.

For more information about the **Ignore** option, see the *User's Guide for Software Distribution* and the *Reference Manual for Software Distribution*.

3. Due to message length constraints, some messages displayed in the lower pane may be truncated. For more information on the message, refer to *Messages and Codes* and search for the message by its ID number.

Activity Plan Status

Activity plans, activities, and targets can have several intermediate status. The same status may have different meanings depending on whether it applies to activity plans, activities, or targets. The following tables describe the statuses and their meanings.

Submitting and Monitoring Activity Plans

Table 2. Target statuses

Status	Meaning
Successful	The activity or activity plan has completed successfully on the target
Started	The activity or activity plan is ongoing
Waiting	The activity or activity plan is waiting to be submitted to the target
Canceled by condition	The target has been canceled as a condition on the activity has not been fulfilled
Paused	The target is paused
Cancel pending	A cancel command has been submitted, and the Activity Planner engine is waiting for the related reports
Canceled	The target has been canceled
Failed	The target has failed
Registered	The operation was registered with the plug-in but has not started on the target. This status is available only when conditioning by target or by depot is applied

Table 3. Activity statuses

Status	Meaning
Starting	The plug-in for the activity is calling the related application, for example, the install operation for a Software Distribution activity
Successful	The activity has completed successfully
Started	The activity is ongoing
Waiting	The activity is waiting to be submitted to the application
Canceled by condition	The activity has been canceled as a condition on the activity has not been fulfilled
Paused	The activity is paused
Cancel pending	A cancel command has been submitted, and the Activity Planner engine is waiting for the related reports
Canceled	The activity is canceled
Failed	At least one target has failed
Registered	The operation was registered with the plug-in but has not started on any targets. This status is available only when conditioning by target or by depot is applied

Table 4. Activity plan statuses

Status	Meaning
Starting	All activities in the plan are in Starting status

Table 4. Activity plan statuses (continued)

Status	Meaning
Successful	At least one activity in the plan is in Successful state. Any remaining activities are in Canceled or Canceled by Condition state
Started	The activity plan is ongoing
Waiting	The activity plan is waiting to be submitted to the application
Paused	The activity plan is paused
Cancel pending	A cancel command has been submitted, and the Activity Planner engine is waiting for the related reports
Canceled	The activity plan is canceled
Failed	At least one activity in the plan has failed

The final status of an activity plan can be any one of the following:

- Successful
- Failed
- Canceled

To view detailed information about an activity plan, including its status, select the plan in the upper pane of the window and the pie chart icon on the toolbar. The lower pane displays the detailed information in list form as a pie chart.

To view information about a specific activity or an activity on a specific target, select the activity or target in the top pane and click the pie chart icon on the toolbar. The upper pane displays detailed information about the selected item in the top pane.

Note: You can also view the status of an activity plan from the CLI. For information about this, see “wmonpln” on page 110.

Filtering Activity Plans, Targets, or Activities

You can specify one or more filters to be applied to plans, targets, or activities so that only the objects matching certain criteria are displayed in the Activity Plan Monitor graphical interface or command line. You can also modify or delete an existing filter.

To manage a filter to be applied to plans, targets, or activities, perform the following steps:

1. Select **Settings >> Filter Management** from the menu of the Activity Plan Monitor window. The Filter Management dialog box is displayed.

Submitting and Monitoring Activity Plans

2. Specify a new filter name in the Filter name text field, or select an existing filter from the pull-down menu.
3. Specify the object the filter applies to in the **Apply filter on** pull-down menu.
4. Define one or more filter criteria, and specify whether you want the filter to be used as default.
5. When you have finished, you can perform one of the following operations:
 - Click **Save filter** to save the filter in the Activity Planner database.
 - Click **Delete filter** to delete the specified filter.
 - Click **Apply** to apply the filter immediately. If you click **Apply**, the filter is not saved in the Activity Planner database, but is applied only temporarily, and deleted when you close the Activity Plan Monitor GUI.

Note: You can also manage filters from the CLI. For more information on this, see “wapmfltr” on page 91.

You can filter objects based on one or more of the following criteria:

- Name
- Recursion settings
- Status
- Start date
- Completion date

The filters for the user currently logged on are displayed in the Activity Plan Monitor main window in a pull-down menu. You can use the pull-down menu to switch filters.

Submitting an Activity Plan

To submit an activity plan for execution, you select it from a list of template plans. You can submit an activity plan by performing the following steps.

1. Select **Plans >>Submit** from the menu of the Activity Plan Monitor window.

The Select plan for submission dialog box is displayed.

The dialog box displays a list of plans saved as templates in the activity plan database.

2. Select a plan in the dialog box and click **OK**. The Plan Submission Parameters notebook is displayed. This notebook is the same as the Plan Properties notebook used during the preparation of an activity plan using the Activity Plan Editor. For more information about the pages of this notebook see the following sections:
 - General. See “Defining the General Properties of the Activity Plan” on page 37.
 - Execution Time. See “Scheduling Plans to Execute within a Time Interval” on page 38.
 - Frequency. See “Scheduling Activity Plans to Run Periodically” on page 40
 - TARGET_LIST specification. See “Defining the Variable Values” on page 50.

The **General** page is displayed by default. Before submitting the plan for execution you can edit the attributes that were defined at the time the plan was created. The modifications are applied only to the current submission instance and have no effect on the template. Using the notebook, you can modify the following plan attributes:

- Plan name

Note: If you do not specify a plan name in the **Name** text box, the name of the plan is generated at submission time in the format:

name@submission_date submission_time

- Priority level
- Execution time
- Recursion information
- Expiration information
- TARGET_LIST specification

If any of the subscribers specified in the activity plan are not valid, the operation fails on those subscribers, and continues on the other subscribers. Information about the subscribers on which the operation did not complete is written to the apmlog0 file in the Activity Planner trace files directory.

You can also submit an activity plan from the CLI. For information about this, see “wsubpln” on page 122.

Notes:

- a. When the Activity Planner executes an activity, the role required to run the operation defined in the activity is compared with the information of the administrator who initially submitted the plan containing the executing activity. This ensures that the administrator who initially submitted the plan still exists and that administrator authorization has not been lowered. If the administrator who initially submitted the plan no longer exists, or no longer has the authorization necessary to run the operation requested in the activity, the execution fails when it is run.
- b. If, after submitting an activity plan for execution, an entry of the submitted plan does not appear in the upper pane of the Activity Plan Monitor window, an error has occurred while the Activity Planner engine was preparing the plan for submission. For example, an error might occur during an attempt to resolve target information specified in the plan. Check the log file to determine the problem.

Defining the Variable Values

In the Plan Submission Parameters notebook you can define new variables, change values previously assigned to variables, and specify the targets for the plan to be submitted. To set the targets using variables, see “Using Variables to Specify Targets” on page 32.

To define the targets for the plan, perform the following steps:

1. From the Activity Plan Monitor window, select and submit a plan as explained in “Submitting an Activity Plan” on page 49.
2. From the Plan Properties notebook, select the **Variables** tab.
3. Select either item **by list** or **by file**, as follows:
 - by list** Select this item to activate the **Target types** pull-down menu. Enter a list of targets separated by commas or click the browse (...) button to navigate to the file.
 - by file** Select this item to activate the **Managed node** and **Path to file** text boxes. Type the name of the managed node where the file is stored, in the **Managed node** text box. Type the path of the file on the managed node that contains the list of targets names, in the **Path to file** text box.

If you used a variable when defining targets for the plan, the same variable must be specified in the **Variables** tab in the **Plan Submission Parameters** notebook. For more information, see “Using Variables to Specify Targets” on page 32.

Click **OK** to save the changes and close the Plan Submission Parameters notebook.

The plan is submitted for execution. An entry of the submitted plan is added to the upper pane of the Activity Plan Monitor window from where the status of the plan and its activities can be monitored.

Updating a Submitted Plan

After submitting an activity plan in the Activity Plan Monitor, you can update some of the fields in the plan, provided it has not yet started. You can update the plan by performing the following steps.

1. Select the plan you wish to update in the Activity Plan Monitor window.
2. Right-click the plan to display the **Selected** menu, or choose **Selected > Update**.

The Update Parameters notebook is displayed. The **General** page is displayed by default.

For more information about how to complete the pages of this notebook see the following sections:

- General. See “Defining the General Properties of the Activity Plan” on page 37.
- Execution Time. See “Scheduling Plans to Execute within a Time Interval” on page 38.
- Frequency. See “Scheduling Activity Plans to Run Periodically” on page 40
- TARGET_LIST specification. See “Defining the Variable Values.”

Before the plan is started, you can edit the attributes that were defined at the time the plan was created. Using the notebook, you can modify the following plan attributes:

- Name
- Priority

- Mail-ID
- Post notice
- Execution time
- Recursion information
- Expiration information
- Targets resolution
- TARGET_LIST

Click **OK** to save the changes and close the Update Parameters notebook. The changes are saved in the activity plan database.

Note: You can also delete an activity plan from the CLI. For information about this, see “wupdpn” on page 127.

Stopping the Recursion of a Submitted Activity Plan

After submitting an activity plan, you can stop the recursion mechanism defined when the plan was created.

To stop the recursion of a submitted activity plan, perform the following steps:

1. Select an activity plan in the Activity Plan Monitor main window.
2. Select **Stop recursion** in the **Selected** menu, or press the **Stop recursion** button in the task bar. A confirmation message is displayed.

Activity Planner completes any instances of the plan currently executing, then interrupts the recursion mechanism.

Note: You can also stop the recursion of a submitted activity plan from the CLI. For information about this, see “wcntpln” on page 98.

Controlling the Execution of Activity Plans and Activities

You can use the Activity Plan Monitor to pause, resume, cancel, and restart activity plans, activities, and targets for plug-ins that support this option, by performing the following steps:

1. From the Activity Plan Monitor window, select an activity plan, a target, or an activity.
2. Select either the **Pause**, **Resume**, **Restart**, **Cancel** or **Cancel Force** option from the **Selected** menu.

You can also cancel, pause, and resume the activity plan or the activity on a set of targets only, by selecting the targets on which you wish to perform the operation. Depending on the current state of the activity, only some options are selectable.

Pause Pauses the execution of an activity plan, or activity. Activities not yet executed are in paused state. For activities executed but not yet complete, the pause is managed by the application to which the activity belongs. For activities of the type Task, for example, the pause, resume, and cancel options are not activated. If you are distributing to devices, the pause operation might fail if the distribution has already reached the resource gateway. At that point, use the **wwebgw** command. See the *Reference Manual for Software Distribution* for details.

Resume

Resumes the execution of all activities in the activity plan that are in the paused state.

Submitting and Monitoring Activity Plans

Cancel

Cancels the selected activity plan or activity even if it is running and has not completed. The activity is canceled by the application to which it belongs. For example, for an activity related to a Software Distribution operation, the activity is canceled by the MDist 2 service. For activities that have not yet been performed, the Activity Planner engine performs the cancel operation. If you are distributing to devices, the cancel operation might fail if the distribution has already reached the resource gateway. At that point, use the **wwebgw** command. See the *Reference Manual for Software Distribution* for details.

Cancel Force

Changes the operation state of an activity or activity plan to **canceled**, even though submitted operations are not canceled. This setting has effect only on the status of the activity or activity plan in Activity Planner, but has no effect on the operations associated to the activity or activity plan, and no activity is performed at MDist 2 or Tivoli Web Gateway level. Use this option only for recovery purposes. When using this command to cancel distributions addressed to devices, you can use the **wwebgw -c** command to cancel the distribution. For more information on this command, refer to *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*. The **Cancel Force** option deploys the same functionality as the **wcntpln -f** command. For more information on this command, see “wcntpln” on page 98.

Restart

Restarts the execution of activities in an activity plan that failed on targets of a previous execution. The activities are restarted only on the targets that failed. If you restart an activity plan, only the failed activities are restarted.

Note: You can also control the execution of an activity plan or of activities from the CLI. For information about this, see “wcntpln” on page 98

Generating a Recovery Plan

If a plan fails, you can restart the whole plan, or you can generate a recovery plan template, containing only the failed activities for all the failed nodes.

Any conditions present in the original plan are not maintained in the recovery plan.

You can generate a recovery plan by performing the following steps.

Note: You can also generate a recovery plan template from the CLI. For information about this, see “wgenpln” on page 103.

1. From the Activity Plan Monitor window, select the failed plan you wish to restart.
2. Right-click the plan to display the **Selected** menu, or choose **Selected » Generate recovery plan**.

The Generate recovery plan notebook is displayed. The **General** page is displayed by default.

If you select the **Submit immediately after generation** check box, the template plan is submitted immediately after it is generated.

If you select the **Overwrite original plan** check box, the recovery plan template overwrites the original plan in the activity plan database, provided you remove the suffix appended automatically at the plan name in the **Name** text box.

Click **OK** to save the changes and close the Generate recovery plan notebook. The changes are saved in the activity plan database.

Launching the Distribution Status Console

The Distribution Status console enables you to view the status of distributions, such as which targets receive a distribution and which ones experience errors. You also can pause, resume, cancel, or delete a distribution if, for example, you encounter unavailable targets, failed installations, or network outages.

You can launch the Distribution Status console from the Activity Plan Monitor window to monitor and control Software Distribution operations defined in an activity. To launch the Distribution Status console, select **View »Activity Progress** from the menu.

Note: If you perform operations (pause, cancel, resume) from the Distribution Status console, or by using the **wmdist** command on Software Distribution operations defined in an activity, the Activity Planner is not able to detect that an operation has occurred. The status displayed in the Activity Plan Monitor window is then not consistent with the distribution status. For example, if you cancel an operation using **wmdist -c**, the distribution status is canceled, but the status reported in the Activity Plan Monitor is failed.

Refer to the *Tivoli Management Framework: User's Guide* for more information about the Distribution Status console.

Updating the Status of an Activity Plan

You can set the Activity Plan Monitor to automatically update and display the current status of all plans and activities, or set it to update upon request. The data displayed in the window is reloaded with the current date in the activity plan database. For example, to automatically update the Activity Plan Monitor window every 2 minutes and 40 seconds, perform the following steps:

1. From the **Settings** menu, select **Automatic Update**. The Automatic Update dialog box is displayed.
2. Select the **Enable Automatic Update** check box.
3. Drag the slider horizontally to the 2 minutes and 40 seconds position on the bar.
4. Click **OK** to save the information and close the dialog box.

To update the Activity Plan Monitor at any time, select **Refresh** from the **View** menu.

Deleting Completed Activity Plans

You can delete the status of a plan, or a specific instance of a recursive plan that has completed, by performing the following steps.

Note: You can also delete the status of an activity plan from the CLI. For information about this, see “**wdelstat**” on page 101.

1. Select a submitted plan in the Activity Plan Monitor window.
2. Choose **Selected » Delete**.

Cleaning Up the Database

To eliminate plans from the activity plan database that have been submitted, you can use the Activity Plan Monitor to schedule a cleanup operation. You can use the force option to eliminate plans in states other than completed states. Cleanup operations can be scheduled to occur at any of the following times:

- Immediately
- One time only on a specific day and at a specific time
- Particular days of the week
- Particular days of the month
- A date interval
- A time interval

To define the plans you want to remove, you can specify any of the following options:

- Plans with a particular status (canceled, failed, successful)
- Days elapsed since plans completed
- Days elapsed since plans started
- Plans with a specific age

When all the clean up parameters have been set, the Activity Plan Monitor relies on the Tivoli Management Framework Scheduler to perform the cleanup.

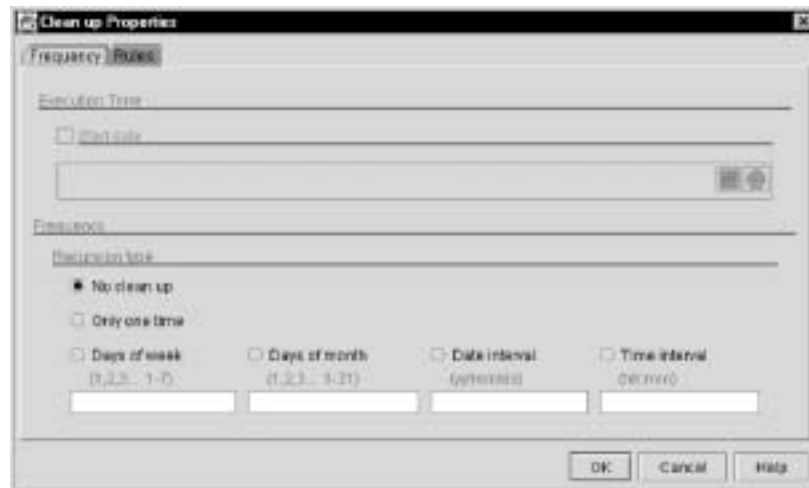
The following list gives the dates and times of an example cleanup of all plans in the activity plan database that have completed successfully:

- Days completed: at least 30
- Cleanup frequency: every 15 days
- Start date: March 20, 2003
- Start time: 8 a.m.

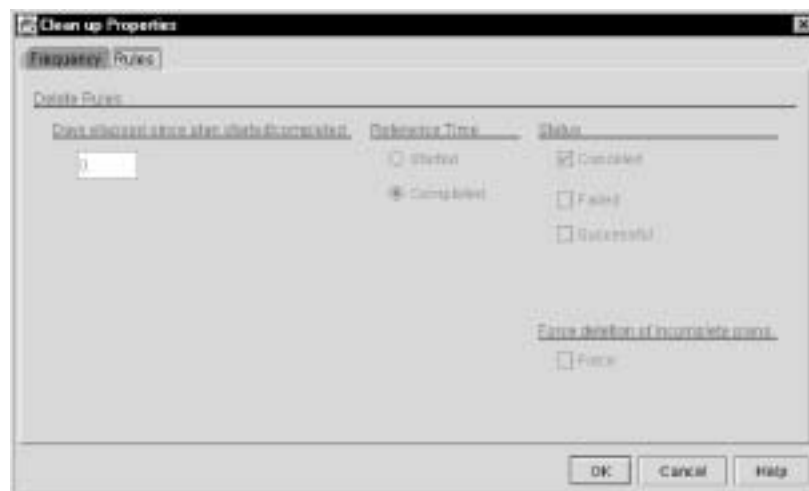
You can perform the cleanup as described in the following steps.

Note: You can also set filters for an activity plan from the CLI. For information about this, see “wsfdpln” on page 117.

1. From the Activity Plan Monitor main window, select **Settings** » **Clean Up** to display the Clean up Properties dialog box. The **Frequency** page is displayed by default.



2. In the **Recursion type** group, select the **Date interval** radio button and, in the text box below, type **00/00/15**. This specifies that the cleanup is performed every 15 days.
3. In the **Execution Time** group, select the **Start date** check box.
4. Click the Calendar icon and set the start date in the calendar displayed.
5. Select the **Rules** tab to display the **Rules** page.



6. In the **Days elapsed since plan started/completed** text box, type **30**.
7. Under **Reference Time**, select the **Completed** radio button.
8. Under **Status**, clear the **Canceled** check box and select the **Successful** check box.
9. Click **OK** to save the settings and close the dialog box.

Submitting and Monitoring Activity Plans

Chapter 3. Using the Command Line

This chapter describes how to use Activity Planner through the command line interface (CLI). The CLI interface is a textual interface that you use to manage an activity plan in the *activity plan definition file* format and manage the activities defined in the activity plan. An activity plan definition file is a file in Extensible Markup Language (XML) format. The XML file is made up of components called elements. Tags are used to describe elements. A start tag marks the beginning of an element and an end tag marks the end of the element. Elements can contain other elements. The element that contains all of the other elements is known as the root element. The elements that are contained in the root element are called subelements and they also may contain other subelements.

The activity plan definition file makes use of this structure to define activity plans. The root element in this file begins with the <activity_plan> tag and ends with the </activity_plan> tag. The information nested between these tags defines what type of element it is. This information is called the element-type name. The elements and subelements specified in the activity plan definition file define information such as:

- Activities to be performed
- Time of execution of the plan
- Plan recursion information
- Target systems of a plan or activities

Activity Plan Definition File

The following represents, in XML format, all the elements and subelements contained in the activity plan definition file. This is only a skeleton of the file. The element-type name of each element is not defined.

```
<?xml version="1.0">
<!DOCTYPE activity_plan >
<activity_plan>
  <name> </name>
  <description> </description>
    <priority> </priority>
    <submit_paused> </submit_paused>
    <start_not_before> </start_not_before>
    <compl_not_after> </compl_not_after>
    <cancel_at_cutoff> </cancel_at_cutoff>
    <stop_on_error> </stop_on_error>
    <mail_id> </mail_id>
    <post_notice> </post_notice>
    <notify_date> </notify_date>
    <final_activity> </final_activity>
    <target_resource_type> </target_resource_type>
    <targets type = "type"> </targets>1
    <excl_targets> </excl_targets>
    <targets_computation> </targets_computation>2
    <frequency_information>3
```

1. For possible target types, see "Defining Targets" on page 68.

2. This keyword is available only in the activity_plan section or in the activity section, but not in both sections at the same time.

3. For a list of all subelements of the <frequency_information> element, see "Setting Frequency Information" on page 62.

Activity Plan Definition File

```

    <date_interval> </date_interval>
    <stop_on_overlap> </stop_on_overlap>
    <stop_rec_on_error_level> </stop_rec_on_error_level>
    <expiration_date> </expiration_date>
    <targets_resolution> </targets_resolution>
</frequency_information>
<variable>
    <name> </name>
    <value> </value>
</variable>
<activity>
    <application> </application>
    <name> </name>
    <description> </description>
    <target_resource_type> </target_resource_type>
    <targets type = "type"> </targets>4
    <excl_targets> </excl_targets>
    <targets_computation> </targets_computation>5
    <execution_window>
        <start_at> </start_at>
        <suspend_at> </suspend_at>
    </execution_window>
    <operation> </operation>
    <condition> </condition>
    <FailOnPostScript> </FailOnPostScript>
    <parameter>
        <name> </name>
        <value> </value>
    </parameter>
</activity>
</activity_plan>

```

Table 5 details the subelements contained in the root element <activity_plan> and </activity_plan>.

Table 5. Subelements that define the activity plan

Subelement	Values	Required	Default
name	A name used to identify activity plans saved as drafts or templates in the activity plan database.		
	The maximum length is 128 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	String	Yes	None
description	A description of the activity plan.		
	The maximum length is 251 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	String	No	None
priority	Activities are executed based on the execution window and priority specified. The activities of all plans in submission that are ready to be executed, are queued and performed in order of priority.		
	high, normal, low	No	normal

4. For possible target types, see “Defining Targets” on page 68.

5. This keyword is available only in the activity_plan section or in the activity section, but not in both sections.

Table 5. Subelements that define the activity plan (continued)

Subelement	Values	Required	Default
submit_paused	The activity plan is submitted for execution and all actions necessary for execution of the plan are prepared and paused until a resume operation is explicitly requested.		
	y: yes n: no	No	n
start_not_before	The activity plan is not to be executed before the time specified in this subelement. If no value is specified, the plan starts immediately. The time reference refers to the time on the local machine where the plan is created.		
	String	No	Submission time in the format "yyyy-mm-dd hh:mm".
compl_not_after	<p>The date and time, within which the plan must complete. The format is "yyyy-mm-dd hh:mm". You can also set a relative date and time within which the plan must complete, based on the date and time the plan starts. The format is hh:mm. If the execution of the plan does not complete within this time, activities not yet executed are cancelled.</p> <p>The behavior of activities that have already been sent to the application depends on the cancel_at_cutoff setting. If it is set to y, the activities are cancelled. If the compl_not_after time is not specified, the plan continues indefinitely.</p> <p>The time reference refers to the time on the local machine where the plan is created.</p> <p>If the plan contains activities which cannot be cancelled, that is Tivoli Framework tasks or Software Distribution operations addressed to devices or users, a warning message is displayed. If you choose to submit the plan and the compl_not_after time expires, all activities are cancelled with the exception of Tivoli Framework tasks and Software Distribution operations addressed to devices or users.</p>		
	String	No	None
cancel_at_cutoff	<p>If set to n, the time specified by the compl_not_after attribute is ignored for those activities already running. If set to y, activities running when the compl_not_after time is reached are cancelled. If the plan contains activities which cannot be cancelled, that is Tivoli Framework tasks or Software Distribution operations addressed to devices or users, a warning message is displayed. If you choose to submit the plan, and the compl_not_after time expires, all activities are cancelled with the exception of Tivoli Framework tasks and Software Distribution operations addressed to devices or users.</p>		
	y: yes n: no	No	n

Table 5. Subelements that define the activity plan (continued)

Subelement	Values	Required	Default
stop_on_error	If an activity completes with the error level specified, the execution of the plan is stopped. If this attribute is not specified, all activities in the plan are executed, even if one or more complete with error.		
	warning fatal error	No	None
mail_id	The user e-mail address to notify when an operation is performed on the plan, such as, when the plan is submitted, when the plan has completed, or when a pause, resume, cancel, or restart operation is performed on the plan. More than one address can be specified, each separated by a comma.		
	The maximum length is 251 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	String	No	None
post_notice	Specifies whether to send a notice to the Activity Planner Notice group when the plan is submitted, when the plan has completed, or when a pause, resume, cancel, or restart operation is performed on the plan.		
	y : yes n : no	No	n
notify_date	Specifies whether to have Activity Planner send the administrator a message on the status of the plan. If you select this feature, a notification message is sent to the e-mail address specified in the mail_id subelement, or logged in the Activity Planner Notice group at the specified date and time. If a relative date and time are specified, the value is calculated starting from the start of the plan.		
	String	No	Current time in the format "yyyy-mm-dd hh:mm".
final_activity	The name of an activity to be executed after the completion of all activities in the activity plan. The final_activity is executed even if a stop_on_error occurs in the plan.		
	activity_name	No	None
target_resource_type	Specifies the kind of targets the activity plan is executed on. You can define one of the mutually exclusive targets by entering them between the <target-resource_type> </target-resource_type> tags.		
	Unknown Endpoint User Device ManagedNode	No	Unknown

Table 5. Subelements that define the activity plan (continued)

Subelement	Values	Required	Default
targets	Targets can be defined either at the activity plan level, and therefore apply to all contained activities, or they can be defined for each activity within the <activity> </activity> tags.		
	For more information, see “Defining Targets” on page 68	When not specified at activity level.	None
excl_targets	Targets can be excluded either at the activity plan level, and therefore apply to all contained activities, or they can be excluded for each activity within the <activity> </activity> tags.		
	For more information, see “Defining Targets” on page 68	No.	None
targets_computation	Targets can be resolved either at the plan submission time, or at the activity execution time. If targets are defined at plan level and targets resolution is specified at plan submission, targets are calculated when the plan is submitted and inserted in the ACT_STATUS_TGT table. In case the target exists, the OID in the table is the current OID, otherwise it is ----.		
	p : at plan submission a : at activity execution	No.	p
frequency_information	Defines the rules for the regular execution of an activity plan.		
	For more information, see “Setting Frequency Information” on page 62.	No	No recursion.
activity	Each activity subelement defines an operation that is part of the activity plan.		
	For more information, see “Defining Activities in the Activity Plan Definition File” on page 63.	Yes	None
variable	Variables can be defined to identify targets and software packages in an activity plan.		
	For more information, see “Defining Variables with the Variable Subelement” on page 63.	No	None

Setting Frequency Information

You specify frequency information to schedule repetitive submissions of the activity plan. You can specify the subelements shown in Table 6.

Table 6. Subelements that define frequency information

Subelements	Values	Required	Default
days_of_week	Executes the activity plan on specific days of the week represented by a number from 1 through 7 separated by a comma or hyphen. 1 represents Sunday. For example, <code><days_of_week> 2-4,6 </days_of_week></code> , executes the plan on Monday, Tuesday, Wednesday, and Saturday.		
	String	No	None
days_of_month	Executes the plan on specific days of the month, represented by a number from 1 through 31 separated by a comma or hyphen. For example, <code><days_of_month> 1-5 </days_of_month></code> , executes the plan on the 1st, 2nd, 3rd, 4th, and 5th day of the month. If the specification is greater than the number of days in the month, the plan is executed on the last day of the month.		
	String	No	None
date_interval	Specifies how often the execution of the plan should be repeated. The format is <i>yy/mm/dd</i> . For example, <code><date_interval> 00/02/01 </date_interval></code> , executes the plan every two months and one day. <code><date_interval> 01/00/00 </date_interval></code> , executes the plan once a year. The Activity Planner uses the time defined in the <code><start_not_before></code> <code></start_not_before></code> tags as a point of reference. If these tags are not specified, the Activity Planner uses the submission time as a point of reference.		
	String	No	None
time_interval	Specifies how often the execution of the plan should be repeated. The format is <i>hh:mm</i> . For example, <code><time_interval>01:00 </time_interval></code> repeats the execution of the plan every hour.		
	String	No	None
stop_on_overlap	If a new instance of a repetitive plan is generated before the previous instance has completed and this subelement is set to y , the scheduled repeat of the plan is stopped, but the instance currently in execution completes. If set to n , the new instance is skipped until the current instance completes. The new instance is executed at the next scheduled repeat.		
	y : yes n : no	No	n
stop_rec_on_error_level	If an instance of the plan is completed with an error at the specified level or higher, the plan recursion mechanism is stopped.		
	warning error fatal	No	None

Table 6. Subelements that define frequency information (continued)

Subelements	Values	Required	Default
expiration_date	This date specifies when the scheduled repeat of the plan must be stopped. The format is <i>yyyy/mm/dd:hh:mm</i> . You can also set a relative date within which the recursion mechanism must complete, based on the date the plan or the first recursion starts. The format is <i>hh:mm</i> .		
	String	No	None
targets_resolution	The target list is recalculated when a new instance of the plan is submitted (dynamic), or the list already calculated when the first instance was submitted is used (static).		
	If the targets resolution (targets_computation=a) is at activity execution, the targets are evaluated when the activity starts.		
	static, dynamic	No	static

Defining Variables with the Variable Subelement

You can define variables directly in the activity plan definition file using the variable subelement. For more information on variables, see “Using Variables” on page 25. The variable subelement contains the name and value subelements. When you have defined these, you can use the variables when specifying the plug-in parameters, for example, software packages or the targets of the activities contained in the activity plan. For example, you can define a list of gateways, and the targets assigned to those gateways, in the variable subelement as follows:

```
<variable>
  <name> gateway_list </name>
  <value> gw1, gw2, gw3 </value>
</variable>
<variable>
  <name> gw1_targets </name>
  <value> targ11, targ12, targ13, targ14, targ15 </value>
</variable>
<variable>
  <name> gw2_targets </name>
  <value> targ21, targ22 </value>
</variable>
<variable>
  <name> gw3_targets </name>
  <value> targ31 </value>
</variable>
```

The maximum length for a variable name is 128 alphanumeric single-byte characters. The maximum length for a variable value is 2000 single-byte characters. Double-byte characters count as two single-byte characters.

Defining Activities in the Activity Plan Definition File

You define activities in the activity subelement. You can specify the subelements shown in Table 7 on page 64.

Activity Plan Definition File

Table 7. Subelements that define activities in the activity plan definition file

Subelements	Values	Required	Default
application	The name of the application to be called on to manage the activity.		
	SoftwareDistribution TaskLibrary Inventory	Yes	None
name	Identifies the activity in the activity plan. Each activity must have a unique name. The maximum length is 128 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	String	Yes	None
description	A description of the activity. The maximum length is 251 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	String	No	None
target_resource_type	Specifies the kind of targets the activity plan is executed on. You can define one of the following mutually exclusive targets by entering them between the <target-resource_type> </target-resource_type> tags.		
	Unknown Endpoint User Device ManagedNode	No	Unknown
targets	The syntax for targets is the same as described in Table 5 on page 58. If targets are specified at the activity plan level, they cannot be specified at the activity level.		
	For more information, see “Defining Targets” on page 68	Only if not specified at Activity Plan level.	None
excl_targets	Targets can be excluded either at the activity plan level, and therefore apply to all contained activities, or they can be excluded for each activity within the <activity> </activity> tags.		
	For more information, see “Defining Targets” on page 68	No.	None
targets_computation	Targets can be resolved either at the plan submission time, or at the activity execution time.		
	p : at plan submission a : at activity execution	No.	p
execution_window	This subelement defines a time window during which the activity is to be performed. The availability of this option depends on the installed plug-ins.		
	For more information, see “The Execution Window” on page 65	No	None

Table 7. Subelements that define activities in the activity plan definition file (continued)

Subelements	Values	Required	Default
operation	Activities can perform Tivoli Management Framework Task Library tasks, automatically available when you install Activity Planner. If you install Software Distribution and Inventory and register the plug-ins, Software Distribution and Inventory operations are also available.		
	For more information, see “Supported Task Library Activities” on page 69 and <i>User’s Guide for Software Distribution</i> and <i>User’s Guide for Inventory</i> .	Yes	None
condition	By defining this subelement, an activity can be made conditional on the outcome of another activity. The maximum length is 2000 alphanumeric single-byte characters. Double-byte characters count as two single-byte characters.		
	For more information, see “Using Conditions” on page 66.	No	None
FailOnPostScript	Use this option to specify that the whole data moving operation must be considered as failed if the post-script on the source host fails. This option is available only for send and retrieve operations.		
	<ul style="list-style-type: none"> • T (true) • F (false) 	No	F
parameters	Task Library parameters, automatically available when you install Activity Planner. Software Distribution and Inventory parameters available. If you install Software Distribution and Inventory and register the plug-ins, Software Distribution and Inventory operations are also available.		
	For more information, see “Supported Task Library Activities” on page 69 and <i>User’s Guide for Software Distribution</i> and <i>User’s Guide for Inventory</i> .	Yes	None

The Execution Window

You can manage the time frame within which activities are executed and completed by specifying information in the `execution_window` subelement. The `execution_window` is a subelement of the activity subelement. You can define the time frame within the execution window for some operations, based on the plug-in you installed. The time you specify in the `execution_window` subelement refers to the local time on the workstation where the activity plan is created. At the end of the time period specified in this subelement, any activities that have not completed are suspended. Suspended activities are resumed in the next execution window.

The Table 8 on page 66 describes the subelements specified in the `execution_window` subelement:

Table 8. Subelements that define the execution window

Subelements	Values	Required	Default
start_at	Specifies the day of the week and the time the execution of the activity starts. The format is <i>d, hh:mm</i> , where <i>d</i> is a number from 1 through 7. 1 represents Sunday.		
	String	Yes	None
suspend_at	Specifies the day of the week and the time after which an activity cannot execute. The format is <i>d, hh:mm</i> , where <i>d</i> is a number from 1 through 7. 1 represents Sunday. An activity still in process at this time is paused and the execution is resumed at the next specified start_at time.		
	String	Yes	None

Note: The duration of an execution window must be more than five minutes. If the time to the start of the next execution window is not more than five minutes, the two execution windows are merged, so that the window starts at the start_at time of the first window and ends at the suspend_at time of the second window.

Using Conditions

The execution of an activity can be conditioned by the outcome of the execution of one or more conditioning activities in the same activity plan. A condition has the following syntax:

Condition = (condition)|condition bool_operator condition|sub-condition

where,

Sub-condition = type_of_condition (activity_name [percentage])

Bool_operator = AND|OR

Type_of_condition = ResultFor_Targets

Percentage = number %

Result = C|S|F

For_Targets = A|T|D

The *type_of_condition* parameter is composed of two different parts: **Result** and **For_Targets**.

Result conditions the execution of the conditioned activity depending on the outcome of the execution of the conditioning activity.

Note: When expressing conditions in the XML file, the conditioned activity must follow the conditioning activity sequentially, otherwise a looping situation may occur and no activities are executed.

For example, assume an activity plan includes Activity A, Activity B, and Activity C. The execution of Activity B is conditioned by the successful completion of Activity A, and the execution of Activity C is conditioned by the successful execution of Activity B. Activity A cannot be conditioned by Activity C.

The execution of the conditioning activity can have any one of the following results:

C (Completion)

The conditioned activity is performed when the conditioning activity completes, regardless of whether it completes successfully or with error.

S (Success)

The conditioned activity is performed only if the conditioning activity completes successfully with no error.

F (Failure)

The conditioned activity is performed if the conditioning activity fails with an error.

The second part of the *type_of_condition* parameter is used to determine the targets of the conditioned activity. Use **For_Targets** to indicate the following:

A (All)

The conditioned activity is performed when the conditioning activity has completed execution on all its targets. A completion percentage of targets can be specified. For example, you can specify to execute the conditioned activity when the conditioning activity A has completed executing on 50% of its targets.

T (Target)

The conditioned activity depends on the execution of the conditioning activity on target X. The operation specified in the conditioned activity is executed on target X when the conditioning activity has completed execution on target X. Conditioning by target is supported for all activities whose plug-in supports the multi-start function, that is all Software Distribution operations with the exception of the load, unload and send from endpoint to endpoint operations. For more information on Software Distribution operations, refer to the *User's Guide for Software Distribution*.

D (Depot)

The conditioned activity on a set of targets sharing the same depot is performed when the conditioning activity has completed execution on the specified depot.

When specifying a condition using a percentage, the conditioned activity might not start when that percentage is reached. This might occur where a single report for the completion of an operation groups many targets together. Even if the number of targets required to satisfy the percentage has been reached, if the report is not sent to the Activity Planner engine promptly, the conditioned activity does not start.

For Software Distribution activities, you can modify the MDist 2 **notify_interval** parameter of the gateway to return the reports more frequently. Refer to the *Tivoli Management Framework: Reference Manual* for more information about modifying MDist 2 parameters using the **wmdist** command. Note that setting the **notify_interval** parameter to a small interval might decrease product performance.

In addition to the **Result** and **For_Targets** parameters, you can specify the AND and OR Boolean operators to add further flexibility to the condition, as described below:

AND If one of the conditions is not met, the activity is set to **Canceled by condition**.

OR If at least one of the conditions is met, the activity is successful.

If some targets are included in the target list of the conditioned activity but are not included in the target list of the conditioning activity, by default they are started when the conditioned activity starts. To change this behavior, modify the

cancel_unique_targets parameter in the apm.ini file. For more information on this parameter, see “Defining the Activity Planner Engine Parameters” on page 9.

Defining Targets

You must define targets at either the activity plan level or at the activity level in the activity plan definition file. You can choose the kind of targets to address by entering **endpoint**, **user**, **device**, or **managed node** between the `<target_resource_type>` `</target_resource_type>` tags. If targets are specified at the activity plan level, then the operations defined in the contained activities are executed on these targets. Targets cannot be specified at the activity level if they have been specified at the activity plan level. You specify targets at the activity level by defining the target subelement within the activity subelement. You specify targets at the activity plan level by defining the target subelement in the `activity_plan` root element, but outside the activity element.

Note: In a Software Distribution environment, targets of a load or unload operation must be addressed using the managed node label. If you address a target using a gateway label, the operation fails.

You can specify targets in any of the following ways:

Target names

You can define a list of target names as the subelement-type of the target subelement with separating commas. The following is the syntax of the target subelement using target names:

```
<targets type = "list"> trg_name1, trg_name2, ...trg_namen </targets>
```

File name

You can specify targets by indicating the path to a file containing a list of target names with separating commas, or with target names entered on separated lines. The following is the syntax of the target subelement using a file name.

```
<targets type = "file"> @managed_node:c:\astral.txt </targets>
```

Inventory subscriber

You can specify queries that access the configuration repository and select a list of targets as the query result. The following is the syntax of the target subelement using an Inventory subscriber:

```
<targets type = "query"> SQL_query </targets>
```

Query library subscriber

You can specify a query representing a query library to return a list of possible subscribers from the configuration repository. The following is the syntax of the target subelement using a query library from the Tivoli object database to retrieve subscribers:

```
<targets type = "query_library"> query2 </targets>
```

Profile subscriber

You can specify profile subscribers as targets of an activity plan. Profile subscribers comprise profile managers, and resource groups of devices and users. The activities contained in the plan are performed on the subscribers associated with the specified profile subscriber. It is possible to specify profile subscribers with the same name; to avoid ambiguities, it is recommended you use the following syntax:

```
<targets type = "profile"> ResourceGroup:TEST </targets>
<targets type = "profile"> ProfileManager:TEST </targets>
```


The following is the syntax of the target subelement using a profile manager name:

```
<targets type = "profile"> PM1, PM2 </targets>
```

Directory query library subscriber

You can specify a query representing a directory query library to return a list of possible subscribers from the Lightweight Directory Access Protocol (LDAP) database. The following is the syntax of the target subelement using a directory query library from the Tivoli object database to retrieve subscribers:

```
<targets type = "enterprise_directory"> query1 </targets>
```

Pristine Target Subscriber

You can specify a list of machines registered in the Pristine Manager database. The following is the syntax of the target subelement using a Pristine Manager subscriber:

```
<targets type="pristine">pristinetxt1</targets>
```

Variables

You can specify targets using variables. The values of the variables are specified at the time the plan is created or submitted and the list of targets is automatically evaluated at execution time without user intervention. You can specify more than one variable, with separating commas. This feature is maintained for backward compatibility with Software Distribution 4.1.

Table 9 details the supported types of built-in variables:

Table 9. Supported built-in variables for specifying targets

Supported Built-in Variable	Description
\$(TARGET_LIST)	The targets are specified when the activity plan is created or submitted.
\$(DEPOT_LIST)	The targets are the managed nodes configured as depots to which the endpoints, specified in the \$(TARGET_LIST) variable, are assigned.
\$(ORIGINATOR)	The target is the hostname of the machine from which the activity plan is submitted.

The following is the syntax of the target subelement using built-in variables:

```
<targets type = "variable"> $(TARGET_LIST) </targets>
```

Supported Task Library Activities

If you set the operation attribute to task, you can define Tivoli Management Framework Task Library tasks in the activity stanza. Each task has its own set of permitted parameters. For more information about task library parameters refer to the **wruntask** command, in the *Tivoli Management Framework: Reference Manual*.

Parameters of Task Library Operations

Table 10 on page 70 details task library operations supported by the Activity Planner feature and the corresponding parameters.

Supported Task Library Activities

Table 10. Supported task library operations and parameters

Operations	Parameters
ExecuteTask	Argumentn ExecutionMode Library OutputFile OutputFormat OutputHost StageCount StageInterval Task Timeout

Parameter Values for Task Library Operations

Table 11 details the possible values and default for each parameter.

Table 11. Parameter values

Parameter	Values	Default
Argumentn	String	
ExecutionMode	EXEC_MODE_PARALLEL EXEC_MODE_SERIAL EXEC_MODE_STAGED	EXEC_MODE_PARALLEL
Library	String	
OutputFile	String	
OutputFormat	A single occurrence of one or more of the following characters: H (header) R (return value) O (standard output) E (standard error)	HROE
OutputHost	String (must be a managed node)	
StageCount	Integer	<256
StageInterval	Integer expressed in seconds	
Task	String	
Timeout	Integer expressed in seconds	300 seconds

Supported Software Distribution Activities

The activities defined in an activity plan can be any of the following Software Distribution operations:

- **Change management**
 - Install
 - Remove
 - Undo
 - Accept
 - Commit
- **Depot management**
 - Load
 - Unload

- **Data moving**
 - Send Data
 - Retrieve Data
 - Delete Data

Parameters of Software Distribution Operations

Each supported Software Distribution operation has its own set of supported parameters. The tables in this section detail operations supported by the Activity Planner feature and the corresponding parameters, as follows:

- The accept operation, see Table 12 on page 72.
- The install operation, see Table 13 on page 73.
- The remove operation, see Table 14 on page 74.
- The commit operation, see Table 15 on page 75.
- The undo operation, see Table 16 on page 76.
- The load operation, see Table 17 on page 77.
- The unload operation, see Table 18 on page 77.
- The send operation, see Table 19 on page 78.
- The retrieve operation, see Table 20 on page 79.
- The delete operation, see Table 21 on page 80.

Supported Software Distribution Activities

Table 12. Parameters for the accept operation

Accept operation parameters		
AllowDefer	MobileEscalateDate_6	MobileEscalateMessage_9
AllowReject	MobileEscalateDate_8	MobileForceMandatory
Deadline	MobileEscalateDate_7	MobileHidden
DefaultAction	MobileEscalateDate_9	MobileMandatoryDate
DefaultTimeout	MobileEscalateMessage_0	NotifyInterval
DisconnectedOperation	MobileEscalateMessage_1	NotifyIntervalUnit
EnableMulticast	MobileEscalateMessage_2	Priority
EnableNotification	MobileEscalateMessage_3	RelativeDeadline
ExecutionTimeout	MobileEscalateMessage_4	RetryUnicast
ExecutionTimeoutUnit	MobileEscalateMessage_5	RoamEp
Ignore	MobileEscalateMessage_6	SendTimeout
Label	MobileEscalateMessage_7	SendTimeoutUnit
LenientDistribution	MobileEscalateMessage_8	SoftwarePackage
MobileEscalateDate_0		UserNotification
MobileEscalateDate_1		WakeOnLan
MobileEscalateDate_2		
MobileEscalateDate_3		
MobileEscalateDate_4		
MobileEscalateDate_5		

Table 13. Parameters for the install operation

Install operation parameters		
AllowDefer	MobileEscalateDate_1	MobileEscalateMessage_9
AllowReject	MobileEscalateDate_3	MobileForceMandatory
AutoAccept	MobileEscalateDate_2	MobileHidden
AutoCommit	MobileEscalateDate_4	MobileMandatoryDate
BasePackage	MobileEscalateDate_5	NotifyInterval
Deadline	MobileEscalateDate_6	NotifyIntervalUnit
DefaultAction	MobileEscalateDate_7	Priority
DefaultTimeout	MobileEscalateDate_8	Reboot
DependencyCheck	MobileEscalateDate_9	RelativeDeadline
DisconnectedOperation	MobileEscalateMessage_0	RetryUnicast
Disposable	MobileEscalateMessage_1	RoamEp
DistributionMode	MobileEscalateMessage_2	SendTimeout
EnableMulticast	MobileEscalateMessage_3	SendTimeoutUnit
EnableNotification	MobileEscalateMessage_4	SoftwarePackage
ExecutionTimeout	MobileEscalateMessage_5	Transactional
ExecutionTimeoutUnit	MobileEscalateMessage_6	Undo
Force	MobileEscalateMessage_7	UserNotification
FromCD	MobileEscalateMessage_8	WakeOnLan
FromDepot		%variableName
FromFileServer		
Ignore		
Label		
LenientDistribution		
MobileEscalateDate_0		

Supported Software Distribution Activities

Table 14. Parameters for the remove operation

Remove operation parameters		
AllowDefer	MobileEscalateDate_3	MobileEscalateMessage_9
AllowReject	MobileEscalateDate_2	MobileForceMandatory
AutoAccept	MobileEscalateDate_4	MobileHidden
AutoCommit	MobileEscalateDate_5	MobileMandatoryDate
Deadline	MobileEscalateDate_6	NotifyInterval
DefaultAction	MobileEscalateDate_7	NotifyIntervalUnit
DefaultTimeout	MobileEscalateDate_8	Priority
DependencyCheck	MobileEscalateDate_9	Reboot
DisconnectedOperation	MobileEscalateMessage_0	RelativeDeadline
Disposable	MobileEscalateMessage_1	RetryUnicast
DistributionMode	MobileEscalateMessage_2	RoamEp
EnableMulticast	MobileEscalateMessage_3	SendTimeout
EnableNotification	MobileEscalateMessage_4	SendTimeoutUnit
ExecutionTimeout	MobileEscalateMessage_5	SoftwarePackage
ExecutionTimeoutUnit	MobileEscalateMessage_6	Transactional
Force	MobileEscalateMessage_7	Undo UserNotification
FromCD	MobileEscalateMessage_8	WakeOnLan
FromDepot		%variableName
FromFileServer		
Ignore		
Label		
LenientDistribution		
MobileEscalateDate_0		
MobileEscalateDate_1		

Table 15. Parameters for the commit operation

Commit operation parameters		
AllowDefer	MobileEscalateDate_8	MobileHidden
AllowReject	MobileEscalateDate_9	MobileMandatoryDate
Deadline	MobileEscalateMessage_0	NotifyInterval
DefaultAction	MobileEscalateMessage_1	NotifyIntervalUnit
DefaultTimeout	MobileEscalateMessage_2	Priority
DisconnectedOperation	MobileEscalateMessage_3	Reboot
EnableMulticast	MobileEscalateMessage_4	RelativeDeadline
EnableNotification	MobileEscalateMessage_5	RetryUnicast
ExecutionTimeoutUnit	MobileEscalateMessage_6	RoamEp
Ignore	MobileEscalateMessage_7	SendTimeout
Label	MobileEscalateMessage_8	SendTimeoutUnit
LenientDistribution	MobileEscalateMessage_9	SoftwarePackage
MobileEscalateDate_0	MobileForceMandatory	Transactional
MobileEscalateDate_1		UserNotification
MobileEscalateDate_2		WakeOnLan
MobileEscalateDate_3		
MobileEscalateDate_4		
MobileEscalateDate_5		
MobileEscalateDate_7		
MobileEscalateDate_6		

Supported Software Distribution Activities

Table 16. Parameters for the undo operation

Undo operation parameters		
AllowDefer	MobileEscalateDate_6	MobileForceMandatory
AllowReject	MobileEscalateDate_7	MobileHidden
AutoCommit	MobileEscalateDate_8	MobileMandatoryDate
Deadline	MobileEscalateDate_9	NotifyInterval
DefaultAction	MobileEscalateMessage_0	NotifyIntervalUnit
DefaultTimeout	MobileEscalateMessage_1	Priority
DependencyCheck	MobileEscalateMessage_2	Reboot
DisconnectedOperation	MobileEscalateMessage_3	RelativeDeadline
EnableMulticast	MobileEscalateMessage_4	RetryUnicast
EnableNotification	MobileEscalateMessage_5	RoamEp
ExecutionTimeout	MobileEscalateMessage_6	SendTimeout
ExecutionTimeoutUnit	MobileEscalateMessage_7	SendTimeoutUnit
Ignore	MobileEscalateMessage_8	SoftwarePackage
Label	MobileEscalateMessage_9	Transactional
LenientDistribution		UserNotification
MobileEscalateDate_0		WakeOnLan
MobileEscalateDate_1		
MobileEscalateDate_2		
MobileEscalateDate_3		
MobileEscalateDate_4		
MobileEscalateDate_5		

Table 17. Parameters for the load operation

Load operation parameters		
AllowDefer	Ignore	RelativeDeadline
AllowReject	Label	SendTimeout
BasePackage	NotifyInterval	SendTimeoutUnit
Deadline	NotifyIntervalUnit	SoftwarePackage
DefaultAction	Priority	RetryUnicast
DefaultTimeout		UserNotification
EnableMulticast		
EnableNotification		
ExecutionTimeout		
ExecutionTimeoutUnit		
Force		

Table 18. Parameters for the unload operation

Unload operation parameters		
AllowDefer	Ignore	RelativeDeadline
AllowReject	Label	SendTimeout
BasePackage	NotifyInterval	SendTimeoutUnit
Deadline	NotifyIntervalUnit	SoftwarePackage
DefaultAction	Priority	RetryUnicast
DefaultTimeout		UserNotification
EnableMulticast		
EnableNotification		
ExecutionTimeout		
ExecutionTimeoutUnit		
Force		

Supported Software Distribution Activities

Table 19. Parameters for the send operation

Send operation parameters		
AllowDefer	MobileEscalateDate_6	MobileMandatoryDate
AllowReject	MobileEscalateDate_7	NotifyInterval
CodepageTranslation	MobileEscalateDate_8	NotifyIntervalUnit
Deadline	MobileEscalateDate_9	OrigPath
DefaultAction	MobileEscalateMessage_0	OrigPostScript
DefaultTimeout	MobileEscalateMessage_1	OrigPreScript
DestPath	MobileEscalateMessage_2	Priority
DestPostScript	MobileEscalateMessage_3	Reboot
DestPreScript	MobileEscalateMessage_4	Recursive
DisconnectedOperation	MobileEscalateMessage_5	RelativeDeadline
EnableMulticast	MobileEscalateMessage_6	RetryUnicast
EnableNotification	MobileEscalateMessage_7	RoamEp
ExecutionTimeout	MobileEscalateMessage_8	SendTimeout
ExecutionTimeoutUnit	MobileEscalateMessage_9	SendTimeoutUnit
FileName	MobileForceMandatory	SourceHost
Ignore	MobileHidden	UserNotification
Label		WakeOnLan
MobileEscalateDate_0		<i>\$\$SoftwareDependency</i>
MobileEscalateDate_1		<i>%variableName</i>
MobileEscalateDate_2		
MobileEscalateDate_3		
MobileEscalateDate_4		
MobileEscalateDate_5		

Table 20. Parameters for the retrieve operation

Retrieve operation parameters		
AllowDefer	MobileEscalateDate_6	MobileMandatoryDate
AllowReject	MobileEscalateDate_7	NotifyInterval
CodepageTranslation	MobileEscalateDate_8	NotifyIntervalUnit
Deadline	MobileEscalateDate_9	OrigPath
DefaultAction	MobileEscalateMessage_0	OrigPostScript
DefaultTimeout	MobileEscalateMessage_1	OrigPreScript
DestPath	MobileEscalateMessage_2	Priority
DestPostScript	MobileEscalateMessage_3	Reboot
DestPreScript	MobileEscalateMessage_4	Recursive
DisconnectedOperation	MobileEscalateMessage_5	RelativeDeadline
EnableMulticast	MobileEscalateMessage_6	RetryUnicast
EnableNotification	MobileEscalateMessage_7	RoamEp
ExecutionTimeout	MobileEscalateMessage_8	SendTimeout
ExecutionTimeoutUnit	MobileEscalateMessage_9	SendTimeoutUnit
FileName	MobileForceMandatory	SourceHost
Label	MobileHidden	UserNotification
MobileEscalateDate_0		WakeOnLan
MobileEscalateDate_1		<i>\$SoftwareDependency</i>
MobileEscalateDate_2		<i>%variableName</i>
MobileEscalateDate_3		
MobileEscalateDate_4		
MobileEscalateDate_5		

Supported Software Distribution Activities

Table 21. Parameters for the delete operation

Delete operation parameters		
AllowDefer	MobileEscalateDate_5	MobileForceMandatory
AllowReject	MobileEscalateDate_6	MobileHidden
CodepageTranslation	MobileEscalateDate_7	MobileMandatoryDate
Deadline	MobileEscalateDate_8	NotifyInterval
DefaultAction	MobileEscalateDate_9	NotifyIntervalUnit
DefaultTimeout	MobileEscalateMessage_0	Priority
DestPath	MobileEscalateMessage_1	Reboot
DestPostScript	MobileEscalateMessage_2	Recursive
DestPreScript	MobileEscalateMessage_3	RelativeDeadline
DisconnectedOperation	MobileEscalateMessage_4	RetryUnicast
EnableMulticast	MobileEscalateMessage_5	RoamEp
EnableNotification	MobileEscalateMessage_6	SendTimeout
ExecutionTimeout	MobileEscalateMessage_7	SendTimeoutUnit
ExecutionTimeoutUnit	MobileEscalateMessage_8	SourceHost
FileName	MobileEscalateMessage_9	UserNotification
Label		WakeOnLan
MobileEscalateDate_0		<i>\$\$SoftwareDependency</i>
MobileEscalateDate_1		<i>%variableName</i>
MobileEscalateDate_2		
MobileEscalateDate_3		
MobileEscalateDate_4		

Parameter Values for Software Distribution Operations

Table 22 details the possible values and default for each parameter.

Table 22. Parameter values

Parameter	Values	Default
AllowDefer	T (true) F (false)	F
AllowReject	T (true) F (false)	T

Table 22. Parameter values (continued)

Parameter	Values	Default
AutoAccept	T (true) F (false)	F
AutoCommit	T (true) F (false)	F
CodePageTranslation	T (true) F (false)	F
Deadline	Date and time in the format: "yyyy-mm-dd hh:mm"	
DefaultAction	Accept Reject	Accept
DefaultTimeout	seconds	60 seconds
DependencyCheck	T (true) F (false)	T
DestPath	String	
DestPostScript	String	
DestPreScript	String	
DisconnectedOperation	T (true) F (false)	T
Disposable	T (true) F (false)	F
DistributionMode	CHNG_ALL CHNG_SRC CHNG_REPAIR	CHNG_ALL
EnableMulticast	T (true) F (false)	F
EnableTimeout	T (true) F (false)	F
ExecutionTimeout	Integer	If not specified, the default at the last gateway in the path will be used.
ExecutionTimeoutUnit	S (seconds) M (minutes) H (hours) D (days)	S
FileName	String	
Force	T (true) F (false)	F
FromCD	T (true) F (false)	F
FromDepot	T (true) F (false)	F
FromFileServer	T (true) F (false)	F
Ignore	T (true) F (false)	F

Supported Software Distribution Activities

Table 22. Parameter values (continued)

Parameter	Values	Default
Label		<i>SoftwarePackageName</i>
LenientDistribution	T (true) F (false)	T
MobileEscalateDate_0	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_1	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_2	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_3	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_4	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_5	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_6	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_7	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_8	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_9	"yyyy-mm-dd hh:mm"	
MobileEscalateMessage_0	String	
MobileEscalateMessage_1	String	
MobileEscalateMessage_2	String	
MobileEscalateMessage_3	String	
MobileEscalateMessage_4	String	
MobileEscalateMessage_5	String	
MobileEscalateMessage_6	String	
MobileEscalateMessage_7	String	
MobileEscalateMessage_8	String	
MobileEscalateMessage_9	String	
MobileForceMandatory	T (true) F (false)	T
MobileHidden	T (true) F (false)	F
MobileMandatoryDate	"yyyy-mm-dd hh:mm"	
NotifyInterval	Integer	If not specified, the default at the last gateway in the path will be used.
NotifyIntervalUnit	S (seconds) M (minutes) H (hours) D (days)	M
OrigPath	String	
OrigPostScript	String	
OrigPreScript	String	
Priority	PRTY_HIGH PRTY_MEDIUM PRTY_LOW	PRTY_MEDIUM

Table 22. Parameter values (continued)

Parameter	Values	Default
Reboot	REBOOT_YES REBOOT_NO REBOOT_IF_NEC REBOOT_AUTO	REBOOT_NO
Recursive	T (true) F (false)	F
RelativeDeadline	hh:mm	The hours and minutes specified are added to the start not before time to obtain the deadline. If a deadline is not specified, the default deadline of 72 hours is used.
RetryUnicast	T (true) F (false)	F
RoamEp	T (true) F (false)	T
SendTimeout	Integer	If not specified, the default at the last gateway in the path will be used.
SendTimeoutUnit	S (seconds) M (minutes) H (hours) D (days)	S
SoftwarePackage	String	
SourceHost	String	
Transactional	TRAN_NO TRAN_YES TRANS_IF_NEC	TRAN_NO
Undo	UNDO_NO UNDO_YES UNDO_IF_POS UNDO_TRANS	UNDO_NO
UserNotification	T (true) F (false)	F
WakeOnLan	T (true) F (false)	F
<i>\$SoftwareDependency</i>	A Software Dependency can be specified as a parameter. The name must be prefixed by the \$ character. The value must be one of the following: PREREQ EXREQ	
<i>%variableName</i>	All user defined variables can be specified as parameters. The name must be prefixed by the % character. The value must be of type string.	

Supported Software Distribution Activities

For example, to define an activity that installs the software package SP1^1.0 in transactional mode using the force option, the activity stanza is specified as follows:

```
<activity>
  <operation> install </operation>
  <parameter>
    <name> force </name>
    <value> y </value>
  </parameter>
  <parameter>
    <name> transactional </name>
    <value> y </value>
  </parameter>
  <parameter>
    <name> softwarepackage </name>
    <value> SP1^1.0 </value>
  </parameter>
</activity>
```

Supported Inventory Activities

You can include the Inventory scan operation in the activities defined in an activity plan.

Parameters of Inventory Operations

The scan operation has its own set of supported parameters. The table in this section lists the parameters for the scan operation.

Table 23. Parameters for the scan operation

Scan operation parameters		
Deadline	MobileEscalateDate_7	MobileForceMandatory
DisconnectedOperation	MobileEscalateDate_8	MobileHidden
ExecutionTimeout	MobileEscalateDate_9	MobileMandatoryDate
ExecutionTimeoutUnit	MobileEscalateMessage_0	NotifyInterval
Force	MobileEscalateMessage_1	NotifyIntervalUnit
LenientDistribution	MobileEscalateMessage_2	Priority
MobileEscalateDate_0	MobileEscalateMessage_3	RelativeDeadline
MobileEscalateDate_1	MobileEscalateMessage_4	RoamEp
MobileEscalateDate_2	MobileEscalateMessage_5	SendTimeout
MobileEscalateDate_3	MobileEscalateMessage_6	SendTimeoutUnit
MobileEscalateDate_4	MobileEscalateMessage_7	WakeOnLan
MobileEscalateDate_5	MobileEscalateMessage_8	
MobileEscalateDate_6	MobileEscalateMessage_9	

Parameter Values for the Scan Operation

The following table details the parameters of the scan operation and the possible values for each parameter.

Table 24. Parameter values

Parameter	Values	Default
Deadline	Date and time in the format: "yyyy-mm-dd hh:mm"	
DisconnectedOperation	T (true) F (false)	T
ExecutionTimeout	Integer	If not specified, the default at the last gateway in the path will be used.
ExecutionTimeoutUnit	S (seconds) M (minutes) H (hours) D (days)	S
Force	T (true) F (false)	F
LenientDistribution	T (true) F (false)	T
MobileEscalateDate_0	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_1	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_2	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_3	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_4	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_5	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_6	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_7	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_8	"yyyy-mm-dd hh:mm"	
MobileEscalateDate_9	"yyyy-mm-dd hh:mm"	
MobileEscalateMessage_0	String	
MobileEscalateMessage_1	String	
MobileEscalateMessage_2	String	
MobileEscalateMessage_3	String	
MobileEscalateMessage_4	String	
MobileEscalateMessage_5	String	
MobileEscalateMessage_6	String	
MobileEscalateMessage_7	String	
MobileEscalateMessage_8	String	
MobileEscalateMessage_9	String	
MobileForceMandatory	T (true) F (false)	T
MobileHidden	T (true) F (false)	F

Supported Inventory Activities

Table 24. Parameter values (continued)

Parameter	Values	Default
MobileMandatoryDate	"yyyy-mm-dd hh:mm"	
NotifyInterval	Integer	If not specified, the default at the last gateway in the path will be used.
NotifyIntervalUnit	S (seconds) M (minutes) H (hours) D (days)	M
Priority	PRTY_HIGH PRTY_MEDIUM PRTY_LOW	PRTY_MEDIUM
RelativeDeadline	hh:mm	The hours and minutes specified are added to the start not before time to obtain the deadline. If a deadline is not specified, the default deadline of 72 hours is used.
RoamEp	T (true) F (false)	T
SendTimeout	Integer	If not specified, the default at the last gateway in the path will be used.
SendTimeoutUnit	S (seconds) M (minutes) H (hours) D (days)	S
WakeOnLan	T (true) F (false)	F

The following example defines an activity for the scan operation:

```

<activity>
  <application>Inventory</application>
  <name>Activity1</name>
  <operation>Scan</operation>
  <parameter>
    <name>RoamEp</name>
    <value>T</value>
  </parameter>
  <parameter>
    <name>MobileForceMandatory</name>
    <value>T</value>
  </parameter>
  <parameter>
    <name>Priority</name>
    <value>PRTY_MEDIUM</value>
  </parameter>
  <parameter>
    <name>Label</name>
    <value>test</value>
  </parameter>
  <parameter>
    <name>NotifyIntervalUnit</name>
    <value>S</value>
  </parameter>

```

```

<parameter>
  <name>NotifyInterval</name>
  <value>543</value>
</parameter>
<parameter>
  <name>MobileHidden</name>
  <value>F</value>
</parameter>
<parameter>
  <name>WakeOnLan</name>
  <value>T</value>
</parameter>
<parameter>
  <name>DisconnectedOperation</name>
  <value>T</value>
</parameter>
<parameter>
  <name>Ignore</name>
  <value>F</value>
</parameter>
<parameter>
  <name>LenientDistribution</name>
  <value>T</value>
</parameter>
<parameter>
  <name>Profile</name>
  <value>test</value>
</parameter>
</activity>

```

If the parameter values are not specified, then Activity Planner uses the defaults at the last gateway that the endpoints were logged into.

Supported Pristine Manager Activities

You can include the pristine installation operation in the activities defined in an activity plan.

Parameter of Pristine Operations

The pristine installation operation has only one supported parameter: Easement.

Parameter Values for the Pristine Installation Operation

The following table details the parameter of the pristine installation operation and the possible values.

Table 25. Parameter values

Parameter	Values	Default
OSElement	String. Specifies the type of configuration element.	

The following example defines an activity for the pristine installation operation. Cortlandt is the name of the pristine machine on which the operating system is installed and Windows2000SP3 is the name of the operating system element.

```

<activity>
  <application>OSElement</application>
  <name>osl</name>
  <target_resource_type>endpoint</target_resource_type>
  <targets type="pristine">cortlandt</targets>
  <operation>PristineInstall</operation>
  <parameter>

```

```
<name>OSElement</name>
<value>Windows2000SP3</value>
</parameter>
</activity>
```

Managing Activity Plans

Activity Planner provides a command line interface to manage and control activity plans. The commands can be classified as follows:

- Commands that manage the information in the activity plan database
- Commands that manage the submission and execution of activity plans
- Commands that enable you to check on the status of the execution of activity plans or individual activities
- Commands that enable you to map error levels to return values
- Commands that enable you to perform administrative tasks

Note: To use the commands, at least the IBM Tivoli Framework role "user" is required.

To request help from the command line about a particular command, enter the command name followed by a question mark (?). For example, for information about the **wsubpln** command, enter **wsubpln ?**.

Commands are divided into the following groups:

Managing the Activity Plan Database

The activity plan database maintains information about activity plan drafts and templates. Activity Planner supports the following databases: Sybase, Oracle, DB2, Informix, MS SQL. Activity plans are stored in the databases as templates that can be used for submission and execution. Each time an activity plan is submitted an instance of the activity plan is executed.

The commands shown in Table 26 are used to manage the activity plan database.

Table 26. Commands for managing the activity plan database

Command	Purpose	Details on page
wapmfltr	Specifies one or more filters to be applied to plans, targets, or activities	91
wdelpln	Removes an activity plan from the activity plan database.	100
wexppln	Exports an activity plan stored in the activity plan database in the activity plan definition file XML format.	102
wimppln	Imports a specified activity plan in XML file format into the activity plan database	107
wlstpln	Returns a list of the activity plans maintained in the activity plan database.	108
wunlockpln	Displays a list of locked plans and unlocks plans currently locked.	126

Managing Activity Plans

You can manage activity plans by controlling the following operations:

- Submission
- Pause/resume

- Cancel
- Display
- Restart
- Update of submitted plans not yet started
- Generation of recovery plans

The commands shown in Table 27 are used for managing activity plans.

Table 27. Commands for managing activity plans

Command	Purpose	Details on page
wcntpln	Controls the execution of a specified activity plan or the activities contained within it.	98
wsubpln	Submits an activity plan to the Activity Planner engine for execution.	122
wupdpln	Updates an activity plan that has been submitted, but has not yet started.	127
wgenpln	After a failure, generates a new plan containing only the failed activities for all failed nodes.	103

Monitoring Activity Plans

You can display important information related to the execution of an activity plan such as the completion status of the plan, the status of a specific activity, or the status of an activity on a specific target.

The commands shown in Table 28 are used to monitor activity plans.

Table 28. Commands for monitoring activity plans

Command	Purpose	Details on page
wdelstat	Removes the status of a submitted plan.	101
wmonpln	Displays the status for all activities in an activity plan.	110
wsfdpln	Sets filters to automatically remove completed activity plans from the activity plan database.	117

Managing Return Values and Error Levels

You can customize the value returned by the application that indicates the results of an operation by mapping specific error levels to return values.

By default, a "successful" error level maps to a zero return value, and any other return value indicates an error. Assigning further codes to different levels of error, for example, information, warning, and fatal, is useful for Task Library tasks rather than for Software Distribution tasks.

The commands shown in Table 29 are used to manage return values and error levels.

Table 29. Commands for managing return values and error levels

Command	Purpose	Details on page
wgeterrlev	Displays current error level mappings.	106
wresetlev	Deletes the defined mapping of particular error levels to return values.	113

Managing Activity Plans

Table 29. Commands for managing return values and error levels (continued)

Command	Purpose	Details on page
wseterrlev	Maps specific error levels of the application to a return values.	115

Using Commands to Perform Administrative tasks

Activity Planner provides commands to enable you to perform the following administrative tasks:

- Start and stop the Activity Planner engine
- Modify the RIM object name used to access the activity plan database
- Modify the user password that enables the Activity Planner engine to function properly

The commands shown in Table 30 are used to perform administrative tasks.

Table 30. Commands for performing administrative tasks

Command	Purpose	Details on page
wapmrim	Modifies the RIM object used to access the activity plan database.	97
wsetapmpw	Modifies the user password that enables the Activity Planner engine to function correctly.	114
wstartapm	Starts the Activity Planner engine.	120
wstopapm	Stops the Activity Planner engine.	121
wapmgui	Starts the Activity Plan Editor and Activity Plan Monitor graphical user interface from the CLI.	94

Using Plug-ins to Register Applications

By using plug-ins, you can register an application with the Activity Planner and use the Activity Planner to automate and monitor the execution of the application itself.

Activity Planner currently supports Tivoli Management Framework tasks, Software Distribution, and Inventory operations. The Tivoli Management Framework task plug-in is automatically registered with the Activity Planner at installation time, while the Software Distribution and Inventory plug-ins need to be registered manually using the **“wapmplugin” on page 95** command. You register the Inventory plug-in using the `reg_invscan_plugin.sh` script. For more information, see “Enabling IBM Tivoli Configuration Manager Components for the Activity Planner” on page 7.

The command shown in Table 31 is used to register a plug-in with the Activity Planner.

Table 31. Commands for registering plug-ins

Command	Purpose	Details on page
wapmplugin	Registers a plug-in with the Activity Planner.	95

Full details of the commands are as follows:

wapmfltr

Specifies one or more filters to be applied to plans, targets, or activities so that only the objects matching certain criteria are displayed in the Activity Plan Monitor command line. You can also modify or delete an existing filter.

Note: You can also manage plans, activities, or targets from the GUI. For information about this, see “Activity Plan Status” on page 45.

Syntax

wapmfltr -l

wapmfltr {-p|-t|-a} -n *new_filter_name* [-u *user@hostname.domain*] [-f "*key operator value*"]...

wapmfltr [-p|-t|-a] -r *filter_name* [-f "*key operator value*"]...

wapmfltr -v *filter_name*

wapmfltr -d *filter_name*

Description

The **wapmfltr** command specifies one or more filters to be applied to plans, targets, or activities, modifies existing filters, applies existing filters to different kinds of objects, for example, modifies a filter originally created for plans so that it applies to targets or activities. This command also allows the user to modify or delete an existing filter.

Options

-l Displays a list of existing filters.

-p|-t|-a Specifies whether the new filter created using the **-n** option applies to plans, targets, or activities.

-n *new_filter_name* Specifies a name for the new filter. Maximum supported number of characters is 198. The following special characters are not supported: ("), (<), (>).

-u *user@hostname.domain* Enables you to specify the owner of the filter you create when you have the APM_View role.

-f "*key operator value*" Specifies a value for filter keywords. To cancel an existing setting, type the minus sign enclosed in round brackets, (-), after the = operator. Default keys, definitions, operators, and values are described in Table 32:

Table 32. Keys, operators and values for the **-f** option.

KEY	DEFINITION	OPERATOR	OPERATOR SHORTCUT	VALUE	VALUE SHORTCUT
NAME	Specifies a name to be used as filter.	= LIKE		<i>string</i>	

Table 32. Keys, operators and values for the -f option. (continued)

KEY	DEFINITION	OPERATOR	OPERATOR SHORTCUT	VALUE	VALUE SHORTCUT
STATUS	Specifies the status to be used as filter.	=		Cancel pending Canceled Canceled by condition Failed Paused Registered Started Starting Successful Waiting	CP C CC F P R St Sa Su W
REC_NUM	Specifies a recursion number to be used as filter.	BETWEEN EQUAL EXTERNAL GREATERTHAN LESSTHAN	BT EQ EX GT LT	One or more recursion values, separated by commas and enclosed in round brackets.	
START_TIME	Specifies a start time to be used as filter.	AFTER BEFORE BETWEEN EXTERNAL	BT AFTER BEFORE EX	Date and time in the yyyy-mm-dd format, enclosed in round brackets.	
COMPLETE_TIME	Specifies a completion time to be used as filter.	AFTER BEFORE BETWEEN EXTERNAL	BT AFTER BEFORE EX	Date and time in the yyyy-mm-dd format, enclosed in round brackets.	
DEFAULT_FILTER	Determines whether the specified filter is set as default filter.	=		y	

-r *filter_name*

Applies the modifications defined using the -f option to the specified filter.

-v *filter_name*

Displays the filter criteria defined for the specified filter.

-d *filter_name*

Deletes the specified filter.

Authorization

APM_Manage

Return Values

The **wapmfltr** command returns one of the following values:

0 Indicates that **wapmfltr** started successfully.

Nonzero

Indicates that **wapmfltr** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To create a new filter with name test to display only plans in failed status, enter the following command:

```
wapmfltr -p -n test -f "STATUS=Failed"
```

2. To modify filter test so that it applies to activities whose names start with ccm, enter the following command:

```
wapmfltr -a -r test -f "NAME LIKE ccm%"
```

3. To set filter computer on plans with recursion number external to the 3, 5 interval as default filter, enter the following command:

```
wapmfltr -p -n computer -f "REC_NUM EXTERNAL (3,5)" -f "DEFAULT_FILTER=y"
```

4. To modify the default filter setting for filter computer, so that it is no longer the default filter, enter the following command:

```
wapmfltr -r computer -f "DEFAULT_FILTER= -"
```

5. To define a new filter called planstest filtering targets with names beginning with part, in Failed, Successful, and Canceled by condition statuses, starting between June 6, 2003 and May 5, 2005, and completing by June 6, 2004, enter the following command:

```
wapmfltr -t -n planstest -f "NAME LIKE part%" -f "STATUS = Failed,Su,CC"
-f "START_TIME BT (2003-06-06,2006-05-05)"
-f "COMPLETE_TIME BEFORE (2004-06-06)"
```

See Also

wmonpln

wapmgui

Starts the Activity Plan Editor and Activity Plan Monitor graphical user interface from the CLI.

Note: In a Window environment the user that starts the GUIs must belong to the Tivoli_Admin_Privileges group.

Syntax

wapmgui ed

wapmgui mon

Description

The **wapmgui** command starts the Activity Plan Editor and Activity Plan Monitor graphical user interface from the CLI. Because this command is a bash script it must be run in the bash environment on supported Windows operating systems: precede the command with the string **sh**. To open the Activity Planner GUI, the user specified in the login panel must have at least the APM_VIEW role.

Options

ed Starts the Activity Plan Editor graphical user interface.

mon Starts the Activity Plan Monitor graphical user interface.

Authorization

None

Return Values

The **wapmgui** command returns one of the following values:

0 Indicates that **wapmgui** started successfully.

Nonzero

Indicates that **wapmgui** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To start the Activity Plan Editor graphical user interface, enter the following command:

```
wapmgui ed
```

2. To start the Activity Plan Monitor graphical user interface on supported Windows operating systems, enter the following command:

```
sh wapmgui mon
```

See Also

None

wapmplugin

Registers or removes a plug-in.

Syntax

wapmplugin [-a *name* [-p *package*] [-d *description*] [-f *toc_file*] [-c *classpath*]]

wapmplugin [-s *name* [-p *package*] [-d *description*] [-f *toc_file*] [-c *classpath*]]

wapmplugin -r *name*

wapmplugin -l

Description

The **wapmplugin** command registers or removes a plug-in with Activity Planner.

Options

-a *name*

Registers a new plug-in.

-s *name*

Modifies the plug-in parameters.

-p *package*

Specifies the package name of the classes implementing the plug-in. The recommended naming convention is as follows: *com.tivoli.apm.plugin.<xyz>*;

-d *description*

Specifies a description for the plug-in.

-f *toc_file*

Specifies the path to a .toc file containing the list of files composing the plug-in.

-c *classpath*

Specifies the classpath to the plug-in classes and resources. The following built-in variables are available: \$(BINDIR), \$(DBDIR), \$(SEP), \$(LCF_BINDIR). For information about the first two variables, refer to the *Tivoli Management Framework: Planning for Deployment Guide* and *IBM Tivoli Enterprise Console: Installation Guide*. The \$(SEP) variable specifies the character used as a separator in Windows and UNIX, and the \$(LCF_BINDIR) variable specifies the path to the Tivoli Management Framework DSWIN installation directory on the endpoints.

-r *name*

Removes the specified plug-in.

-l

Displays a list of the registered plug-ins.

Authorization

One of the following: APM_Admin or install_product

Return Values

The **wapmplugin** command returns one of the following values:

0 Indicates that **wapmplugin** started successfully.

Nonzero

Indicates that **wapmplugin** failed due to an error. See Table 33 on page 130 for more information about return values.

wapmplugin

Examples

To register the Software Distribution plug-in, enter the following command:

```
wapmplugin -a SoftwareDistribution -d "Software Distribution plug-in for APM" \  
-p com.tivoli.apm.plugin.sd -c '' -f swd_toc
```

See Also

None

wapmrim

Displays and optionally changes the RIM object name.

Syntax

wapmrim [*RIM_object_label*]

Description

The **wapmrim** command displays and optionally changes the RIM object name used to access the activity plan database. The default RIM object name is **planner**. To make the RIM object name change effective after running the command, close the GUI and stop the engine using the “**wstopapm**” on page 121 command, that is, if the engine is active at the moment the command is run. Because this command is a bash script it must be run in the bash environment on supported Windows operating systems: precede the command with the string **sh**.

Options

[*RIM_object_label*]

Submitting the **wapmrim** command without this option, displays the current RIM object name being used. Specifying a RIM object name with the command, changes the name to the name specified in the command.

Authorization

One of the following: APM_Admin, senior, admin, super

Return Values

The **wapmrim** command returns one of the following values:

0 Indicates that **wapmrim** started successfully.

Nonzero

Indicates that **wapmrim** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To display the current RIM object name being used by the activity plan database, enter the following command:

```
wapmrim
```
2. To change the current RIM object name to **manager** on supported Windows operating systems, enter the following command:

```
sh wapmrim manager
```

See Also

“**wstopapm**” on page 121

wcntpln

Controls the execution of the specified activity plan or activities.

Note: You can also control the execution of an activity plan or activities from the GUI. For information about this, see “Controlling the Execution of Activity Plans and Activities” on page 51.

Syntax

```
wcntpln -c [-e activity]... [-t targetName]... actPlan_name
```

```
wcntpln -p [-e activity]... [-t targetName]... actPlan_name
```

```
wcntpln -r [-e activity]... [-t targetName] ... actPlan_name
```

```
wcntpln -s [-e activity]... [-t targetName]... actPlan_name
```

```
wcntpln -f [-e activity]... actPlan_name
```

```
wcntpln -R {s / v} [-e activity]... [-t targetName]... actPlan_name
```

Description

The **wcntpln** command controls the execution of either the specified activity plan or the individual activities contained within it. Some options, **-c -p -r**, might not be supported depending on the plug-in you installed.

Options

- c** Cancels the execution of the specified activity plan or activity by canceling those activities that have not yet completed. For activities that have not yet been executed, the Activity Planner engine performs the cancel operation. For an activity that has been executed but not yet complete, the activity is canceled by the application to which it belongs. For example, for an activity related to a Software Distribution operation, the activity is canceled by the MDist 2 service. Refer to the *Tivoli Management Framework: User's Guide* for more information about the MDist 2 service. For activities of the type Task, the cancel option is not supported.
- p** Pauses the execution of the specified activity plan or activity. For activities not yet executed, the activities are paused by the Activity Planner engine. For activities executed but not yet complete, the pause is managed by the application to which the activity belongs. For activities of the type Task, the pause option is not supported.
- r** Resumes the execution of the specified activities in the activity plan that are in the paused state.
- s** Restarts the execution of the specified activities in an activity plan that failed on targets of a previous execution or that are in the canceled state.
- R {s / v}** Controls the recursion mechanism of a recursive plan.
 - s** Specifies to complete the execution of any instances currently in execution, and to interrupt the recursion mechanism of the specified plan.
 - v** Displays the parameters defined for the specified recursive plan.

-e activity

The name of an activity. Only one activity can be defined with a single use of the **-e** option. To define more activities, you must include the option multiple times. When the **-e activity** option is specified, the **wcntpln** command is performed on the specified activity or activities and not on the activity plan.

-t target

Specifies the target the operation will be executed on.

-f

Changes the specified operation state to canceled, even though submitted operations are not canceled. This setting has effect only on the status of the activity or activity plan in Activity Planner, but has no effect on the operations associated to the activity or activity plan, and no activity is performed at MDist 2 or Tivoli Web Gateway level. When using this command to cancel distributions addressed to devices, you can use the **wwebgw -c** command to cancel the distribution. For more information on this command, refer to *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*. Use this option only for recovery purposes. This setting deploys the same functionality as the **Cancel Force** menu item. For more information on this menu item, see “Controlling the Execution of Activity Plans and Activities” on page 51.

actPlan_name

The name of the activity plan.

Authorization

APM_Manage

Return Values

The **wcntpln** command returns one of the following values:

0 Indicates that **wcntpln** started successfully.

Nonzero

Indicates that **wcntpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To pause ActPlan1 that has already been started, enter the following command:

```
wcntpln -p ActPlan1
```
2. To resume the execution of ActPlan 1 which was formerly paused, enter the following command:

```
wcntpln -r ActPlan1
```
3. To cancel the execution of activity1 contained in ActPlanY, enter the following command:

```
wcntpln -c -e activity1 ActPlanY
```
4. To cancel the execution of ActPlanY, enter the following command:

```
wcntpln -c ActPlanY
```
5. If activity2 contained in ActPlanZ has failed, you can restart the execution of activity2 by entering the following command:

```
wcntpln -s -e activity2 ActPlanZ
```

See Also

None

wdelpln

Removes an activity plan from the activity plan database.

Note: You can also delete an activity plan from the GUI. For information about this, see “Deleting Activity Plans or Activities” on page 44.

Syntax

wdelpln *actPlan_name*

Description

The **wdelpln** command removes an activity plan from the activity plan database.

Options

actPlan_name

The name of the activity plan.

Authorization

APM_Edit

Return Values

The **wdelpln** command returns one of the following values:

0 Indicates that **wdelpln** started successfully.

Nonzero

Indicates that **wdelpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To remove actplan1 from the activity plan database, enter the following command:

```
wdelpln actplan1
```

See Also

wsfdpln

wdelstat

Removes the status of a submitted activity plan.

Note: You can also delete the status of an activity plan from the GUI. For information about this, see “Deleting Completed Activity Plans” on page 53.

Syntax

wdelstat *-r recursion_number ActPlanName*

wdelstat *-a ActPlanName*

Description

The **wdelstat** command removes the status of a submitted plan if the plan is completed. If the plan is recursive, you can specify to delete the status of all instances of the plan or specify a particular instance. In order to use **wdelstat** on a recursive plan, you must stop the recursion with the **wcntpln -Rs PlanName** command. If you deleted all recursion instances for a recursive plan before the recursion mechanism completes or is stopped, the entry for the submitted plan is maintained in the Activity Planner database. To delete the plan, stop the recursion mechanism using “wcntpln” on page 98, then delete the plan using “wdelpln” on page 100.

Options

- r** Specifies a particular instance of the specified activity plan is to be removed.
- a** Specifies that the status of all instances of the specified activity plan is to be removed.

ActPlanName

The name of a submitted activity plan.

Authorization

APM_Manage

Return Values

The **wdelstat** command returns one of the following values:

0 Indicates that **wdelstat** started successfully.

Nonzero

Indicates that **wdelstat** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To remove the status of all instances of the recursive plan ActPlan2 from the activity plan database, enter the following command:

```
wdelstat -a ActPlan2
```
2. To remove the status of only the first instance of recursive plan ActPlan2, enter the following command:

```
wdelstat -r1 ActPlan2
```

See Also

wdfdpln

wexppln

Exports an activity plan stored in the activity plan database in the activity plan definition file XML format.

Note: You can also export an activity plan into XML format from the GUI. For information about this, see “Saving the Activity Plan” on page 43.

Syntax

wexppln [-f *filePath* [-o]] *ActPlan_name*

Description

The **wexppln** command exports the activity plan in the activity plan definition file format.

Options

-f *filePath*

Stores the activity plan definition file output in the specified path. If not specified, the activity plan definition file is displayed in standard output.

-o

Overwrites an existing activity plan definition file with the current activity plan.

ActPlan_name

The name of the activity plan.

Authorization

APM_View

Return Values

The **wexppln** command returns one of the following values:

0

Indicates that **wexppln** started successfully.

Nonzero

Indicates that **wexppln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To store the actplan1 in the activity plan definition file format in c:\plans\plan1.xml, enter the following command:

```
wexppln -f c:\plans\plan1.xml actplan1
```

See Also

wimppln

wgenpln

After a failure, generates a recovery plan template containing only the failed activities for all failed nodes.

Note: You can also generate a recovery plan template from the GUI. For information about this, see “Generating a Recovery Plan” on page 52.

Syntax

```
wgenpln -i plan_instance_identifier [-D option_to_modify=value]... [-r recursionNumber]
[-s] [-o] -n plan_name
```

```
wgenpln -p plan_instance_name [-D option_to_modify=value]... [-r recursionNumber]
[-s] [-o] -n plan_name
```

Description

The **wgenpln** command generates a new plan template, after the failure of the original one, containing only the failed activities for all failed nodes. Any conditions present in the original plan are not maintained in the recovery plan.

Options

-p *plan_instance_name*

The name of the original plan instance.

-i *plan_instance_identifier*

The identifier of the original plan instance.

-D *option_to_modify=value*

Specifies that some options defined in the original plan are overridden in the new plan template before it is submitted. The options and their possible values are:

name

- string

priority

- low
- medium
- high

mail_address

- string
- - (disables mail notification)

post_notice

- y
- n

submit_paused

- y
- n

start_not_before

- "yyyy-mm-dd hh:mm"
- - (disables start_not_before setting)

complete_not_after

- "yyyy-mm-dd hh:mm"
- hh:mm
- - (disables complete_not_after setting)

stop_plan_on_error

- warning
- error
- fatal

- ignore (disables stop_plan_on_error setting)
- cancel_at_cutoff**
- y
 - n
- recursion_type**
- di (date_interval)
 - ti (time_interval)
 - dom (days_of_month)
 - dow (days_of_week)
 - nr (no recursion)

You can modify the following options only if the **recursion_type** option is specified.

- recursion_info**
- if recursion_type=di, *yy/mm/dd*
 - if recursion_type=ti, *hh:mm*
 - if recursion_type=dom, *d1,d2,d3,...dn*
 - if recursion_type=dow, *d1,d2,d3,...dn*
- expiration_date**
- *"yyyy-mm-dd hh:mm"*
 - *hh:mm*
 - - (disables expiration_date setting)
- stop_on_overlap**
- y
 - n
- stop_rec_on_error_level**
- warning
 - error
 - fatal
 - ignore (disables stop_rec_on_error_level setting)
- target_resolution**
- static
 - dynamic

-r *recursionNumber*

Specifies the original plan instance from which the recovery plan is generated. As recursive plans are submitted multiple times, the plan name is no longer unique, and a recursion number must be specified. If no recursion number is specified, 0 is assumed.

-s Submits the generated plan template.

-o Specifies that the original plan in the activity planner database is overwritten if a plan with the same name already exists.

-n *plan_name*

The name of the activity plan template.

Authorization

APM_Manage

Return Values

The **wgenpln** command returns one of the following values:

0 Indicates that **wgenpln** started successfully.

Nonzero

Indicates that **wgenpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To generate a new plan called RecPlan1 after the failure of ActPlan1 and change the priority to High, enter the following command:
2. To generate a new plan called RecPlan1 after the failure of the fifth recursion instance of the ActPlan1 plan and submit the plan immediately, enter the following command:

```
wgenpln -p ActPlan1@10/01/02 -D priority=high -n RecPlan1
```

```
wgenpln -p ActPlan1@10/01/02 -r5 -s -n RecPlan1
```

3. To generate a new plan called RecPlan1 after the failure of ActPlan1, and overwrite the original plan, enter the following command:

```
wgenpln -i 10106668898485176470 -o -n ActPlan1
```

See Also

None

wgetterlev

Displays current error level mapping.

Syntax

wgetterlev [-l s] [-a *applicationId*]

wgetterlev [-l w] [-a *applicationId*]

wgetterlev [-l f] [-a *applicationId*]

wgetterlev [-l e] [-a *applicationId*]

Description

The **wgetterlev** command displays the current mapping of error levels, or only those error levels specified.

Options

-l s | w | f | e

Specifies the error level.

s Successful

w Warning

f Fatal

e Error

-a *applicationId*

Specifies the type of application. Possible values can vary depending on the plug-in you registered.

Authorization

APM_View

Return Values

The **wgetterlev** command returns one of the following values:

0 Indicates that **wgetterlev** started successfully.

Nonzero

Indicates that **wgetterlev** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To view the current mapping of error levels, enter the following command:

```
wgetterlev
```

The following is the output of the command:

Error level	Return code	Application type
w	23	TaskLibrary
s	18	TaskLibrary

2. If the default return values are set for the Task Library application, you can display the current return value associated with success for a Task Library operation by entering the following command:

```
wgetterlev -l s -a TaskLibrary
```

The output of the command is 0.

See Also

wresetlev, **wseterrlev**

wimppln

Imports an activity plan in the activity plan database.

Syntax

wimppln [-c *encoding*] -f *plan_def_file*

wimppln [-o] [-c *encoding*] -f *plan_def_file*

wimppln [-e] [-c *encoding*] -f *plan_def_file*

Description

The **wimppln** command imports an activity plan, in the XML file format, into the activity plan database and returns an error if an activity plan with the same name is already present, unless the option to overwrite is specified.

Options

-c *encoding*

Specifies the encoding to be used for the activity plan definition file. For a complete list of the supported encoding types, see the Sun Microsystems Web site.

-e Specifies that the imported activity plan is saved as a draft. If this option is not specified, the imported activity plan is saved as a template.

-o Overwrites an existing activity plan with the current activity plan.

-f *plan_def_file*

Specifies the file name of the activity plan to import.

Authorization

APM_Edit

Return Values

The **wimppln** command returns one of the following values:

0 Indicates that **wimppln** started successfully.

Nonzero

Indicates that **wimppln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To import the activity plan described in the XML file `actplan1.xml`, and replace it if it already exists in the activity plan database, enter the following command:

```
wimppln -o -f d:\testplans\actplan1.xml
```

2. To import the activity plan described in the XML file `actplan2.xml` and save it as a draft, enter the following command:

```
wimppln -e -f d:\testplans\actplan2.xml
```

See Also

wexppln

wlstpIn

Returns a list of names of activity plan templates and drafts maintained in the activity plan database.

Syntax

wlstpIn [*plan_name*]

wlstpIn [-l] [*plan_name*]

wlstpIn [-e] [*plan_name*]

wlstpIn [-Q*option_to_query=value*]

Description

The **wlstpIn** command returns a list of activity plan templates and drafts maintained in the activity plan database and optionally, descriptions of plans.

Options

- l Displays a description of all activity plan templates or drafts, if available, depending on whether the -e option is specified. If an activity plan template name is specified, only the description of the specified activity plan template is displayed.
- e Displays a list of all activity plan drafts. When used with the -l option, also displays a description of the activity plan drafts, if available. If an activity plan draft name is specified, only the description of the specified activity plan draft is displayed.

plan_name

The name of the specified activity plan.

-Q*option_to_query=value*

Specifies filters to be applied to the query. The query returns all submitted plans, templates, and drafts, based on their recursion settings. The option and its possible values are:

recursion_type

- * Lists all submitted plans, templates, and drafts, regardless of whether they are recursive
- di Lists all submitted plans, templates, and drafts with recursion type Date Interval
- ti Lists all submitted plans, templates, and drafts with recursion type Time Interval
- dom Lists all submitted plans, templates, and drafts with recursion type Day of month
- dow Lists all submitted plans, templates, and drafts with recursion type Day of week
- nr Lists all submitted plans, templates, and drafts with no recursion settings.
- r Lists all recursive submitted plans, templates, and drafts.

Authorization

APM_View

Return Values

The **wlstpln** command returns one of the following values:

0 Indicates that **wlstpln** started successfully.

Nonzero

Indicates that **wlstpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To display a list of all activity plans stored as templates in the activity plan database, enter the following command:

```
wlstpln
```

2. To display a list of all activity plans stored as drafts in the activity plan database, enter the following command:

```
wlstpln -e
```

3. To display a list of all activity plans stored as drafts in the activity plan database with the corresponding description if available, enter the following command:

```
wlstpln -l -e
```

4. To display a description of activity plan ActPlan1 stored in the activity plan database, enter the following command:

```
wlstpln -l ActPlan1
```

5. To display a list of all plans with recursion type set to days of week, enter the following command:

```
wlstpln -Qrecursion_type=dow
```

The following output is displayed:

Plan	Status	Recursion
test	Template	1
test_plan@06/10/03 09:19:34	Submitted	1,2,4
submission	Draft	4,5
jupiter	Submitted	st
juno	Template	2,3
venus	Draft	3,5

The st symbol indicates that the recursion for the specified plan was stopped.

See Also

None

wmonpln

Displays status information related to an activity plan that has been submitted.

Note: You can also view the status of an activity plan from the GUI. For information about this, see “Activity Plan Status” on page 45.

Syntax

wmonpln **-l** **[-w]**

wmonpln **-f** *[FilterName]* **[-w]**

wmonpln **-p** *ActPlanName* **[-r RecNum]****[-a ActName** **[-t Target]** **[-s status,...]** **[-n maxRows]** **[-w]**

wmonpln **-p** *ActPlanName* **[-r RecNum]****[-a ActName** **[-g]** **[-s status,...]** **[-n maxRows]** **[-w]**

wmonpln **-i** *PlanId* **[-r RecNum]****[-a ActName** **[-t Target]** **[-s status,...]** **[-n maxRows]** **[-w]**

wmonpln **-i** *PlanId* **[-r RecNum]****[-a ActName** **[-g]** **[-s status,...]** **[-n maxRows]** **[-w]**

Description

The **wmonpln** command displays statistical information related to the current status of an activity plan, its activities, and activities on specific targets.

Options

- l** Displays, in table format, status information for all plans and instances currently in submission or completed and not yet deleted from the activity plan database. The table displays the following information: plan identifier, recursion, status, plan name. The table is displayed truncated. Use **wmonpln -l -w** to display the whole table and display start and completion time.
- w** Formats information in a table. When used with the **-l** or the **-f** option, displays the output for the following information: plan identifier, recursion, status, plan name, start and completion time of all plans and instances currently in submission or completed and not yet deleted from the activity plan database. When used with the **-p** or **-i** option, it returns the distribution ID and the application type for the specified activity.
- f FilterName**
Displays a list of plans matching the filter criteria defined in the specified filter.
- p plan_name**
The name of an activity plan.
- i planid**
The unique identifier number of an activity plan.
- r RecNum**
The name of a specific instance of a recursive activity plan.
- a ActName**
The name of a specific activity.

-n maxRows

Limits the number of targets displayed for the specified activity.

-s status

Filters the specified activity targets based on their status. You specify status by entering the word in full, for example: started, waiting, and so on. You can specify several statuses, by separating them using a comma. Spaces are not allowed. This option is not case-sensitive.

-t target

Displays status information regarding the execution of the specified activity on the specified target.

-g

Displays the status of the specified activity on the manager addressed by the activity. The manager could be a gateway, managed node, or endpoint depending on the target type. This option is meaningless if you set the **retrieve_gateways_info** parameter to **no** in the apm.ini file. For devices, the gateway is represented by an endpoint, therefore the **retrieve_gateways_info** parameter has no effect. For more information on the **retrieve_gateways_info** parameter, see “The Activity Planner Configuration File” on page 8.

Authorization

APM_View

Return Values

The **wmonpln** command returns one of the following values:

0 Indicates that **wmonpln** started successfully.

Nonzero

Indicates that **wmonpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To display the status information of all plans and instances currently in submission or completed in a table, enter the following command:

```
wmonpln -l -w
```

The following output is displayed:

Plan ID	Rec	Status	Name	Started at	Completed at
9704980686912736793	0	Waiting	WUBPLNTEST@2003-05-10 16:36:48	2003-05-10 16:36:48	2003-05-10 16:47:46
9704983915952845037	0	Waiting	WUBPLNTEST@2003-05-10 16:36:09	2003-05-10 16:36:09	2003-05-10 16:58:05
9704986383202968326	0	Waiting	WUBPLNTEST@2003-06-10 16:57:18	2003-05-10 16:57:18	2003-05-10 17:02:03
9704991828932653332	0	Waiting	WUBPLNTEST@2003-06-10 17:06:22	2003-05-10 17:06:22	2003-05-10 17:20:43
9704992343672771552	0	Waiting	WUBPLNTEST@2003-05-10 17:07:14	2003-05-10 17:07:14	2003-05-10 17:46:14
9704983915952845037	0	Waiting	WUBPLNTEST@2003-05-10 16:36:09	2003-05-10 16:36:09	2003-05-10 16:58:05

2. To display details on the status of activities for the activity plan having the 9704980686912736793 plan ID, enter the following command:

```
wmonpln -i 9704980686912736793
```

The following output is displayed:

Activity	Status	Targets	Completed	Successful	Failed
Activity2	Successful	1	1 (100%)	1 (100%)	0 (0%)
Activity1	Successful	1	1 (100%)	1 (100%)	0 (0%)

- To display details on the status of Activity2 in the activity plan having the 9704980686912736793 plan ID, enter the following command:

```
wmonpln -i 9704980686912736793 -a Activity2
```

The following output is displayed:

Target	Status	Started at	Completed at	Return
vienna-ep	Successful	2003/06/17 11:22	2003/06/17 11:43	0

- To display the status of the t1 activity on the target named saturn for the activity plan having the 9704980686912736793 plan ID, enter the following command:

```
wmonpln -i 9704980686912736793 -r 0 -a t1 -t saturn
```

The following output is displayed:

```
Target:          saturn
Manager:         gw
Started at:      2003/05/10 16:36
Completed at:    2003/05/10 15:47
Status:          Successful
Return:          0
Message:         DISSE0001I Operation successful
```

Note: Due to message length constraints, some messages may be truncated. For more information on the message, refer to *Messages and Codes* and search for the message by its ID number.

- To display the status for activity TestAct1 in plan TestPlan1 limiting the number of lines displayed to 5 and the target status to waiting and failed, enter the following command:

```
wmonpln -p TestPlan1 -a TestAct1 -n 5 -s waiting,failed
```

See Also

91

wresetlev

Deletes any previously specified mapping of error levels and resets the default values.

Syntax

wresetlev [-l s] *applicationID*

wresetlev [-l w] *applicationID*

wresetlev [-l f] *applicationID*

wresetlev [-l e] *applicationID*

Description

The **wresetlev** command resets any previously defined mapping of error levels to the specified values. If no value is specified, the mapping of all error levels is reset to defaults.

Options

-l s | w | f | e
 Specifies the error level.
 s Successful
 w Warning
 f Fatal
 e Error

applicationID

Specifies the type of application. Possible values can vary depending on the plug-in you registered.

Authorization

APM_Admin

Return Values

The **wresetlev** command returns one of the following values:

0 Indicates that **wresetlev** started successfully.

Nonzero

Indicates that **wresetlev** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To reset the error level to interpret a return value of 0 as a successful (default) error level for a TaskLibrary operation, enter the following command:

```
wresetlev -l s TaskLibrary
```

See Also

wseterrlev, **wgeterrlev**

wsetapmpw

Changes the user password for the Activity Planner engine. You must use this command to synchronize the password maintained by the engine with that on the operating system. For more information on password management, see “Activity Planner Installation Notes” on page 139.

Syntax

wsetapmpw

Description

When you enter the command, you are prompted to insert the old password, then to enter and confirm the new password. If used during the installation, the Activity Planner will not work.

Options

None.

Authorization

One of the following: APM_Admin, senior, admin, super.

Return Values

The **wsetapmpw** command returns one of the following values:

0 Indicates that **wsetapmpw** started successfully.

Nonzero

Indicates that **wsetapmpw** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To change the password for the Activity Planner engine, enter the following command:

```
wsetapmpw
```

See Also

None.

wseterrlev

Maps specific error levels to return values.

Syntax

wseterrlev *-l s -c returncode applicationID*

wseterrlev *-l s -f filename applicationID*

wseterrlev *-l w -c returncode applicationID*

wseterrlev *-l w -f filename applicationID*

wseterrlev *-l f -c returncode applicationID*

wseterrlev *-l f -f filename applicationID*

wseterrlev *-l e -c returncode applicationID*

wseterrlev *-l e -f filename applicationID*

Description

The **wseterrlev** command maps specific error levels to return values. As Activity Planner by default interprets a return value of 0 as a success and any other return value as a failure, you have to map different return values from your application to success or failure states for Activity Planner to interpret them correctly.

Options

-l s | w | f | e

Specifies the error level.

s Successful

w Warning

f Fatal

e Error

-c returncode

Specifies the return value to map to the specified error level.

-f filename

A file containing a list of return values to be mapped to the specified error level.

applicationId

Specifies the type of application. Possible values are SoftwareDistribution, Inventory, or TaskLibrary, depending on the plug-ins you registered.

Authorization

APM_Admin

Return Values

The **wseterrlev** command returns one of the following values:

0 Indicates that **wseterrlev** started successfully.

Nonzero

Indicates that **wseterrlev** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. If the TaskLibrary operation returns 5 as a success state, Activity Planner interprets it as a failure, as it is different from 0. To map the return value 5 to indicate a successful TaskLibrary operation, enter the following command:

```
wseterrlev -l s -c 5 TaskLibrary
```

2. To map the return value 7 to indicate a failed TaskLibrary operation, enter the following command:

```
wseterrlev -l f -c 7 TaskLibrary
```

See Also

wgeterrlev, wresetlev

wsfdpln

Sets filters to automatically remove a set of submitted activity plans from the activity plan database.

Note: You can also set filters for an activity plan from the GUI. For information about this, see “Cleaning Up the Database” on page 54.

Syntax

wsfdpln *[[*-D *ref_date*] *[-T ref_time]* *[-s status]* *[-y c | s]* *[-f]* *-o olderthan -d date_interval]*

wsfdpln *[[*-D *ref_date*] *[-T ref_time]* *[-s status]* *[-y c | s]* *[-f]* *-o olderthan -t time_interval]*

wsfdpln *[[*-D *ref_date*] *[-T ref_time]* *[-s status]* *[-y c | s]* *[-f]* *-o olderthan -w days_of_week]*

wsfdpln *[[*-D *ref_date*] *[-T ref_time]* *[-s status]* *[-y c | s]* *[-f]* *-o olderthan -m day_of_month]*

wsfdpln *[[*-D *ref_date*] *[-T ref_time]* *[-s status]* *[-y c | s]* *[-f]* *-o olderthan -O]*

wsfdpln *-S*

wsfdpln *-o*

Description

The **wsfdpln** command uses filters to automatically remove a submitted activity plan from the activity plan database.

Options

-D *ref_date*

Specifies the date, in the format *yyyy-mm-dd*, on which the command is issued.

-T *ref_time*

Specifies the time, in the format *hh:mm*, at which the command is issued.

Note: If the **-D *ref_date*** and **-T *ref_time*** options are not specified, the current date and time at which the command is issued is used.

-s *status*

Specifies the state of the plans to be removed.

c Plans in the complete and canceled state.

e Plans in the complete state with failure.

s Plans successfully completed.

-y *c | s*

Indicates that either the started time or the completed time of an activity plan is to be used as a reference when calculating the **-o *olderthan*** option.

c Indicates that the completed time of the plan is to be used as reference when calculating the **-o *olderthan*** option. This is the default.

- s** Indicates that the started time of the plan is to be used as reference when calculating the **-o olderthan** option.
- f** Remove the specified activity plan, independent of its state.
- o olderthan**
Specifies the number of days that must have elapsed since the completion or start of a plan for the plan to be included in the cleanup. Any plan that is older than the specified number of days is deleted.
- d date_interval**
Specifies in the format "yy/mm/dd" how often to launch the cleanup process. For example, a date interval of 00/02/00 executes the cleanup every two months. A date interval of 00/02/01 executes the cleanup every two months and one day. A date interval of 01/00/00 executes the cleanup once a year. Calculation of the date interval starts at the reference date and time, specified using the **-D** and **-T** options.
- t time_interval**
Specifies a time interval, in the format *hh:mm*. For example, a time interval of 02:00 executes the cleanup every two hours. A time interval of 00:20 executes the cleanup every twenty minutes. Calculation of the time interval starts at the reference date and time, specified using the **-D** and **-T** options.
- w days_of_week**
Specifies days of the week, each represented by a number from 1 through 7, where 1 represents Sunday. For example, 1,6 executes the cleanup on Sunday and Friday. To specify a range of days, separate the two numbers with a hyphen. For example, 1-6 executes the cleanup on each day Sunday through Friday, starting from the reference date and time, specified using the **-D** and **-T** options.
- m days_of_month**
Specifies days of the month represented by numbers from 1 through 31, separated by commas or hyphens. For example 1-5, specifies the 1st, 2nd, 3rd, 4th, and 5th day of the month, starting from the reference date and time, specified using the **-D** and **-T** options.
- O** Specifies that the cleanup is to be performed one time only.
- S** Displays the current cleanup settings.
- r** Specifies that the values of options are to be set to the default values.

Authorization

APM_Manage

Return Values

The **wsfdpln** command returns one of the following values:

- 0** Indicates that **wsfdpln** started successfully.

Nonzero

Indicates that **wsfdpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To remove plans twice a month, on the 1st and 15th of the month, that completed successfully 60 days ago, enter the following command:

```
wsfdpln -s s -o 60 -m 1,15
```

2. To display the current cleanup settings, enter the following command:

```
wsfdpln -S
```

The following is example output of the command:

```
[I] APM-000060 Automatic cleanup settings

Frequency:      wsfdpln      No recursion
Reference date:      2005-12-11
Reference time:      00:00
Force option:      false
Time type:      s
Time elapsed since plan started/completed: 0
Status:      0
```

In this example, no cleanup operation is performed.

3. To remove plans that are in canceled or success state, and that started at least 10 days before July 15, 2005, enter the following command:

```
wsfdpln -D 2005-07-15 -T 15:00 -s c s -y s -o 10 -d 00/00/05
```

The remove process starts on July 15, 2005 at 15.00 and is repeated every fifth day. The plans in canceled or success state at these times and that started before July 05, 2005 are deleted.

4. To remove the plans that have been completed and that are in canceled status, enter the following command:

```
wsfdpln -s c -y c -o 0 -0
```

5. To remove plans that start at least 5 days before a remove process that begins on July 1, 2005 10:00, and to repeat the removal once every successive seventh day thereafter, enter the following command:

```
wsfdpln -D 2005-07-01 -T 10:00 -f -y s -o 5 -d 00/00/07
```

The process operates in the following way:

- On 2005-07-01 at 10:00, the plans started 2005-06-26 are removed
- On 2005-07-08 at 10:00, the plans started 2005-07-03 are removed
- On 2005-07-08 at 10:00, the plans started 2005-07-10 are removed

See Also

wdelpln

wstartapm

Starts the Activity Planner engine.

Syntax

wstartapm

Description

The **wstartapm** command starts the Activity Planner engine. Because this command is a bash script it must be run in the bash environment on supported Windows operating systems: precede the command with the string **sh**. This command should never be invoked directly, and should be used only for debugging purposes.

Options

None.

Authorization

APM_Admin

Return Values

The **wstartapm** command returns one of the following values:

0 Indicates that **wstartapm** started successfully.

Nonzero

Indicates that **wstartapm** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To start the Activity Planner engine on supported Windows operating systems, enter the following command:

```
sh wstartapm
```

See Also

“wstopapm” on page 121

wstopapm

Stops the Activity Planner engine.

Syntax

wstopapm [-f]

Description

The **wstopapm** command stops the Activity Planner engine abruptly. Because this command is a bash script it must be run in the bash environment on supported Windows operating systems: precede the command with the string **sh**. This command should never be invoked directly, and should be used only for debugging purposes. If the checkpoint restart mechanism is disabled, some information might be lost. The Activity Planner engine restarts automatically when a request, for example a report, is received.

Options

-f Interrupts any running operations and stops the Activity Planner engine. If this option is not specified, and the engine is busy, the command is ignored. To use this option, you must have *root* authority in addition to the required Tivoli roles.

Authorization

APM_Admin, root

Return Values

The **wstopapm** command returns one of the following values:

0 Indicates that **wstopapm** started successfully.

Nonzero

Indicates that **wstopapm** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

To stop the Activity Planner engine on supported Windows operating systems, enter the following command:

```
sh wstopapm
```

See Also

wstartapm

wsubpln

Submits an activity plan to the Activity Planner engine for execution.

Note: You can also submit an activity plan from the GUI. For information about this, see “Submitting an Activity Plan” on page 49.

Syntax

```
wsubpln [-w] [-t targets] -r actPlan_name [-D option_to_modify=value] [-V variable=value]
```

```
wsubpln [-w] [-T [@managedNode:]file_name] -r actPlan_name [-D option_to_modify=value] [-V variable=value]
```

```
wsubpln [-w] [-t targets] -f actPlan_def_file [-c encoding] [-i] [-o] [-D option_to_modify=value] [-V variable=value]
```

```
wsubpln [-w] [-T [@managedNode:]file_name] -f actPlan_def_file [-c encoding] [-i] [-o] [-D option_to_modify=value] [-V variable=value]
```

Description

The **wsubpln** command submits an activity plan to the Activity Planner engine for execution. The activity plan can be stored in the activity plan database or in an activity plan definition file.

If any of the subscribers specified in the activity plan are not valid, the operation fails on those subscribers, and continues on the other subscribers. Information about the subscribers on which the operation did not complete is written to the apmlog0 file in the Activity Planner trace files directory.

Options

- w** Specifies that the plan is submitted even if some incongruities have been detected. For example, if the target of an activity has been specified using a variable and the value of the variable has not been defined, the target cannot be resolved. Specify the **-w** option to continue with the submission of the plan or do not specify this option to stop the submission if such a problem is detected.
- t *targets*** Specifies a list of targets, with separating commas, that is assigned to the \$(TARGET_LIST) variable.
- T [*@managedNode:*] *file_name*** Specifies a file from which to load the list of targets to be assigned to the \$(TARGET_LIST) variable. The specified file can be stored on a generic managed node in the Tivoli management region. If no managed node is specified, the managed node from which the command is submitted is used. In this file, you can optionally precede the target name with the @ sign; list one target name per line or separate target names with commas.
- r *actPlan_name*** Specifies the name of the activity plan in the activity plan database that is to be submitted.
- f *actPlan_def_file_path*** Specifies the name of the activity plan definition file to be imported and submitted.

-c encoding

Specifies the encoding to be used for the activity plan definition file. For a complete list of the supported encoding types, see the Sun Microsystems Web site.

-i Specifies that the imported activity plan definition file is deleted from the activity plan database after it has been submitted.

-o Specifies that the activity plan definition file in the activity plan database is overwritten if a plan with the same name already exists.

-D option_to_modify=value

Specifies that some options in the plan are overridden before it is submitted. The options and their possible values are:

name

- string

priority

- low
- medium
- high

mail_address

- string
- - (disables mail notification)

post_notice

- y
- n

notify_date

- "yyyy-mm-dd hh:mm"
- hh:mm
- - (disables notify_date setting)

submit_paused

- y
- n

start_not_before

- "yyyy-mm-dd hh:mm"
- - (disables start_not_before setting)

complete_not_after

- "yyyy-mm-dd hh:mm"
- hh:mm
- - (disables complete_not_after setting)

stop_plan_on_error

- warning
- error
- fatal
- ignore (disables stop_plan_on_error setting)

cancel_at_cutoff

- y
- n

is_cancel_preferred

The possible values are:

- -y to set Cancel as preferred final status of a plan.
- -n not to set Cancel as preferred final status of a plan.

pre_eval_conditions

The possible values are:

- -true to evaluate all the conditioned activities of a plan as soon as the plan is submitted.

- -false not to evaluate all the conditioned activities of a plan as soon as the plan is submitted.

recursion_type

- di (date_interval)
- ti (time_interval)
- dom (days_of_month)
- dow (days_of_week)
- nr (no recursion)

You can modify the following options only if the **recursion_type** option is specified.

recursion_info

- if recursion_type=di, *yy/mm/dd*
- if recursion_type=ti, *hh:mm*
- if recursion_type=dom, *d1,d2,d3,...dn*
- if recursion_type=dow, *d1,d2,d3,...dn*

expiration_date

- *"yyyy-mm-dd hh:mm"*
- *hh:mm*
- - (disables expiration_date setting)

stop_on_overlap

- y
- n

stop_rec_on_error_level

- warning
- error
- fatal
- ignore (disables stop_rec_on_error_level setting)

target_resolution

- static
- dynamic

-V variable=value

Defines a user variable that is added to the existing variables defined in the plan.

Authorization

APM_Manage

Return Values

The **wsubpln** command returns one of the following values:

0 Indicates that **wsubpln** started successfully.

Nonzero

Indicates that **wsubpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To submit the plan Test, that is stored in the activity plan database, enter the following command:
`wsubpln -r Test`
2. To import the plan, test.xml, in the activity plan definition file format and overwrite the plan if it already exists and then submit it for execution, enter the following command:


```
wsubpln -f test.xml -o
```

3. To submit the plan Test stored in the activity plan database and overwrite the submit_paused and start_not_before options, enter the following command:

```
wsubpln -r Test -D submit_paused=y  
-D start_not_before="2003-10-23 18:00"
```

4. To import the plan test.xml, in the activity plan definition file format, and submit it with the addition of the variable var1, enter the following command:

```
wsubpln -f c:\test.xml -Vvar1=3
```

5. To submit the plan ActPlan1 and assign the variable \$(TARGET_LIST) the values tg1, tg2, enter the following command:

```
wsubpln -t tg1,tg2 -r ActPlan1
```

6. To submit the plan ActPlan1 and specify targets using a variable referencing a file /tmp/targets.txt containing the target names, enter the following command:

```
wsubpln -V targets=/tmp/targets.txt -r ActPlan1
```

7. To submit plan template PLANX, that is a recursive plan with mail_address=psmith@summers.it, and disable the recursion mechanism and mail notification so that the plan is submitted only once, with no notification, to psmith@summers.it, enter the following command:

```
wsubplan -r PLANX -D recursion_type=nr -D mail_address=-
```

8. To submit plan template PLANY without any time restrictions, disable the start_not_before and complete_not_after settings by entering the following command:

```
wsubpln -r PLANY -D start_not_before=- -D complete_not_after=-
```

9. To submit plan template PLANZ, that is a recursive plan and disable the expiration setting, expiration_date= 2003-01-31 13:00, enter the following command:

```
wsubplan -r PLANZ -D expiration_date=-
```

10. To submit plan template PLANW in paused state specifying a list of targets assigned to the \$(TARGET_LIST) variable and setting a new variable (test_variable) to define more targets, enter the following command:

```
wsubplan -r PLANW -t tg1, tg2, tg3 -D submit_paused=y  
-V test_variable=tg4,tg5,tg6
```

See Also

wimppln

wunlockpln

Displays a list of locked plans, and unlocks plans currently locked.

Syntax

wunlockpln -l

wunlockpln -i *plan_id*

wunlockpln -n *actPlan_name*

Description

The **wunlockpln** command displays a list of activity plans currently locked, and unlocks locked activity plans.

Options

- l** Displays a list of activity plans stored in the activity plan database that are currently in use and therefore locked. The plan name is displayed along with the user's name, the managed node name, and the date and time the plan was locked.
- i** *plan_id*
Unlocks an activity plan with the specified plan ID.
- n** *actPlan_name*
Unlocks the specified activity plan.

Authorization

APM_Admin

Return Values

The **wunlockpln** command returns one of the following values:

- 0** Indicates that **wunlockpln** started successfully.

Nonzero

Indicates that **wunlockpln** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To display a list of activity plans currently locked by users, enter the following command:

```
wunlockpln -l
```

The following is example output:

Name	User	Managed Node	Lock Date
-----	-----	-----	-----
Plan1	mars	eur1057	2003-03-25 15:54
Plan2	pluto	us22345	2003-02-30 10:15
Plan3	saturn	eur55567	2003-03-05 22:10

2. To unlock Plan0, enter the following command:

```
wunlockpln -n Plan0
```

See Also

None

wupdpn

Updates an activity plan that has been submitted to the Activity Planner engine for execution, but has not yet started.

Note: You can also delete an activity plan from the GUI. For information about this, see “Updating a Submitted Plan” on page 50.

Syntax

```
wupdpn -r actPlan_name [-t targets] [-D option_to_modify=value] [-V variable=value]
```

```
wupdpn -r actPlan_name [-T[@managedNode:]file_name] [-D option_to_modify=value] [-V variable=value]
```

Description

The **wupdpn** command updates an activity plan that has been submitted to the Activity Planner engine for execution, but has not yet started. The activity plan can be stored in the activity plan database or in an activity plan definition file.

Options

-r actPlan_name

Specifies the name of the submitted activity plan in the activity plan database that is to be updated.

-t targets

Specifies a list of targets, with separating commas, that are the values of the \$(TARGET_LIST) variable. You can modify the targets only if they have been resolved at activity execution. If the targets are resolved when the plan is submitted, you can no longer change them. For more information on target resolution, see “Selecting Targets for an Activity” on page 26.

-T [@managedNode:] file_name

Specifies a file from which to load the list of targets to be assigned to the \$(TARGET_LIST) variable. The specified file can be stored on a generic managed node in the Tivoli management region. If no managed node is specified, the managed node from which the command is submitted is used. You can modify the targets only if they have been resolved at activity execution. If the targets are resolved when the plan is submitted, you can no longer change them. For more information on target resolution, see “Selecting Targets for an Activity” on page 26. In this file, you can optionally precede the target name with the @ sign; list one target name per line or separate target names with commas.

-D option_to_modify=value

Specifies that some options in the plan are overridden before it starts. The options and their possible values are:

name

- string

priority

- low
- medium
- high

mail_address

- string
- - (disables mail notification)

post_notice

- y

- n
- notify_date**
 - "yyyy-mm-dd hh:mm"
 - hh:mm
 - - (disables notify_date setting)
- start_not_before**
 - "yyyy-mm-dd hh:mm"
 - - (disables start_not_before setting)
- complete_not_after**
 - "yyyy-mm-dd hh:mm"
 - hh:mm
 - - (disables complete_not_after setting)
- stop_plan_on_error**
 - warning
 - error
 - fatal
 - ignore (disables stop_plan_on_error setting)
- cancel_at_cutoff**
 - y
 - n
- recursion_type**
 - di (date_interval)
 - ti (time_interval)
 - dom (days_of_month)
 - dow (days_of_week)
 - nr (no recursion)

You can modify the following options only if the **recursion_type** option is specified.

- recursion_info**
 - if recursion_type=di, yy/mm/dd
 - if recursion_type=ti, hh:mm
 - if recursion_type=dom, d1,d2,d3,...dn
 - if recursion_type=dow, d1,d2,d3,...dn
- expiration_date**
 - "yyyy-mm-dd hh:mm"
 - hh:mm
 - - (disables expiration_date setting)
- stop_on_overlap**
 - y
 - n
- stop_rec_on_error_level**
 - warning
 - error
 - fatal
 - ignore (disables stop_rec_on_error_level setting)
- target_resolution**
 - static
 - dynamic

-V variable=value

Defines a user variable that is added to the existing variables defined in the plan.

Authorization

APM_Manage

Return Values

The **wupdpIn** command returns one of the following values:

0 Indicates that **wupdpIn** started successfully.

Nonzero

Indicates that **wupdpIn** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To update the submitted plan Test overwriting the stop_on_overlap and start_not_before options, enter the following command:

```
wupdpIn -r Test -D stop_on_overlap=y -D start_not_before="2003-10-23 18:00"
```
2. To update the plan ActPlan1 assigning the variable \$(TARGET_LIST) the values tg1, tg2, and adding the variable var1, enter the following command:

```
wupdpIn -r ActPlan1 -t tg1,tg2 -V var1=3
```
3. To update the plan ActPlan1 assigning the variable \$(TARGET_LIST) to the targets specified in the file /tmp/targets.txt, enter the following command:

```
wupdpIn -r ActPlan1 -T /tmp/targets.txt
```
4. To update plan template PLANX disabling the recursion mechanism and mail notification so that the plan is submitted only once, with no notification, enter the following command:

```
wupdpIn -r PLANX -D recursion_type=nr -D mail_address=-
```
5. To update plan template PLANY, removing all time restrictions, disable the start_not_before and complete_not_after settings by entering the following command:

```
wupdpIn -r PLANY -D start_not_before=- -D complete_not_after=-
```
6. To update plan template PLANZ, disabling the expiration setting, expiration_date= 2003-01-31 13:00, enter the following command:

```
wupdpIn -r PLANZ -D expiration_date=-
```

See Also

wsubpIn

Return Values

Return values help you identify the result of the command: a return value of 0 indicates that the command completed successfully, while a return value other than zero indicates that an error has occurred. A list of all return values other than zero is given in Table 33.

Table 33. Return values

Value	Description
1	A syntax error occurred while a command was being submitted.
2	The user submitting the command does not have the required permission.
3	An error occurred while Tivoli Management Framework objects were being accessed.
4	The timeout expired before the result of the requested operation was received
5	An unidentified error occurred while a command was being performed.
6	An error occurred while a file was being accessed.
7	The supplied password differs from the current one, or the confirmation password is different from the newly-entered password.
8	An error occurred during an attempt to access the database.
9	The specified plan does not exist.
10	The specified activity does not exist.
11	The specified target does not exist.
12	The value supplied for a parameter is not valid
13	The requested operation is not compatible with the current state of the plan.
14	An error occurred while a class was being loaded.
15	The specified plan is already in use.
16	The application is not registered.
17	The Plan Repository is empty.
18	An error occurred during an attempt to connect to the database.
19	A plan with the same name already exists.
20	The specified plug-in does not exist.
21	A plug-in with the same name already exists.
22	The specified syntax is not valid.

Chapter 4. Troubleshooting

This section gives an overview of the troubleshooting process and provides descriptions of sources of information that will help you do the following:

- Avoid problems
- Work around problems
- Gather information to analyze and solve problems.

Activity Planner Logs and Traces

You can use the logs and traces described in this chapter to define and troubleshoot problems with Activity Planner. Use the **wtrcapm** command to set the trace level for the Activity Planner engine, or choose **Settings » Trace Level** in the Activity Planner GUI to change the trace level for the GUI. The change is applied at runtime, so no restart of the Activity Planner is necessary.

The Activity Planner can write information to a variety of log and trace files. You can control, in most cases, whether the information is written, and where it is stored.

Note: If you do not have write access to the folder where the GUI traces are written, the trace information is written to the user's home directory.

Most of the files are controlled using keys in the `apm.ini` file, a description and example of which can be found in "The Activity Planner Configuration File" on page 8.

You can also specify the maximum number of trace files to be created by adding the **trace_files_num=3** parameter to the `apm.ini` file. You can set this parameter to any positive integer. If this parameter is not specified, the maximum number of trace files is 3.

You can also enable tracing for the Activity Planner RIM object by typing the following command:

```
wrimtrace RIM_object_label ERROR|INFORMATION
```

where

RIM_object_label

Specifies the RIM object you want to trace. The default name of the Activity Planner RIM object is `planner`. To obtain the current name of the RIM object, use "wapmrim" on page 97.

For more information on the **wrimtrace** command, refer to *Tivoli Management Framework: Reference Manual*.

The tracing function is intended for debugging purposes. If enabled for extended periods of time, tracing can decrease performance and slow the processing of the product considerably.

wtrcapm

Enables and displays the current trace and log setting for the Activity Planner engine, and defines new settings.

Syntax

wtrcapm -f

wtrcapm -v

wtrcapm -M-s *key=value*

wtrcapm -M -q *key*

wtrcapm -H-s *key=value*

wtrcapm -H -q *key*

wtrcapm -E -s *key=value*

wtrcapm -E -q *key*

wtrcapm -C -s *key=value*

wtrcapm -C -q *key*

wtrcapm -D -s *key=value*

wtrcapm -D -q *key*

wtrcapm -a -s *key=value*

wtrcapm -a -q *key*

Description

This command enables and displays the current trace and log settings and defines new values for the components of the Activity Planner engine. The Activity Planner engine comprises the following components: APMMain, APMHandler, APMExecuter, APMCli.

Trace files are stored in the `working_dir` path, which is defined in the `apm.ini` file; a trace is available for each engine component. The default trace level is 0, so no trace file is created unless the trace level is modified.

Options

- f** Dynamically updates the trace and log parameters to the values specified in the `apm.ini` file without restarting the Activity Planner engine.
- M** Specifies that the settings defined using the **-s** *key=value*, or **-v**, or **-q** options apply to the APMMain component. The default value is 0.
- H** Specifies that the settings defined using the **-s** *key=value*, or **-v**, or **-q** options apply to the APMHandler component. The default value is 0.
- E** Specifies that the settings defined using the **-s** *key=value*, or **-v**, or **-q** options apply to the APMExecuter component. The default value is 0.

- C Specifies that the settings defined using the **-s key=value**, or **-v**, or **-q** options apply to the APMCLI component. The default value is 0.
- D Specifies that the settings defined using the **-s key=value**, or **-v**, or **-q** options apply to the DEFAULT section of the apm.ini file.
- a Specifies that the settings defined using the **-s key=value** option apply to all sections of the apm.ini file.

-s key=value

Sets the value for the specified key in the specified component. Default keys are:

working_dir	Identifies the working directory where the main persistent data and traces are stored.
trace_level	Identifies the trace level. Possible values are: <ul style="list-style-type: none"> • 0 (none) • 1 (fatal) • 2 (error) • 3 (warning) • 4 (information) • 5 (verbose) <p>The default value is 0.</p>
trace_size	Identifies the size of the trace file. The default is 1 000 000 bytes.
log_max_file	Identifies the maximum size of the log file (only for the DEFAULT section of the apm.ini file). The default is 1 000 000 bytes.
log_level	Identifies the log level (only for the DEFAULT section of the apm.ini file). Possible values are: <ul style="list-style-type: none"> • 0 none • 1 (information) • 2 (warning) • 3 (error) <p>The default value is 0.</p>
log_file	Specifies the name of the log file that is created in the working_dir (only for the DEFAULT section of the apm.ini file). The default is apm.log.

-q key Displays the value for the specified key in the specified component.

-v Displays the value for the current trace and log settings.

Authorization

APM_Admin

Return Values

The **wtrcapm** command returns one of the following:

0 Indicates that **wtrcapm** started successfully.

Nonzero

Indicates that **wtrcapm** failed due to an error. See Table 33 on page 130 for more information about return values.

Examples

1. To view the current trace and log settings, enter the following command:
`wtrcapm -v`
2. To set the trace level to the value specified in the apm.ini file, enter the following command:
`wtrcapm -f`
3. To change the APMMain trace level to 5, enter the following command:
`wtrcapm -M -s trace_level=5`
4. To display the value of the working_dir key for the APMExecuter component, enter the following command:
`wtrcapm -E -q working_dir`

See Also**wtrccm**

Main Activity Planner Log

The main Activity Planner log is called **apmlogx**, where the x indicates a number between zero and one, and is controlled by the following keys in the [DEFAULT] section of the apm.ini file:

log_level	The log file is created only if this is set to greater than zero. The default value is 5.
log_file	Name of log file. The default name is apmlog .
working_dir	The directory in which the log file is to be created
log_max_file	Maximum number of bytes that can be written to the file. The default value is 1 000 000.

The log traces the history of all operations performed. If a problem occurs during the processing by the Activity Planner, a trace is written giving details of the problem that has occurred and what operation is responsible for the problem.

Each line in the file describes a single operation, and provides the following information:

- Date and time when the operation occurred
- Message Code
- Description of operation being performed

For every operation, you can determine which plans and activities are involved. This file also logs information on the settings contained in the apmJNI.ini file, which defines the memory allocation pool for the Java Virtual Machine. For more information on the apmJNI.ini file, see “The apmJNI.ini File” on page 14.

Activity Plan Monitor and Editor Startup Trace Files

The Activity Plan Monitor and Editor startup trace files are called **apmon.log** and **apmed.log**. These files can be found in one of the following locations:

Windows operating systems

%SystemDrive%

Unix operating systems

/tmp

These files record the interaction between the Tivoli Desktop and the Activity Plan Monitor and Editor GUIs, and are always written.

Activity Planner Core Trace

The Activity Planner Core trace files include the **apm.log** and **apm_core.log** files, and when they are activated they can be found in one of the following locations:

Windows operating systems

%SystemDrive%

Unix operating systems

/tmp

The trace files can be activated by adding the environment variable **_APM_DEBUG_**, using the **odadmin environ set** command, or as a system variable.

These files contain information relating to the Activity Planner core functionality, including calls made to the IDL interface and the Java Virtual Machine initialization and completion messages.

The `APM_RPC_MAX_THREADS` environment variable has been added to the `APM_core` process. This variable sets the maximum number of concurrent remote procedure call threads handled by the dispatcher. The default value is 250.

Activity Planner Main Trace

The main Activity Planner trace file is called **APMainxxx.trc**. It is controlled by the following keys in the [MAIN] section of the `apm.ini` file:

trace_level	The trace file is created only if this is set to greater than zero The default value is zero.
working_dir	The directory in which the log file is to be created
trace_size	Maximum number of bytes that can be written to the file The default value is 1 000 000.

It records the main thread traces.

Activity Planner Executer Trace

The Activity Planner executer trace file is called **APExecuterxxx.trc**. It is controlled by the following keys in the [EXECUTER] section of the `apm.ini` file:

trace_level	The trace file is created only if this is set to greater than zero The default value is zero.
working_dir	The directory in which the log file is to be created
trace_size	Maximum number of bytes that can be written to the file The default value is 1 000 000.

This trace file records traces associated with the executer thread that manages operations submitted by the Task Library and by Software Distribution plug-ins.

Activity Planner Handler Trace

The Activity Planner handler trace file is called **APHandlerxxx.trc**. It is controlled by the following keys in the *HANDLER* stanza of the `apm.ini` file:

trace_level	The trace file is created only if this is set to greater than zero. The default value is zero.
working_dir	The directory in which the log file is to be created
trace_size	Maximum number of bytes that can be written to the file. The default value is 1 000 000.

This trace file records traces associated with the Activity Planner handler thread.

Activity Planner Command Line Interface Trace

The Activity Planner Command Line Interface trace file is called **APClixxx.trc**. It is controlled by the following keys in the *APMCLI* stanza of the `apm.ini` file:

trace_level	The trace file is created only if this is set to greater than zero. The default value is zero.
working_dir	The directory in which the log file is to be created

trace_size Maximum number of bytes that can be written to the file. The default value is 1 000 000.

This trace file records traces associated with the use of the command line interface.

Activity Plan Monitor Trace

The Activity Plan Monitor trace file is called **APMonitorxxx.trc**. It is controlled by the following keys in the *MONITOR* stanza of the *apm.ini* file:

trace_level The trace file is created only if this is set to greater than zero. The default value is zero.

working_dir The directory in which the log file is to be created

trace_size Maximum number of bytes that can be written to the file. The default value is 1 000 000.

This trace file records traces associated with the use of the Activity Plan Monitor GUI.

Activity Planner Default Trace

The default Activity Planner trace file is called **APDefaultxxx.trc**. It is controlled by the following keys in the *DEFAULT* stanza of the *apm.ini* file:

trace_level The trace file is created only if this is set to greater than zero. The default value is zero.

working_dir The directory in which the log file is to be created

trace_size Maximum number of bytes that can be written to the file. The default value is 1 000 000.

This trace file records the trace messages relating to threads not tracked in other files.

Downloading Plug-ins

After plug-ins have been installed and registered on the server, they can be automatically downloaded to the machines where the Activity Planner GUIs are installed.

Information about the installed plug-ins is maintained locally on each machine, and every time the Activity Planner GUIs start, a check is performed on the server to verify whether a new plug-in has been installed, or if the currently installed plug-ins have been updated or removed. If a change is detected, the new files are sent to the Activity Planner and local information is updated.

You can disable or enable the download mechanism by defining the value of the *plugin_download* key in the appropriate section of the *apm.ini* file, as follows:

- On the server side, in the *DEFAULT* section
- On the Activity Planner GUI side, in the *APMEDITOR* and *MONITOR* sections

By default, the *plugin_download* key is enabled.

If the mechanism is disabled on the Activity Planner GUI side, local information is used to load the plug-ins. If the mechanism is disabled on the server side and an

Activity Planner GUI requests the plug-ins from the server, the requested files are not sent and the application is started using local plug-in information.

Disabling the download mechanism reduces network traffic and improves the application loading time.

Any installations performed after the plug-in registration, for example patch or language pack installations, and modifications affecting the plug-in files are not effective unless the plug-in information is updated. You can update the plug-in information in one of the following ways:

- For example, run the following command in any shell:

```
$BINDIR/TME/APM/SCRIPTS/reg_swd_plugin.sh -r
```

This command updates the plug-in information for the Software Distribution plug-ins. To update the information for the other plug-ins, use the appropriate script. For more information on registration and update scripts, refer to *Planning and Installation Guide*.

- Insert the following key and related value in the apm.ini file in the DEFAULT section:

```
automatic_plugin_refresh=enabled
```

This key causes the information for every registered plug-in to be updated each time the Activity Planner engine starts. This operation can increase the application loading time, so you should remove this key from the apm.ini file after the update is completed.

Managing DB2 Deadlock

The problem described below was encountered with the DB2 database. While writing information to the database tables, an operation can lock the database service and prevent the system from responding to a command until the end of the transaction.

To avoid this problem, create a new RIM object by performing the following steps:

1. Create an alias for the planner database (for example *planner_mon*)
2. Configure the database alias adding a section to the db2cli.ini file (in the DB2 client directories) containing the TXNISOLATION=1 key. The name of the section should be the same name of the new alias created. For example, if the alias is named *planner_mon*, you should add the following lines to the db2cli.ini file:
 - [planner_mon]
 - TXNISOLATION=1
3. Create a new RIM object having the same name as the Activity Planner RIM object (for example *planner*) with the suffix *_mon*. To create a RIM object, use the **wcrtrim** command, and to obtain the name of the RIM, use the **wapmrim** command. For more information on these commands, see “wapmrim” on page 97.

Note: This naming convention is mandatory. If a user changes the name of the original RIM object, also the name of the second RIM must be changed to match it.

Managing Different Domains

If the Activity Planner is on a managed node that is installed on a network domain different from the Tivoli management region domain, you can start the Activity Planner in one of the following ways:

- Use the `wapmgui` command.
- On the Tivoli management region, specify the domain where the managed node resides. To specify a new domain on a UNIX Tivoli management region, add an entry to the `/etc/resolv.conf` file. To specify a new domain on a Windows Tivoli Management region, add the required DNS to the DNS list for the TCP/IP settings.

Activity Planner Installation Notes

During the installation of the Activity Planner server component, a new user called *tivapm* is created on the operating system, if one does not already exist. This user does not have any particular privileges, but is necessary to create a Tivoli Administrator. As a consequence, on the Tivoli desktop, a new Administrator is created called `swd_admin_region-name-region` that is associated with this user. The creation of such a user and of the new Administrator is necessary for the Activity Planner engine. When the engine performs operations, it must authenticate itself with the Tivoli Management Framework, and it requires this Administrator to do so.

The management of the password of this new user is very important. The Activity Planner engine maintains an encrypted copy of this password internally. The password maintained by the engine must always be synchronized with the password of the operating system user. If the password of the user is changed on the operating system, the password maintained by the engine must be changed accordingly using the **wsetapmpw** command. For more information, see “wsetapmpw” on page 114

If the administrator chooses a username that already exists on the operating system during the installation of the Activity Planner server component, a new user is not created. However, if a password different from the one used by the operating system is used during the installation, the Activity Planner will not work. You must use the **wsetapmpw** command to synchronize the password maintained by the engine with that on the operating system.

After installing the Activity Planner, verify whether the engine functions properly by entering, for example, the following command:

```
wlstpln
```

which returns a list of the plans available in the Activity Planner database, if any.

In case a problem has occurred, an error message is displayed or written to a log file. If the problem is related to the password for user *tivapm*, verify the following:

- User *tivapm* has been created on the system.
- The correct password has been set for user *tivapm*.
- Administrator `swd_admin_region-name-region` has been created with a login for user *tivapm*.

Activity Planner by default uses a Socket connection through port number 7070. If this port is used by another application, modify the port number for Activity Planner by defining in the environment the *APM_PORT* variable, as explained in the following steps:

1. Export the Tivoli environment variables to file env.txt by entering the following command:

```
odadmin environ get > env.txt
```
2. In file env.txt add variable *APM_PORT=port_number* where *port_number* is the new port number to be defined using the following command:

```
echo APM_PORT=port_number >> env.txt
```
3. Import file env.txt by entering the following command:

```
odadmin environ set < env.txt
```

Managing Linux Versions

Depending on the version of Linux you have installed, the Activity Planner engine might not start. In this case, you must define the following two variables:

APM_KERNEL_LINUX

The correct value for this variable can vary, depending on the distribution you have installed. The value 2.2.5 should work for most distributions. If you are using United Linux with Kernel Version 2.4.19, add the following variable to the Tivoli environment using the **odadmin environ set** command:

```
APM_KERNEL_LINUX=2.4.19
```

The procedure for defining environment variables is described in “Activity Planner Installation Notes” on page 139.

Using X sessions

X clients used to connect from Windows to UNIX machines might be very slow. In this case, users should install the Tivoli desktop and the Deployment Services GUIs on the local machine, then connect to the remote machine.

Checkpoint restart mechanism

The checkpoint restart mechanism maintains information about internal activities performed by Activity Planner. If Activity Planner activities are interrupted for any reason (for example, a system power off, or loss of connection to the database), at program restart the activities are automatically restarted from the last checkpoint. This mechanism is enabled by default and is controlled by the **checkpoint_restart** parameter in the apm.ini file. For more information, see “Defining the Activity Planner Engine Parameters” on page 9.

Activity Planner can recover from unexpected failures, such as power-offs or Java Virtual Machine failures, and manage transient problems such as some database errors or network interruptions. The distribution is automatically resumed when the cause of failure has been corrected.

Activity Planner handles the error conditions listed above in two ways:

- By setting suitable checkpoints
- By retrying operations according to the type of failure

When an error occurs, for example if a database returns an exception because a deadlock timeout elapsed, Activity Planner waits for a defined time interval and then retries the transaction.

Use the **db_retry_count** and **db_retry_interval** parameters in the APM_TUNING section of the apm.ini file to control this feature. For more information on the apm.ini file and the parameters it contains, see “The Activity Planner Configuration File” on page 8 and “Defining the Activity Planner Engine Parameters” on page 9.

The **db_retry_count** parameter specifies the maximum number of times Activity Planner tries to reconnect to the database after a failed attempt.

The **db_retry_interval** parameter specifies the interval in seconds between each connection attempt to the database. The interval is automatically increased each time a new attempt is performed based on the number of the attempt by the specified retry interval. For example, if the connection attempt fails and the **db_retry_interval** parameter is set to 120 seconds, the first retry is performed after 120 seconds, the second retry after 240 seconds, the third retry after 360 and so on, until either the connection is established or the value defined for the **db_retry_count** parameter is reached and no other retries are performed. After the last unsuccessful retry, the operation is canceled and a message is written to the log file. In this case, the status of the plan involved in the operation might be incorrect.

When Activity Planner restarts after a failure of the Java Virtual Machine, Activity Planner checks the last saved valid checkpoint and continues its operations from that starting point.

Two different recovery strategies take place depending on the type of activities Activity Planner was performing when it stopped:

- If the operation being performed involves only database access, the operation is fully recovered and Activity Planner continues normally.
- If the operation being performed involves external calls to a plug-in, for example if it was submitting an operation to Software Distribution or if it was registering a distribution to MDist 2, the status of the targets involved in the interrupted operation is set to **failed**. In this case, the status of an activity in the Activity Planner might not match the status of the corresponding operation at the MDist 2 level. The activity might be marked as **failed** in the Activity Planner Monitor GUI, while the **wmdist -l** command might indicate that a distribution has started for that activity. To minimize the possibility of data corruption, Activity Planner sets the activity status to **failed** and writes a message to the log file. In this case, you have to perform any recovery steps manually.

The following are two typical scenarios that might occur during the submission of an activity:

Activity Planner was submitting a distribution for an entire activity

If the Activity Planner stops unexpectedly, when the program restarts all the targets are set to **FAILED**, while a distribution might have started at the MDist 2 level. In this case, the Activity Planner has not received the distribution ID for the operation and cannot verify its status. The submitted distribution might be performed unless you issue a cancel command. Issue the **wmdist -l** command to check the activity status and perform the necessary recovery steps, if any.

Activity Planner was registering a distribution to MDist 2

If the Activity Planner stops unexpectedly, when the program restarts all the targets are set to **failed**, while the distribution might have been registered at the MDist 2 level. In this case the distribution might remain in **waiting** status, unless you delete it. Issue the **wmdist -l** command to check the activity status and perform the necessary recovery steps, if any.

Specific Problems and Workarounds

The following list details some common problems and workarounds for Activity Planner:

The Activity Planner GUIs do not start when launched from the Tivoli desktop

Check whether the **odadmin odlist** command returns the fully qualified hostname. If the fully qualified hostname is not returned, add it to the aliases list using the **odadmin add_hostname_alias** command. For more information on the **odadmin odlist** command, refer to *Tivoli Management Framework: Reference Manual*.

Error messages are truncated

If error messages are longer than 250 characters, they are truncated. To solve this problem, add the

`max_error_info_size`

keyword in the DEFAULT section of the `apm.ini` file and enlarge the `ERROR_INFO` column in the `ACT_STATUS_TGT` table to the same value defined for the

`max_error_info_size`

keyword. The maximum size for this column depends on the database you are using.

Values set for the LD_LIBRARY_PATH environment variable are lost

If you are using a Sybase database in your Tivoli environment and the manual configuration of the environment is performed before installing the Activity Planner component, then the same configuration should be repeated after the Activity Planner installation because the settings of the `LD_LIBRARY_PATH` environment variable are overwritten by the Activity Planner installation.

JCF paths for the APMCLASSPATH environment variable are wrong

If you migrate from older releases to Tivoli Configuration Manager 4.2.3 release, the `odadmin` environment variable `APMCLASSPATH` wrongly points to the following paths:

- 4.1 JRIM.jar
- 4.1 JCF.jar
- 4.1 ibmjsse.jar
- 4.1 jsafe.zip

even if you have migrated to the correct Tivoli Management Framework level.

Manually correct the paths by modifying them as follows:

- 4.1.1 JRIM.jar
- 4.1.1 JCF.jar
- 4.1.1 ibmjsse.jar

- 4.1.1 jsafe.zip

Note: On AIX platforms, do not use the VI editor to modify these paths.

Part 2. Modeling the Enterprise Configuration

Chapter 5. Using Change Manager	147
Before You Start	147
Change Manager Concepts	148
Reference Model Overview.	148
Configuration Element Overview.	148
Subscribers	149
Subscriber Types	149
Differencing Mechanism.	151
Understanding the Change Manager	
Environment	151
Change Manager Roles	152
Enabling Configuration Manager Components for	
Change Manager	153
The Change Manager Configuration File	
confccm.xml.	153
The settings.xml file	155
The stable.xml File	155
The invtable.xml File	160
The ostable.xml File	160
Reference Models	161
Reference Model Example	161
Representing a Reference Model in XML Format	163
 Chapter 6. Performing Change Manager	
Operations	171
Launching the Change Manager GUI	171
Using Change Manager	171
Creating the Reference Model Structure	172
Creating a Reference Model from a Target.	173
Importing a Reference Model	175
Exporting a Reference Model in XML Format	175
Adding Elements to a Reference Model.	176
Adding an Inventory Data Element	176
Adding an Inventory Scan Element	177
Adding an Operating System Element	177
Adding a Software Distribution Element	178
Assigning Subscribers to a Reference Model	180
Assigning an Inventory Subscriber	181
Assigning a Profile Manager Subscriber	182
Assigning a CSV Subscriber	182
Assigning a Static Subscriber	183
Assigning Query Library Subscribers	183
Assigning Directory Query Library	
Subscribers	184
Assigning Pristine Manager Subscribers	185
Modifying a Reference Model	186
Change Manager and Activity Plans.	188
Previewing an Activity Plan	189
Assigning a Priority to Configuration Elements	189
Submitting an Activity Plan	191
Creating Change Manager Reports	192
Reference Model Report.	193
Target Report	194
 Chapter 7. Using the Command Line.	195
Managing Reference Models	195
wccmgui	197
wccmplugin.	198
wdelrmod	200
wexprmod	202
wimprmod	204
wlstrmod.	206
wlstsrep	208
wsyncrmod	210
 Chapter 8. Troubleshooting	213
Change Manager Logs	213
Change Manager Traces	213
Checking the Change Manager Configuration File	215
Change Manager GUI Trace	216
wtrccm	217
Managing Different Domains	219
Distributing to a Large Number of Targets	219
Specific Problems and Workarounds.	219

This part describes how you use the Change Manager component of IBM Tivoli Configuration Manager (Change Manager). Change Manager works with Activity Planner to manage specified groups of users, workstations, or devices as single subscriber entities. Change Manager allows users to create reference models that define hardware and software requirements for these subscribers. After a reference model is created, Change Manager can generate an activity plan that includes all the tasks required to ensure that the subscribers match the requirements defined in the reference model. These tasks can include software distribution and inventory operations, if the corresponding plug-ins are installed. The activity plan can then be submitted to Activity Plan Manager for implementation. This part also describes how you use Change Manager with Software Distribution and Inventory.

Locating Related Information

Information related to this service is available from the following sources:

- *Introducing IBM Tivoli Configuration Manager*. This provides an overview of the service.
- *Planning and Installation Guide*. This includes information about how you install the service.
- *Messages and Codes*. This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 5. Using Change Manager

The Change Manager component of IBM Tivoli Configuration Manager, together with the Activity Planner, supports software distribution management and change management in a large network. Activity Plan Manager is a prerequisite of Change Manager. Change Manager works with Activity Plan Manager to manage specified groups of users, workstations, or devices as single subscriber entities. Subscribers can be users, groups of users, or the workstations or devices they use.

The core concept of Change Manager is the reference model. A reference model contains an association of configuration elements and subscribers. Configuration elements define hardware and software requirements. Subscribers define groups of users, workstations, or pervasive devices.

Using Change Manager, you create reference models that use configuration elements to specify a required configuration on a given set of target subscribers. These reference models allow you to define the hardware and software requirements of different groups of users, for example, managers and software developers. This allows you to manage subscribers according to the role each plays within your organization.

After you have created the reference model, you can generate an activity plan that includes all the tasks required to ensure that the subscribers match the requirements you defined in the reference model. You can then submit this activity plan to Activity Plan Manager.

If there is a change to requirements, or if a subscriber changes its role, you can simply update the reference model to reflect the changes and generate a new activity plan.

This chapter provides the following:

- Explanations of the concepts involved in Change Manager. See “Reference Models” on page 161 and “Subscribers” on page 149.
- A description of an example reference model. See “Reference Model Example” on page 161.
- Instructions on how to work with the Change Manager interface. See “Using Change Manager” on page 171.

Before You Start

This section contains introductory information you must understand before you start using Change Manager. It contains the following sections:

- “Understanding the Change Manager Environment” on page 151
- “Change Manager Roles” on page 152

Before you start using Change Manager, you must perform the following operations:

- Ensure that all installation and configuration tasks have been completed.
Ensure that the Activity Planner is correctly installed, the prerequisites for using the Java GUI must have been installed on the system you are using, and the

required RDBMS configurations must have been made. See *Planning and Installation Guide* and *Database Schema Reference*.

- Assign a Change Manager role to the user ID with which you are working. The user ID with which you are working must have at least the following roles:
 - Framework user role.
 - RIM_view and RIM_update roles.
 - Appropriate Change Manager role for operation, as described in “Change Manager Roles” on page 152.
 - Appropriate Activity Plan Manager roles for submitting a plan during the synchronization process. For information about synchronization and submitting an activity plan, see “Change Manager and Activity Plans” on page 188.

There are four Change Manager roles; CCM_View, CCM_Manage, CCM_Edit, and CCM_Admin. To complete all the tasks described in the sections that follow, you must assign at least the APM_Edit role. For more information about Change Manager roles see the “Change Manager Roles” on page 152.

Change Manager Concepts

This section introduces the following Change Manager concepts:

- Reference models.
- Configuration elements.
- Subscribers.

Reference Model Overview

The Change Manager reference model structure represents the relationships between the software and hardware requirements of different categories of user in your organization. The reference model is made up of component models organized in a hierarchical structure. The root level defines requirements that are common to all users and the child models define additional specific requirements that apply only to a particular group of users.

Configuration Element Overview

Configuration elements are associated with each reference model and use the concept of desired state to define the hardware and software requirements of subscribers to the reference model. For each registered plug-in, a configuration element is uniquely and completely identified by the name/desired state pair. The configuration elements available depend on the set of plug-ins you have registered. When you want to add a configuration element to a reference model, you must specify the element's name and the desired state.

The only exception to this is when you want to add an InventoryData element. InventoryData elements do not require you to specify a desired state because this element only checks for information and has no desired state to reach.

Configuration elements defined for models at the top of the hierarchy are inherited by those lower in the hierarchy. The types of configuration elements are listed below:

Software Distribution Elements

A Software Distribution (SWD) element represents the software distribution element as defined in the Tivoli Software Distribution environment. Change Manager retrieves the software package current status from the Tivoli configuration database and produces the actions necessary to reach the desired state.

To define conditions between configuration elements, you can assign a priority to the elements contained in the reference model. For more information, see “Assigning a Priority to Configuration Elements” on page 189.

An SWD element can also be made conditional on a software dependency. A software dependency is defined as one of the following:

- **Prerequisite**, where the changes required by the element are only made if the dependency condition is true.
- **Exerequisite**, where the changes required by the element are not made if the dependency condition is true.
- **Corequisite**, where the changes required by the element can only be made as part of a sequence that includes the requirement defined in the dependency condition. For an example of a corequisite, see “Reference Model Example” on page 161. In this example, the Winmerge software package is a corequisite of the CMVC software package and is installed only if the CMVC software package is already installed.

Inventory Data Elements

An inventory data element verifies the reference model against the Inventory database, for example for hardware requirements or set of requirements. To create an Inventory element, you define an expression that includes, for example, one or more hardware characteristics, such as physical memory size, and software characteristics, such as installed software. You select the characteristics you want to use from one of the available logical views of the Tivoli configuration database.

Inventory Scan Elements

An inventory scan element allows you to perform software and hardware scans of subscriber machines. To create an inventory scan element, you define a profile that includes one or more scan characteristics.

Operating System Element

An operating system element enables you to do a pristine installation using the Pristine Manager. The element represents the operating system to be installed on the machines that are part of the group subscriber. In addition, it includes properties, such as the operating system type, architecture, and version.

Subscribers

The subscribers to a reference model are the workstations, devices, and users that you want to be configured to match the hardware and software requirements defined for the model. For information about assigning subscribers to a reference model, see “Assigning Subscribers to a Reference Model” on page 180.

Subscriber Types: Change Manager supports the following types of subscribers:

Static subscriber

This type of subscriber contains a static list of one or more targets. It is the only subscriber where the targets are identified at the time when the subscriber is defined rather than at synchronization time, when the actions required for the model are generated.

CSV subscriber

For this type of subscriber, you define the fully qualified path to a CSV file. CSV files are a simple form of spreadsheet representation. They are plain text files that can be read or modified with any text editor, as well as spreadsheets. All data on the same line in a CSV file appears in the same row in its

spreadsheet form. Each comma on a line indicates the separation between cells. For example, the following line in a CSV file is interpreted by Change Manager as a list of three targets: Lisbon, Rome, Paris. At synchronization time, the file is read and the targets it includes are imported.

Profile Manager subscriber

For this type of subscriber, you specify a profile manager that exists in the Tivoli object database. When Change Manager creates an activity plan, it generates a query to include all the current subscribers (targets) associated with the profile manager. For more information about Profile Manager subscribers, see the *Tivoli Management Framework User's Guide*.

Inventory subscriber

For this type of subscriber, you select a query from the Tivoli configuration database. The query selects a list of targets that match its criteria. For more information about Inventory subscribers, see the *Tivoli Inventory User's Guide*.

Query Library subscriber

For this type of subscriber, you select a query from the Tivoli object database. When performing the query, information is retrieved from a set of database tables, or more specifically, from a set of fields. For more information about Query Library subscribers, see the *Tivoli Inventory User's Guide*.

Directory Query Library subscriber

For this type of subscriber, you select an enterprise directory from the Tivoli object database. When performing the query, information is retrieved from the Enterprise Directory server. For more information about Directory Query subscribers, see Part 6, “Managing an Enterprise Directory,” on page 415.

Group subscriber

This subscriber type enables you to subscribe a group of pristine machines to a reference model that installs an operating system.

Pristine Target subscriber

This subscriber type enables you to subscribe individual pristine machines to a reference model that installs an operating system.

In addition, if you need backward compatibility with Web User Interface 4.1 you have access to the Web Interface subscriber described below:

Web Interface subscriber

A subscriber accessed by the Web Interface.

Note: This subscriber type is provided only for backward compatibility with Change Configuration Manager 4.1 and should only be used with Web User Interface 4.1.

You use this subscriber to create any of the other five subscriber types. The main difference between a subscriber created as a Web Interface subscriber and any of the other subscriber types, is that no actions are created for a Web Interface subscriber in the activity plan. Instead, the model is processed using the Web Interface module, where a set of actions is prepared and submitted directly to Software Distribution.

To publish a reference model to web users in later versions of Configuration Manager, use the `wweb` command. For more information on the `wweb` command, see “`wweb`” on page 288.

Using Web Interface, Version 4.2: When using Web Interface 4.2, the Web subscriber is disabled when you use the **Add** drop down menu to add subscribers as described in “Assigning Subscribers to a Reference Model” on page 180.

Differencing Mechanism

Change Manager creates required actions by using a differencing mechanism that compares the current state of each element on the specified subscribers with the desired states defined in the reference model. The Change Manager differencing mechanism produces the actions necessary to arrive at the desired state for each element on each target assigned to it, in the form of an activity plan. The activity plan contains a list of actions necessary to attain the desired state and is submitted to the Activity Planner or imported to the Activity Planner database for scheduling. For information about using the Change Manager GUI to submit and schedule the activity plan, see “Submitting an Activity Plan” on page 191

When you have registered the Inventory Data plug-in, Change Manager checks the requirements specified in inventory data elements. If a target subscriber does not meet these requirements, none of the changes defined by the activity plan can be made for that subscriber and an error message is produced listing all the failures detected during the check phase.

Understanding the Change Manager Environment

When you have completed the installation and configuration of Change Manager, as described in the *Planning and Installation Guide* manual, the following Change Manager components are present:

- Change Manager Java GUI on the managed nodes where the Java GUI component has been installed.
- The stable.xml file, when you have registered the Software Distribution plug-in. This file is used by the differencing mechanism to determine the actions required to bring a software distribution element to the desired state. For more information, see “The stable.xml File” on page 155.
- The invtable.xml file, when you have registered the InventoryScan plug-in. This file is used by Change Manager when the differencing mechanism is applied to Inventory configuration elements. For more information, see “The invtable.xml File” on page 160.
- The ostable.xml file when you have registered the Pristine Manager plug-in. The ostable.xml file is the basis on which Change Manager can determine the actions to be taken to bring a subscriber to the Installed state defined in the operating system element. For more information, see “The ostable.xml File” on page 160.

The first time Change Manager is run, the confccm.xml file is created. This file defines the configurable components of Change Manager, for example:

- Persistence information (the RIM name)
- Trace information
- Log information
- Plug-in information

For more information, see “The Change Manager Configuration File confccm.xml” on page 153.

In addition, Change Manager uses the following components:

- The Inventory database, from which Change Manager obtains software package information, for example name, version and current state, and information about subscribers’ hardware configurations, for example, hard disk size and CPU model.
- The Tivoli database, from which Change Manager obtains information about software package objects and Tivoli endpoints.

Change Manager Concepts

- Activity Planner, which is the software component that schedules and executes the activity plan generated by Change Manager to upgrade subscribers to the desired state defined in a reference model.

Change Manager Roles

The following table lists the role required to perform Change Manager operations:

Note: In order to use Change Manager, Tivoli Framework role "user" is required. In addition, for activities that access or modify information in the Change Manager database, you must have at least the following roles:

- user
- rim_view
- rim_update
- ccm_view

Operation	Context	Required Role
Load a reference model	Reference model	CCM_View
Import a reference model	Reference model	
Export a reference model	Reference model	
View the component properties of a reference model	Reference model	
Search for a reference model	Reference model	
Lock the children of a reference model	Reference model	
Hide the children of a reference model	Reference model	
Arrange the child reference models horizontally or vertically	Reference model	
Preview the actions	Reference model	
Access help	Change Manager GUI	
Access product information	Change Manager GUI	
Save the changes to reference models	reference model	CCM_Manage
Restore the reference models from the database	reference model	
Add a new element, subscriber, or dependency	reference model	
Modify a new element, subscriber, or dependency	reference model	
Remove an element, subscriber, or dependency	reference model	
Add a new reference model	reference model	
Remove a reference model	reference model	
All operations with the CCM_Manage role	reference model	CCM_Edit
Submit an activity plan	activity plan	

Operation	Context	Required Role
All operations with the CCM_Edit role	reference model	CCM_Admin
Register and unregister plug-ins	Change Manager plug-in registration	

Enabling Configuration Manager Components for Change Manager

You can use the Change Manager in conjunction with the other Configuration Manager components, Inventory, Software Distribution and Pristine Manager, by registering the related plug-ins as described in “wccmplugin” on page 198. For more information on registering plug-ins, refer to *Planning and Installation Guide*.

The Change Manager Configuration File confccm.xml

The confccm.xml file defines the configurable components that are handled by Change Manager. These include components of the reference model, for example:

- Persistence information (the RIM name)
- Trace information
- Log information
- Plug-in information

Configurable components also define security levels, tracing, and persistence of data. You can edit this file, though this should be done with caution.

Note: When upgrading to Change Manager 4.2 and Change Manager Version 4.3.1 from Change Configuration Manager 4.1, Change Manager creates new configuration files the first time it is launched after the upgrade. The 4.1 configuration files, confccm.xml and confccm.dtd are backed up and renamed confccm.xml.bk and confccm.dtd.bk.

The following is the data type definition for the confccm.xml file.

```
<!ELEMENT ccm_configuration (persistence_data?,trace_data?,log_data?,plugin_
download?,plugin*)>
<!ELEMENT persistence_data (class,user?,password?,url?,driver?,custom?,
panel?,mode?)>
<!ELEMENT trace_data (trace_level?,trace_size?,working_dir?)>
<!ELEMENT log_data (log_level?,logfile_size?,logfile_name?,log_working_dir?)>
<!ELEMENT plugin_download (client?, server?)>
<!ELEMENT plugin (name,classpath,description?,helpset?,package,file_crc*)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT classpath (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT helpset (#PCDATA)>
<!ELEMENT package (#PCDATA)>
<!ELEMENT file_crc (#PCDATA)>
<!ELEMENT class (#PCDATA)>
<!ELEMENT panel (#PCDATA)>
<!ELEMENT custom (#PCDATA)>
<!ELEMENT driver (#PCDATA)>
<!ELEMENT password (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ELEMENT user (#PCDATA)>
<!ELEMENT mode (#PCDATA)>
<!ELEMENT client (#PCDATA)>
<!ELEMENT server (#PCDATA)>
<!ELEMENT trace_size (#PCDATA)>
```

The Change Manager Configuration File

```
<!ELEMENT trace_level (#PCDATA)>
<!ELEMENT working_dir (#PCDATA)>
<!ELEMENT logfile_name (#PCDATA)>
<!ELEMENT logfile_size (#PCDATA)>
<!ELEMENT log_level (#PCDATA)>
<!ELEMENT log_working_dir (#PCDATA)>
```

The first statement in the data definition:

```
<!ELEMENT ccm_configuration (persistence_data?,trace_data?,log_data?,plugin_
download?,plugin*)>
```

identifies the sections of the file and indicates whether they support multiple iterations and whether or not they are required.

Table 34 summarizes the information contained in this statement. The values that can be defined for each of these sections are summarized in subsequent tables.

Table 34. Sections of the confccm.xml file

Section	Required	Multiple Iterations
persistence_data	No	No
trace_data	No	No
log_info	No	No
plugin_download	No	No
plugin	No	Yes

The following is the default definition for the confccm.xml file.

```
<?xml version="1.0"?>
<!DOCTYPE ccm_configuration SYSTEM "confccm.dtd">
<!-- Revision: 61 1.3 confccm.xml, docs, xml4j2, xml4j2_0_0 -->
<ccm_configuration>
  <persistence_data>
    <class>com.tivoli.ccm.data.JRIMManager</class>
    <url>ccm</url>
  </persistence_data>
  <trace_data>
    <trace_level>0</trace_level>
    <trace_size>1000000</trace_size>
    <working_dir>C:\Tivoli\bin\ccm</working_dir>
  </trace_data>
  <log_data>
    <log_level>5</log_level>
    <logfile_size>1000000</logfile_size>
    <logfile_name>ccmlog</logfile_name>
  </log_data>
  <log_working_dir>C:\Tivoli\bin\w32-ix86\..\ccm</log_working_dir>
  </log_data>
  <plugin_download>
    <client>enabled</client>
    <server>enabled</server>
  </plugin_download>
</ccm_configuration>
```

Once the plug-ins are registered on the server side, they are not directly available on the client side, that is, for the GUI. In order for the GUI to use those plug-ins, the downloading mechanism has to take place: with this mechanism the plug-in information is registered also inside the confccm.xml file inserting as many plug-in sections as the number of registered plug-ins. The following is an example of the

contents added to the configuration file after you registered the Software Distribution, Inventory data, Inventory scan, and Pristine Manager plug-ins:

```
<plugin>
  <name>SoftwareDistribution</name>
  <classpath/>
  <description>This is the plug-in supplied by software distribution</description>
  <package>com.tivoli.ccm.plugin.sd</package>
  <file_crc>trm.jar;767397066</file_crc>
  <file_crc>swd_hlp.jar;1611101646</file_crc>
  <file_crc>swd_plugin.jar;1905299872</file_crc>
  <file_crc>swd_stubs.jar;3412524716</file_crc>
</plugin>
<plugin>
  <name>InventoryData</name>
  <classpath/>
  <description>This is the plug-in supplied by inventory - hardware
conditioning</description>
  <package>com.tivoli.ccm.plugin.inv</package>
  <file_crc>inv_data_hlp.jar;3664680824</file_crc>
  <file_crc>inv_plugin.jar;1481619963</file_crc>
</plugin>
<plugin>
  <name>InventoryScan</name>
  <classpath>$(BINDIR)/../generic/inv/jars/inv.jar$(SEP)$(BINDIR)
/TME/CCM/GUI/invscan.jar</classpath>
  <description>This is the plug-in supplied by Inventory</description>
  <package>com.tivoli.ccm.plugin.invscan</package>
</plugin>
<plugin>
  <name>OSElement</name>
  <classpath></classpath>
  <description>Operating System element (used by Pristine Manager)</description>
  <package>com.tivoli.cm4os.oselement</package>
  <file_crc>pristine_gui.jar;3015210905</file_crc>
  <file_crc>pristine_hlp.jar;3454707234</file_crc>
  <file_crc>uil.jar;1075337450</file_crc>
  <file_crc>pristine_ccm_hlp.jar;3030345584</file_crc>
  <file_crc>pristine_ccm_plugin.jar;1686330041</file_crc>
  <file_crc>pristine_stubs.jar;1846172985</file_crc>
  <file_crc>ostable.dtd;3490018062</file_crc>
  <file_crc>ostable.xml;2016769494</file_crc>
</plugin>
```

The settings.xml file

The settings.xml file stores information about the default file path used in file browser dialog boxes. You can customize this information in the graphical interface using the **Settings » Update Default Path** menu item. All file browser dialog boxes will start in the path you specify. This file also stores the information displayed in the login panel, that is the host name of the server and the user name entered at last login.

The stable.xml File

If you have installed the Software Distribution Plug-in, the stable.xml file is the basis on which Change Manager can determine the actions to be taken to bring a subscriber to the desired state defined in SWD element. The file contains definitions for every supported combination of current state and desired state. Each entry defines the actions to be taken to change from the current state to the desired state. For example, if current state is the initial state "-----" and the desired state is installed in undoable mode "ICU--", the required operation is "Install Undoable".

The stable.xml File

The use of this file to define actions required for each current state/desired state pair, allows the introduction of new software package states in later releases. It also allows you to define additional actions. For example, you could add an action for notification, such as an e-mail.

The following is the data type definition for the stable.xml file. It is followed by a set of tables that define the values that can be defined for subelements of this file.

```
<!ELEMENT actions_table (table_entry)+>
<!ELEMENT table_entry (desired_state,current_state, action_entry+) >
<!ELEMENT desired_state (#PCDATA)>
<!ELEMENT current_state (#PCDATA)>
<!ELEMENT action_entry (operation,parameter*)>
<!ELEMENT operation (#PCDATA)>
<!ELEMENT parameter (name,value)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
```

Table 35 shows the subelements included in the <table_entry> elements. These elements identify a current state/desired state pair and the action to be taken.

Table 35. Subelements for <table_entry> element

Subelement	Values	Required	Comments
desired_state	String	Yes	Five character string that indicates a software package state. For example IC--- (installed), ----- (initial state).
current_state	String	Yes	Five character string that indicates a software package state. For example, IC--- (installed), ICU-- (installed undoable).
action_entry	See Table 36	Yes	Defines the action to be taken for this current state/desired state pair.

Table 36 shows the subelements included in the <action_entry> elements. These elements identify the action to be taken and any parameters that are required for the action.

Table 36. Subelements for <action_entry> element

Subelement	Values	Required	Comments
operation	String	Yes	Identifies the operation that is required to change the software package state from its current value to the desired state. For example, Install.
parameter	See Table 37 on page 157.	No	Identifies a parameter that is required for the operation and assigns it a value.

Table 37 on page 157 shows the subelements included in the <parameter> elements. These elements identify parameters that are required for an operation and assign values to them.

Table 37. Subelements for <parameter> element

Subelement	Values	Required	Comments
name	String	Yes	Identifies a parameter that can be specified for the operation identified in the <action_entry> element.
value	String	Yes	Assigns a value to the parameter. For example, if the parameter is Undo, possible values are UNDO_YES and UNDO_NO.

The following extract from the current implementation of the stable.xml file shows the way in which actions are defined.

```
<?xml version='1.0'?>
<!DOCTYPE actions_table SYSTEM "stable.dtd">
<!-- Revision: 61 1.3 stable.xml, docs, xml4j2, xml4j2_0_0>
<actions_table>
  <table_entry>
    <desired_state>IC---</desired_state>
    <current_state>-----</current_state>
    <action_entry>
      <operation>Install</operation>
    </action_entry>
  </table_entry>
  <table_entry>
    <desired_state>IP---</desired_state>
    <current_state>-----</current_state>
    <action_entry>
      <operation>Install</operation>
      <parameter>
        <name>Transactional</name>
        <value>TRAN_YES</value>
      </parameter>
    </action_entry>
  </table_entry>
  <table_entry>
    <desired_state>ICU--</desired_state>
    <current_state>-----</current_state>
    <action_entry>
      <operation>Install</operation>
      <parameter>
        <name>Undo</name>
        <value>UNDO_YES</value>
      </parameter>
    </action_entry>
  </table_entry>
</actions_table>
```

Table 38 summarizes the actions defined in the stable.xml file. Only a subset of the actions are available for Web subscribers.

Table 38. Software states and actions

Desired State	Current State	Operation	Parameters	Available for Web Subscribers
IC-----	----- (initial)	Install		3
IP----	----- (initial)	Install	Transactional	
ICU --	----- (initial)	Install	Undoable	
IPU--	----- (initial)	Install	Transactional Undoable	

Table 38. Software states and actions (continued)

Desired State	Current State	Operation	Parameters	Available for Web Subscribers
IPP--	----- (initial)	Install	Transactional Undoable_in_ transactional	
-----	RC			
IC---	RC---	Install		3
IC---	RC---E	Install	Force	
IP---	RC---	Install	Transactional	
IP---	RC---E	Install	Force	
ICU--	RC---	Install	Undoable	
ICU--	RC---E	Install	Force	
IPU--	RC---	Install	Transactional Undoable	
IPU--	RC---E	Install	Transactional Undoable	
IPP--	RC--	Install	Transactional Undoable_in_ transactional	
IPP--	RC--E	Install	Transactional Undoable_in_ transactional, Force	
IC---	IP---	Commit		3
IC---	IP---E	Force		
-----	IP---	Undo		3
-----	IP---E	Remove	Force	
IP---	IPP--	Accept		
IP---	IPP--E	Install	Transactional Force	
ICU--	IPP--	Commit		
ICU--	IPP--E	Install	Undoable Force	
-----	IPP--	Undo		3
-----	IPP--E	Remove	Force	
IP---	IPU--	Accept		
IP---	IPU--E	Install	Transactional Force	
ICU--	IPU--	Commit		
ICU---	IPU--E	Install	Undoable Force	
-----	IPU--	Undo		3
-----	IPU--E	Remove	Force	
-----	IC---	Remove		3
-----	IC--E	Remove	Force	

Table 38. Software states and actions (continued)

Desired State	Current State	Operation	Parameters	Available for Web Subscribers
RP---	IC---	Remove	Transactional	
RCU--	IC---	Remove	Undoable	
RPU--	IC---	Remove	Transactional Undoable	
RPP--	IC---	Remove	Transactional Undoable	
ICU--	IC---	Install	Undoable Force	
ICU---	IC--E	Install	Undoable Force	
IC---	IC--E	Install	Force	
IC---	ICU--	Accept		3
IC---	ICU--E	Install	Force	
-----	ICU	Undo		3
-----	ICU--E	Remove	Force	
RCU--	-----	Remove	Undoable	
-----	ICR--	Commit		3
RC---	RP---	Commit	Reboot	
-----	RP---	Commit		3
IC	RP	Undo		
RP---	RPP--	Accept		
RCU--	RPP--	Commit		
IC---	RPP--	Undo		3
RP---	RPU--	Accept		
RCU--	RPU--	Commit		
IC---	RPU--	Undo		3
RC---	RCU--	Accept		3
-----	RCU--	Undo	Transactional Reboot	
IC---	RCU--	Undo		3
-----	RCR--	Commit	Reboot	
IC---	RCR--	Commit		3

If the current state of the software package is not among the statuses supported by the stable.xml file, for example if the software package is in - - - C state which is by definition a transitory state, an error message is displayed and the remaining operations are not performed.

In this case, if you want the operations to proceed ignoring the error, you can add the following lines to the stable.xml file:

The stable.xml File

```
<table_entry>
  <desired_state>IC---</desired_state>
  <current_state>---C</current_state>
</table_entry>
```

These lines indicated that no action is to be performed if the current state of the software package is Changing.

The invtable.xml File

If you have installed the Inventory Scan plug-in, the invtable.xml file is the basis on which Change Manager can determine the actions to be taken to bring a subscriber to the desired state defined in Inventory element. The file contains definitions for every possible combination of current state and desired state. Each entry defines the actions to be taken to change from the current state to the desired state. For example, if current state is the initial state "not scanned" and the desired state is "scanned", the required operation is "Scan".

The use of this file to define actions required for each current state/desired state pair, allows the introduction of new inventory configuration profiles in later releases. It also allows you to define additional actions. For example, you could add an action for notification, such as an e-mail.

The following is an example of the invtable.xml file.

```
<?xml version='1.0'?>
<!DOCTYPE actions_table SYSTEM "invtable.dtd">
<!-- Revision: 61 1.3 stable.xml, docs, xml4j2, xml4j2_0_0 -->
<actions_table>
  <table_entry>
    <desired_state>Scanned</desired_state>
    <current_state>NotScanned</current_state>
    <action_entry>
      <operation>Scan</operation>
    </action_entry>
  </table_entry>
```

The ostable.xml File

If you have installed the Pristine Manager plug-in, the ostable.xml file is the basis on which Change Manager can determine the actions to be taken to bring a subscriber to the Installed state defined in the operating system element. Each entry defines the actions to be taken to change from the current state to the desired state. For example, if current state is the initial state NotInstalled and the desired state is Installed, the required operation is PristineInstall.

The following is an example of the ostable.xml file.

```
<?xml version='1.0'?>
<!DOCTYPE actions_table SYSTEM "ostable.dtd">
<actions_table>
  <table_entry>
    <desired_state>Installed</desired_state>
    <current_state>NotInstalled</current_state>
    <action_entry>
      <operation>PristineInstall</operation>
    </action_entry>
  </table_entry>
</actions_table>
```

Reference Models

The reference model is the core concept of Change Manager. Independently of any specific subscriber, it provides a profile of a desired configuration to which subscribers can conform.

A reference model is normally made up of several reference models arranged in a hierarchy. At the top level is the parent, or root reference model, which defines requirements that are inherited by all child reference models within the hierarchy. Each child reference model defines further requirements that can supplement, override, or block the requirements defined in the parent model. The types of actions you can perform on reference models depends on the registered set of plug-ins.

Change Manager allows you to have multiple root models. This provides the ability to manage as many reference model structures as you want. For information about using the **Create new root** check box on the new reference model Properties dialog box, see “Creating the Reference Model Structure” on page 172.

You define the software and hardware requirements by adding configuration elements to a reference model. Change Manager includes the following types of configuration element:

- Software Distribution elements, which allow you to define the desired state (change management status) of a specified software package.
- Inventory data elements, which allow you to check the reference model against the Inventory database, for example to verify hardware prerequisites.
- Inventory scan elements, which allow you to perform hardware and software scans.
- Operating system elements, enable you to do a pristine installation remotely.

For Software Distribution elements, you select a software package and define the state, for example, IC, that you require. For a full description of the use of software package states, refer to the chapter “Understanding Change Management Operations” in the *Reference Manual for Software Distribution*.

For inventory data elements, you define an expression that includes a property selected from one of the Inventory tables. For example, `Physical_Memory >= 131072`.

For inventory scan elements, you specify that you want a previously customized Inventory Configuration to perform a hardware or software scan.

After the reference models have been defined, you can select the workstations, devices, or users you want to be subscribers to the model.

Reference Model Example

In the example organization, Enterprise, the network users can be divided into three distinct functional groups:

- Managers
- C++ developers
- Java developers
- New Hires

Reference Models

Figure 2 shows the hierarchical structure of a reference model that could be used to define requirements for the subscribers to the three groups.

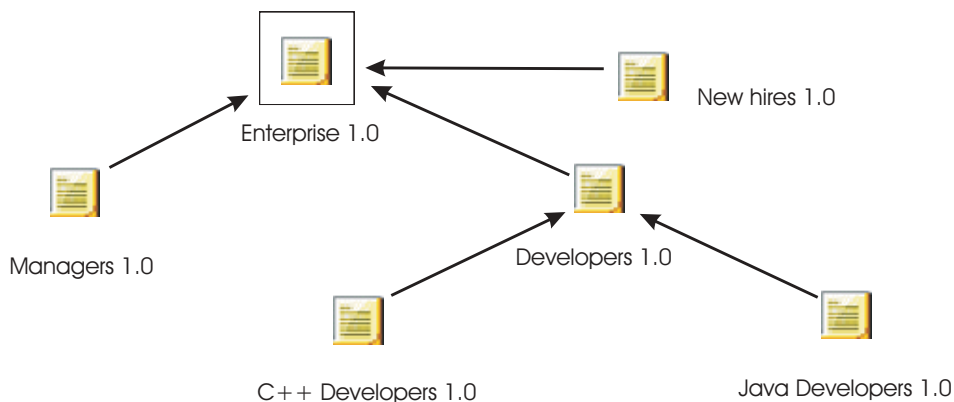


Figure 2. Example Reference Model

The Enterprise 1.0 reference model is the root of a hierarchical tree. It contains configuration elements that define requirements common to all three groups. It has three direct child reference models, Managers 1.0, Developers 1.0 and New hires 1.0, which inherit the configuration elements defined in Enterprise 1.0. Developers 1.0 also has two child reference models, C++ Developers 1.0 and Java Developers 1.0, which inherit the configuration elements defined in Developers 1.0 and those inherited by Developers 1.0 from Enterprise 1.0. New hires has no child reference models.

Table 39 shows a summary of the elements defined for the Enterprise 1.0 reference model.

Table 39. Summary of Elements in the Example Reference Model

Reference Model	Element	
	Type	Description
Enterprise	Inventory Scan	Hardware scan of the target machines.
	SWD	Package Lotus Notes Template.5 in ICU state
	SWD	Lotus Sametime.3.0 in ICU state
Managers	SWD	Package CMVC.1.0 in IC state
	SWD	Package Office2000.1.0 in IC state
Developers	InventoryData	Physical memory at least 523580 KB
	SWD	Package CMVC.1.0 in ICU state
	SWD	Package WinMerge1.8 in ICU state as a corequisite of the CMVC.1.0 package installation
Java Developers	SWD	Package JDK.1.3.1 in IC state
	SWD	Package WSAD.5.0 in IC state
C++ Developers	SWD	Package Visual Studio.6.0 in IC state
New Hire	Operating System	Windows XP operating system in installed state
	SWD	Lotus Client 5.0 in IC state

Table 40 shows the subscribers assigned to each child reference model that represents a group of users. The root reference model, Enterprise, does not have any subscribers. It defines general requirements for all groups.

Table 40. Subscribers to the Example Reference Model

Reference Model	Subscribers
Enterprise	-
Managers	Inventory subscriber, CSV subscriber
Developers	Query Library subscriber, Directory Query Library subscriber
Java Developers	Query Library subscriber, Static subscriber
C++ Developers	Query Library subscriber, Profile Manager subscriber
New Hires	Pristine Target subscriber

Representing a Reference Model in XML Format

One way you can store reference models for reuse is by exporting them to an XML file. You can access this file and make changes to it using any XML or text editor. When you reimport the file to Change Manager, the graphical representation shows the changes you have defined.

If an XML representation is to handle a model that can be correctly imported and displayed graphically within Change Manager, it must conform to the following data definition.

```
<!ELEMENT model (name,version,description?,attribute*,subscriber+,element*,model*)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT version (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT attribute (name,value)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT element (name,type,desired_state,attribute*,dependency*)>
<!ELEMENT type (#PCDATA)> <!ELEMENT desired_state (#PCDATA)>
<!ELEMENT dependency (name,type,desired_state,attribute*)>
<!ELEMENT subscriber (name,type,excluded,attribute*,subscriber*)>
<!ELEMENT excluded (#PCDATA)>
```

This file defines the general structure of the xml reference model file. The root element is the <model> element, representing the reference model as a whole. Further <model> elements can be nested within the root <model> element, representing child reference models within the model.

The attributes of each reference model is a key/value pair that stores information for internal use. The key is a string identifying a property and the value is a string used to store that property's value. For example, an attribute for a reference can store such information as the default view of that model selected by the user. An attribute for an element can store the multicast options for that element. An attribute for a subscriber can store information like the subscriber type.

Table 41 summarizes the subelements contained within a <model> element.

Table 41. Subelements for <model> elements

Subelement	Values	Required	Comments
name	String	Yes	Identifies a reference model.

Table 41. Subelements for <model> elements (continued)

Subelement	Values	Required	Comments
version	String	No	Indicates the version of the reference model.
description	String	No	Contains a short description of the reference model.
element	See Table 42.	Yes	Defines the attributes of a configuration element.
subscriber	See Table 44 on page 165.	No	Defines the attributes of a subscriber to the reference model.
model		No	Defines the attributes of a reference model that is a child of the current model.
attribute	String	No	A key/value pair that defines attributes of the model.

Table 42 summarizes the subelements contained within an <element> element. Each element represents a configuration element, which defines a hardware or software requirement that is part of the reference model profile.

Table 42. Subelements for <element> elements

Subelement	Values	Required	Comments
name	String	Yes	For Software Distribution, and InventoryScan elements this is the name of a profile manager. For Pristine Manager, this is the name of an operating system element. For an InventoryData element this is an expression based on the values of one or more inventory table columns. For example, OS_Name LIKE "Windows%".
type	String	Yes	Identifies the class that implements this element. In the current implementation, possible classes are com.tivoli.ccm.plugin.sd.data.SWDElement (SoftwareDistribution) com.tivoli.ccm.plugin.inv.data.INVElement (InventoryData) com.tivoli.ccm.plugin.invscan.data.InvScanElement (InventoryScan), and com.tivoli.cm4os.ccm.data.oselement (OSElement).
desired_state	String	Yes	The string identifying the final state that subscribers should be changed to. For example for SoftwareDistribution elements one of the allowed values is IC--- (installed) while for InventoryScan elements the only available state is "Scanned". This state does not apply to InventoryData elements. For Pristine Manager, the only applicable state is installed.

Table 42. Subelements for <element> elements (continued)

Subelement	Values	Required	Comments
dependency	See Table 43.	No	Defines the attributes of a software dependency. This is only applicable to SWD elements.
attribute	String	No	A key/value pair that defines attributes of the element.

Table 43 summarizes the subelements contained within a <dependency> element. The <dependency> element is a subelement of <element> that is only allowed if <element> defines an SWD element.

Table 43. Subelements for <dependency> elements

Subelement	Values	Required	Comments
name	String	Yes	Identifies a software package.
type	String	No	Identifies the class that implements this dependency. Possible values in the default implementation are SWDPreReq , SWDCoReq , and SWDExReq .
desired_state	String	Yes	The five character string that indicates a software package state. For example IC--- (installed).
attribute	String	No	A key/value pair that defines attributes of the dependency element.

Table 44 summarizes the subelements contained within a <subscriber> element. Each subscriber element identifies one or more targets that are selected for inclusion in or exclusion from the list of subscribers to the reference model.

Table 44. Subelements for <subscriber> elements

Subelement	Values	Required	Comments
name	String	Yes	Identifies a subscriber.
type	String	No	Identifies the class that implements the selection of subscribers. The default implementation of Change Manager includes the following subscriber types: CSVSubscriber, ProfileManagerSubscriber, StaticSubscriber, QueryLibrarySubscriber, QueryDirectoryLibrarySubscriber, and WebSubscriber (this subscriber is available only when migrating from Configuration Manager 4.2.3). If you install the InventoryData plug-in, the INVSubscriber subscriber type is available. If you install the OSElement plug-in, the PristineTargetSubscriber and GroupSubscriber subscriber types are available.

Reference Models

Table 44. Subelements for <subscriber> elements (continued)

Subelement	Values	Required	Comments
excluded	True False	Yes	Indicates whether the subscriber is to be included or excluded from the list of subscribers to the model.
subscriber	String	No	For Web subscribers, indicates the way in which targets are selected. Possible values are the same as for subscriber type, except WebSubscriber.
attribute	String	No	A key/value pair that defines attributes of the subscriber.

Figure 3 is a schematic representation of the reference model shown in Figure 2 on page 162. Figure 2 shows the reference model with configuration elements and subscribers defined for the component parts of the model. It is followed by the XML mark-up that defines the structure represented in the diagram.

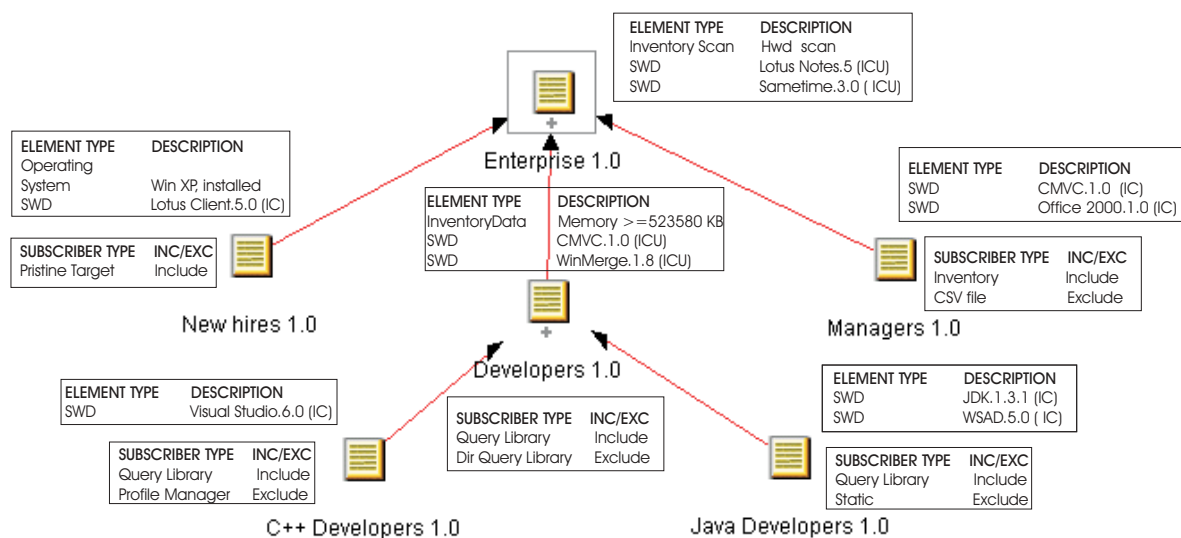


Figure 3. Schematic representation of a reference model

```
<?xml version="1.0"?>
<!DOCTYPE model SYSTEM "model.dtd">
<model>
  <name>Enterprise</name>
  <version></version>
  <description/>
  <attribute>
    <name>DefaultView</name>
    <value>GridViewId</value>
  </attribute>
  <element>
    <name>INVENTORYDATA.PHYSICAL_MEMORY_KB &gt; 131072</name>
    <type>com.tivoli.ccm.plugin.inv.data.INVelement</type>
    <desired state><desired_state/>
    <attribute>
```

```

        <name>priorityLevel</name>
        <value>1</value>
    </attribute>
    <attribute>
        <name>prioritySet</name>
        <value>high</value>
    </attribute>
</element>
<element>
    <name>Lotus Notes^1.0</name>
    <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
    <desired_state>IC---</desired_state>
    <attribute>
        <name>priorityLevel</name>
        <value>2</value>
    </attribute>
    <attribute>
        <name>prioritySet</name>
        <value>high</value>
    </attribute>
    <attribute>
        <name>Retry unicast</name>
        <value>F</value>
    </attribute>
    <attribute>
        <name>Enable multicast</name>
        <value>F</value>
    </attribute>
</element>
<model>
    <name>Developers</name>
    <version></version>
    <description/>
    <element>
        <name>CMVC^1.0</name>
        <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
        <desired_state>ICU--</desired_state>
        <attribute>
            <name>priorityLevel</name>
            <value>1</value>
        </attribute>
        <attribute>
            <name>prioritySet</name>
            <value>high</value>
        </attribute>

        <attribute>
            <name>Retry unicast</name>
            <value>F</value>
        </attribute>
        <attribute>
            <name>Enable multicast</name>
            <value>F</value>
        </attribute>
    </element>
    <subscriber>
        <name>QueryLib1</name>
        <type>com.tivoli.ccm.data.QueryLibrarySubscriber</type>
        <excluded>false</excluded>
        <attribute>
            <name>target_resource_type</name>
            <value>com.tivoli.ccm.data.EndpointType</value>
        </attribute>
    </subscriber>
</model>
    <name>C++ Developers</name>
    <version></version>

```

```

<description/>
<element>
  <name>Visual Studio^6.0</name>
  <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
  <desired_state>IC---</desired_state>
  <attribute>
    <name>priorityLevel</name>
    <value>1</value>
  </attribute>
  <attribute>
    <name>prioritySet</name>
    <value>high</value>
  </attribute>
  <attribute>
    <name>Retry unicast</name>
    <value>F</value>
  </attribute>
  <attribute>
    <name>Enable multicast</name>
    <value>F</value>
  </attribute>
</element>
<subscriber>
  <name>QueryLib2</name>
  <type>com.tivoli.ccm.data.QueryLibrarySubscriber</type>
  <excluded>false</excluded>
  <attribute>
    <name>target_resource_type</name>
    <value>com.tivoli.ccm.data.EndpointType</value>
  </attribute>
</subscriber>
<subscriber>
  <name>pm1</name>
  <type>com.tivoli.ccm.data.ProfileManagerSubscriber</type>
  <excluded>true</excluded>
  <attribute>
    <name>target_resource_type</name>
    <value>com.tivoli.ccm.data.EndpointType</value>
  </attribute>
</subscriber>
</model>
<model>
  <name>Java Develoeprs</name>
  <version></version>
  <description/>
  <subscriber>
    <name>target1</name>
    <type>com.tivoli.ccm.data.StaticSubscriber</type>
    <excluded>true</excluded>
    <attribute>
      <name>target_resource_type</name>
      <value>com.tivoli.ccm.data.EndpointType</value>
    </attribute>
  </subscriber>
  <subscriber>
    <name>QueryLib3</name>
    <type>com.tivoli.ccm.data.QueryLibrarySubscriber</type>
    <excluded>false</excluded>
    <attribute>
      <name>target_resource_type</name>
      <value>com.tivoli.ccm.data.EndpointType</value>
    </attribute>
  </subscriber>
</element>
  <name>JDK^1.8</name>
  <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
  <desired_state>IC---</desired_state>

```

```

        <attribute>
          <name>Retry unicast</name>
          <value>F</value>
        </attribute>
        <attribute>
          <name>Enable multicast</name>
          <value>F</value>
        </attribute>
      </element>
    </model>
  </model>
<model>
  <name>Managers</name>
  <version></version>
  <description/>
  <element>
    <name>Office2000^1.0</name>
    <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
    <desired_state>IC---</desired_state>
    <attribute>
      <name>priorityLevel</name>
      <value>1</value>
    </attribute>
    <attribute>
      <name>prioritySet</name>
      <value>high</value>
    </attribute>
    <attribute>
      <name>Retry unicast</name>
      <value>F</value>
    </attribute>
    <attribute>
      <name>Enable multicast</name>
      <value>F</value>
    </attribute>
  </element>
  <element>
    <name>CMVC^1.0</name>
    <type>com.tivoli.ccm.plugin.sd.data.SWDElement</type>
    <desired_state>IC---</desired_state>
    <attribute>
      <name>priorityLevel</name>
      <value>1</value>
    </attribute>
    <attribute>
      <name>prioritySet</name>
      <value>high</value>
    </attribute>
    <attribute>
      <name>Retry unicast</name>
      <value>F</value>
    </attribute>
    <attribute>
      <name>Enable multicast</name>
      <value>F</value>
    </attribute>
  </element>
  <subscriber>
    <name>F:\CCM\test\subs.csv</name>
    <type>com.tivoli.ccm.data.CSVSubscriber</type>
    <excluded>true</excluded>
    <attribute>
      <name>target_resource_type</name>
      <value>com.tivoli.ccm.data.EndpointType</value>
    </attribute>
  </subscriber>
</subscriber>
</model>

```

Reference Models

```

        <name>select TME_OBJECT_LABEL from INVENTORYDATA
where BOOTED_OS_NAME LIKE &apos;Win%&apos;</name>
        <type>com.tivoli.ccm.plugin.inv.data.INVSubscriber</type>
        <excluded>false</excluded>
        <attribute>
            <name>target_resource_type</name>
            <value>com.tivoli.ccm.data.EndpointType</value>
        </attribute>
    </subscriber>
</model>
</model>
```

Chapter 6. Performing Change Manager Operations

This chapter explains how to use Change Manager through the graphical user interface.

Launching the Change Manager GUI

You launch the Change Manager GUI from the Tivoli desktop. On UNIX machines, launching the GUI displays a login dialog box. Enter the following command before launching the GUIs to enable the graphic session:

```
xhost hostname
```

In the login dialog box, you specify the following information:

1. In the **Host Machine** text box, type the name of the host machine on which Change Manager is installed.
2. In the **Login as** text box, type an identified user account name for the specified host machine. Ensure that the login user account is a valid Tivoli administrator with the necessary Tivoli management region roles assigned.
3. In the **Password** text box, type the password associated with the specified login user.

If the Tivoli management region and the managed node are located on different domains, you can start the Change Manager component using the command line only. For more information about starting the GUI from the command line, see “wccmgui” on page 197.

Note: On Windows machines, if the Tivoli desktop is not installed when you install Change Manager or if you remove the Tivoli desktop after installing Change Manager, the login dialog box is displayed when starting the GUI and must be filled in as explained above.

Using Change Manager

Using the information and instructions included in this section, you can complete the following tasks:

- Create a reference model that includes parent and child models. See “Creating the Reference Model Structure” on page 172.
- Import a reference model. See “Importing a Reference Model” on page 175.
- Create a reference model template using another model as a target. See “Creating a Reference Model from a Target” on page 173.
- Add configuration elements to the reference model. See “Adding Elements to a Reference Model” on page 176.
- Assign subscribers to the reference model. See “Assigning Subscribers to a Reference Model” on page 180.
- Export reference models and save them in an XML file format. See “Exporting a Reference Model in XML Format” on page 175.
- Work with existing reference models, reimporting, modifying, and deleting them. See “Modifying a Reference Model” on page 186.
- Produce and submit an activity plan for the reference model. See “Change Manager and Activity Plans” on page 188.

Using Change Manager

- Assign a priority to configuration elements in the reference model. See “Assigning a Priority to Configuration Elements” on page 189.
- Produce reports. See “Creating Change Manager Reports” on page 192.

To start Change Manager, select the Change Configuration icon on the Tivoli desktop. As a user with the Change Manager Edit role, you can select all of the available Change Manager functions.

When you start Change Manager, the Change Manager window appears. It is divided into upper and lower panes. In the upper pane, you define the structure of the reference model by opening one of the available reference models or creating a new one. In the lower pane, you add configuration elements and subscribers to the reference model structure you have defined in the upper pane.

Note: If the Tivoli Management region and the managed node are located on different domains or in case the name of the managed node is different from the hostname, you can start the Change Manager component only using the command line. For more information on starting the GUIs from the command line, see “wccmgui” on page 197.

There are three methods of accessing the Change Manager functions, although not all three ways are available for all functions:

- From the menus at the top of the Main window.
- By using the icons.
- From the pop-up menu.

You can start Change Manager using the `wccmgui` command. For more information on starting the GUIs from the command line, see “wccmgui” on page 197.

Creating the Reference Model Structure

To create the reference model structure described in the reference model example in this chapter, perform the following steps:

1. From the **Edit** menu, select Create reference model. The Properties dialog is displayed.



2. In the **Name** text box, enter **Enterprise** as the name of the parent reference model. Optionally, you can also enter version and description information.
3. In the **Version** text box enter **1.0** (this text box is optional).

4. In the **Description** text box enter the descriptive text **New enterprise model** to describe the reference model (this text box is optional).
5. If this node in the Reference Model structure is a new root model, select the **Create new root** check box.
6. Click **OK**. Change Manager creates the new parent reference model **Enterprise**.
7. Select the **Enterprise** reference model and right-click to display the pop-up menu.
8. Select **New reference model**. The Properties dialog box appears, prompting you for the properties of the new child reference model.
9. In the **Name** text box, enter the name of the new reference model, **Managers** or **Developers**. Optionally, you can also enter version and description information.
10. Click **OK**. Change Manager creates the new child reference model.
11. Continue adding child reference models until the reference model structure is complete. In the example, this means adding one more child model to **Enterprise**, and two child models to **Developers**.

Creating a Reference Model from a Target

Change Manager provides the capability to automatically create a reference model using the current configuration of a target user, device, or endpoint. Depending on the plug-ins you have registered, you can select which specific configuration elements from the target you want included in the new model. After you have created the reference model, you can then assign subscribers as described in “Assigning Subscribers to a Reference Model” on page 180. After you have assigned subscribers, you can preview or submit an activity plan for the target subscribers as described in “Previewing an Activity Plan” on page 189 and “Submitting an Activity Plan” on page 191.

To create a reference model from a target, perform the following steps:

1. From the **Edit** menu, select **Create reference model from target**. The Properties dialog box is displayed. On the Properties dialog box, the availability of the **Hardware** and **Software** check boxes and their associated configuration elements depends on the plug-ins you have registered.

Using Change Manager



2. In the **Name** text box, enter the name you want to give the new reference model. Optionally, you can also enter version and description information in the **Version** and **Description** text boxes.
3. If the new model is to be a new root model, select the **Create new root** check box.
4. In the **Target name** field, enter the name of the target from which you are creating the new reference model, or click the browse button (...) to navigate to the location of the target and select it. Use the radio buttons below the **Target name** field to specify the appropriate target type. If you selected the endpoint subscriber type, you can browse the endpoints. If you selected the device type, the navigator is disabled since it is not possible to navigate the individual device instances. If you selected the user type, the navigator displays the association between a user and the related endpoint.
5. Select the **Hardware** check box if you want the new model to perform checks on the selected hardware configuration elements from the target. If you select this check box, you must also select the specific hardware elements you want included.
6. Select the **Software** check box if you want the new model to include all the target's software configuration elements.
7. Click **OK**. Change Manager creates the new reference model with the name you specified in the **Name** field.

Importing a Reference Model

To import a reference model, perform the following steps:

1. From the **File** menu, select **Import**. The Import settings dialog box appears.



2. In the **Reference model definition file (XML)** text box, enter the name of the XML file you want to import. If necessary, use the browse (...) button to locate the file.
3. In the **Encoding type** field, enter the encoding type you want to use. For a complete list of the supported encoding types, see the Sun Microsystems Web Site.
4. If you want the imported reference model to be a root, select the **Import as new root model** check box.
5. Click **OK**. Change Manager imports the specified reference model and displays it in the upper pane. If you selected the **Import as new root model** check box, the newly imported root model is displayed in the upper pane.

Exporting a Reference Model in XML Format

When you have finished building the reference model, you can export it to an XML file format. In this format, you can read the file and edit it in any standard text or XML editor and reimport the file to Change Manager.

To export a reference model to xml format, perform the following steps:

1. From the **File** menu select **Export**. The Export settings dialog box appears.



2. In the **Reference model definition file (XML)** field, enter the name you want to give the XML file. If necessary, use the browse (...) button to browse for the appropriate directory in which to store the file.
3. Select the **Include inherited elements** check box if you want to export elements the model inherited from the parent model. Select **Export single reference model** check box if you want to export only the selected reference model itself and no child models. These check boxes are optional so you can select one, both, or neither. If you select neither check box, the model is exported with its child models, and without elements inherited from the parent model.
4. Click **OK**. The reference model is exported in XML format to the specified file.

Adding Elements to a Reference Model

You can add a new element to a new or restored reference model, by choosing an available element type and entering the element name and the desired state. The new element is recorded in the Change Manager repository. This section includes instructions for

- “Adding an Inventory Data Element.”
- “Adding an Inventory Scan Element” on page 177
- “Adding an Operating System Element” on page 177
- “Adding a Software Distribution Element” on page 178

To add a configuration element, you need to have registered the corresponding plug-in, as described in the following table:

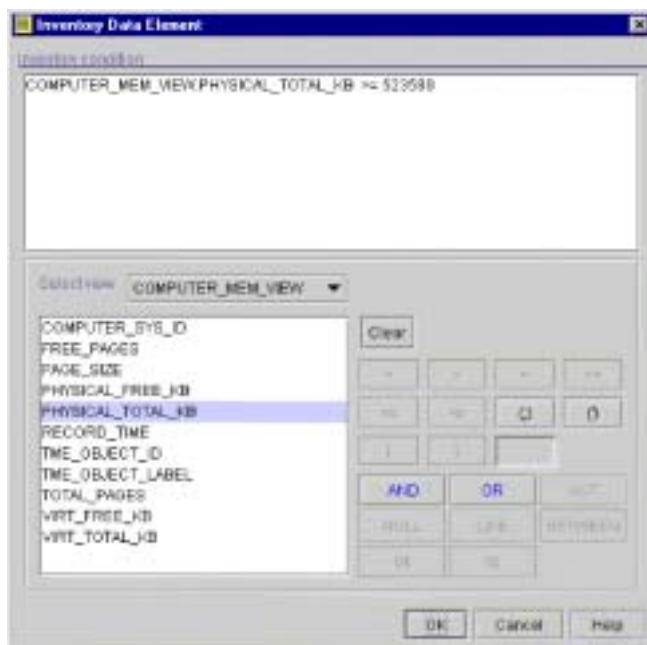
Table 45.

Configuration Element	Plug-in
Inventory Data	Inventory Data
Inventory Scan	Inventory Scan
Operating System element	Pristine Manager
Software Distribution	Software Distribution

Adding an Inventory Data Element

To add an inventory data element to a reference model, perform the following steps:

1. In the upper pane, select the reference model.
2. In the lower pane, select the **Elements** tab.
3. Right-click in the Elements pane to display the pop-up menu.
4. Select **Inventory Element**. The Inventory element dialog box appears:



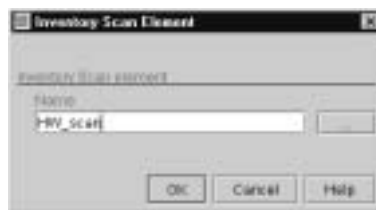
5. Build an expression in the **Inventory Condition** text box, as follows:
 - a. Select an inventory table view from the **Select view** drop down list. A list of columns for the table view appears in the lower text box.

- b. Select a column from the list and double-click. The column name appears in the **Inventory Condition** text box.
 - c. From the panel of operators, select an operator, for example "=", and double-click. The operator is added to the expression in the **Inventory Condition** text box.
 - d. Click in the **Inventory** text box at the end of the expression and type in a value, for example a minimum memory size.
 - e. If you want to build a more complex expression, select the AND or OR operators from the operator panel and then add another condition to the expression.
6. Click **OK**. The new inventory element is added to the reference model.

Adding an Inventory Scan Element

To add an inventory scan element to a reference model, perform the following steps:

1. In the upper pane, select the reference model.
2. In the lower pane, select the **Elements** tab.
3. Right-click in the Elements pane to display the pop-up menu.
4. Select **Inventory Scan Element**. The Inventory Scan Element dialog box appears:

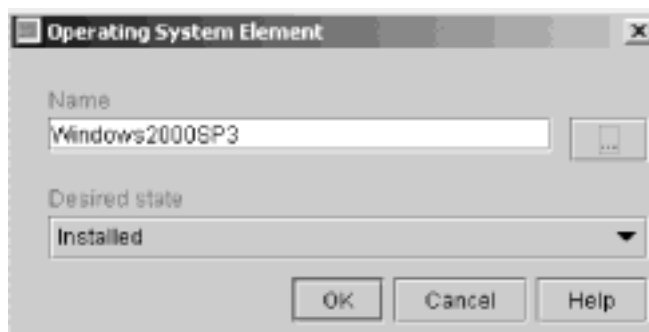


5. In the **Name** field, enter the name of the inventory configuration profile.
If you do not know the name of the profile, you can click the browse button to navigate to the profile you want, or click the Search button available in the navigator dialog.
6. Click **OK**. The new inventory scan element is added to the reference model.

Adding an Operating System Element

Perform the following steps:

1. In the upper pane of the Change Manager window, select the reference model.
2. In the lower pane, select the **Elements** tab.
3. Right-click in the Elements pane to display the pop-up menu.
4. Click **Add** and then select **Operating System Element**. The Operating System Element dialog is displayed.



5. Complete the following fields:

Name Enter the name of the operating system element.

If you do not know the name of the operating system element, click the browse button to find the operating system element you want. The Select Operating System dialog shows a list of operating system elements. Select an operating system element and click **OK**. The Operating System Element dialog displays your choice.

Desired state

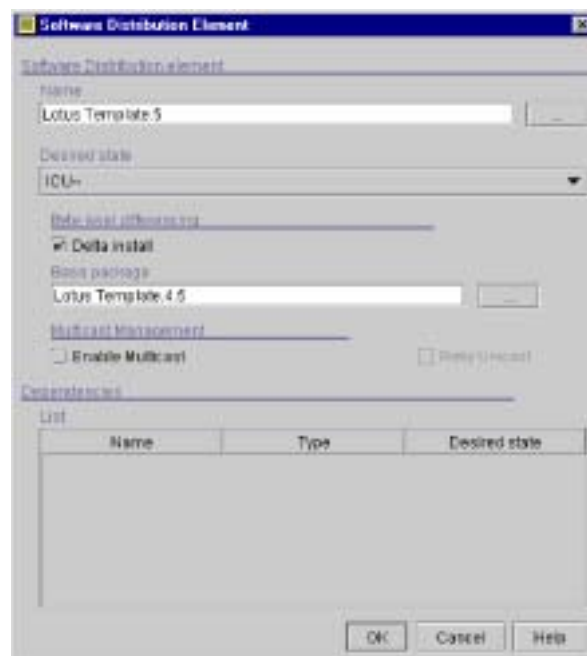
The state that the pristine machine should reach. It is Installed by default and read-only, because the action that the element performs is a pristine installation. You cannot choose an undo or uninstall state.

6. Click **OK**. The new operating system element is added to the reference model.

Adding a Software Distribution Element

To add an software distribution element to a reference model, perform the following steps:

1. In the upper pane, select the reference model.
2. In the lower pane, select the **Elements** tab.
3. Right-click in the Elements pane to display the elements pop-up menu.
4. Select **Software Distribution element**. The Software Distribution element dialog box appears.



5. Enter the name of a software package.

If you do not know the name of the package, you can click the browse button to navigate to a list of the software packages that are available for each accessible profile manager, or click the Search button available in the navigator dialog.

6. Select the required software package state from the **Desired state** drop-down list. For more information on software package states, see *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*.

Note: If the reference model is addressed to devices, the only supported desired states are:

- IC - - - (Installed Committed)
- - - - - (Removed Committed)

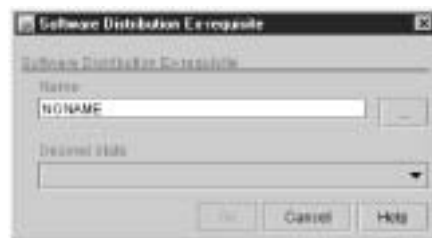
7. Click the **Distribution Options** button to display the **Distribution Options** dialog box. In this dialog box, you can perform one of the following operations:
 - Use the **Multicast Management** check boxes to enable the multicasting distribution capability. Select the **Enable Multicast** check box to enable data broadcasting to multiple repeaters. Multicast sends only one distribution from the source to a group of targets simultaneously. Use this option where there is limited network bandwidth.
 - Optionally select the **Retry Unicast** check box to retransmit the distribution independently to each endpoint that failed to receive the original multicast distribution. For more information about multicasting, see *Tivoli Management Framework: User's Guide*.
8. Use the **Byte-level differencing** field to specify a base software package. Software distribution then manages a delta package in the distribution process. For more information about byte-level differencing, see *User's Guide for Software Distribution*.

Note: The software distribution element uses the byte-level differencing feature of the Software Distribution application. This feature allows Software Distribution to handle a delta package created as a byte-to-byte difference between an already installed base package and a new version of this package that you want to be distributed. Instead of distributing the whole new package, Software Distribution is able to detect what has changed and distribute only those changes. This reduces network traffic and improves performance.

9. To add a dependency, right-click in the Dependencies pane. A pop-up menu gives you the following options:
 - Ex-requisite
 - Pre-requisite
 - Co-requisite

To define conditions between reference models, you can assign a priority to the elements contained in the reference model. For more information, see “Assigning a Priority to Configuration Elements” on page 189.

10. Select the required dependency type. The corresponding dialog box is displayed.



11. In the **Name** text box, Enter the name of the software package that is to be the prerequisite, ex-requisite, or corequisite of the software package you specified for this element.

Using Change Manager

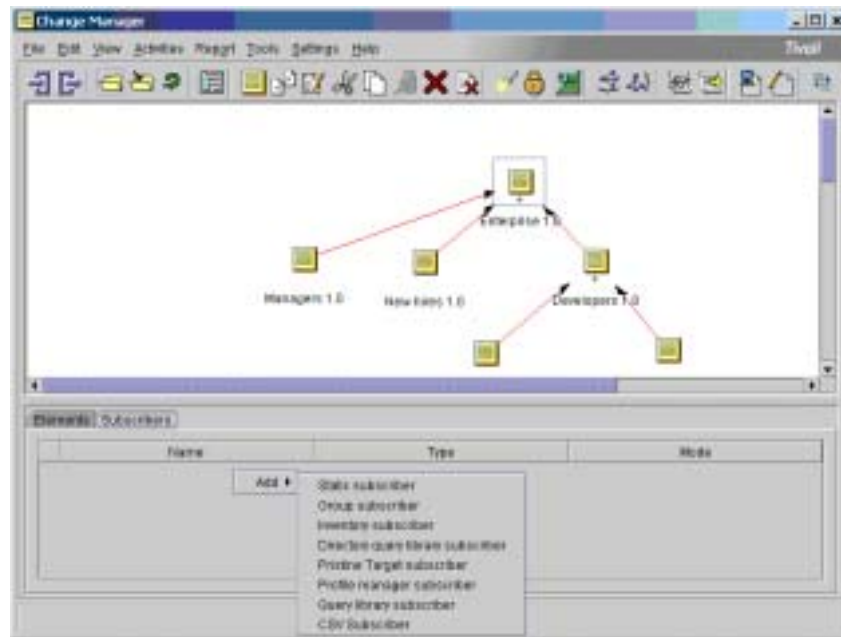
If you do not know the name of the package, you can click the browse button to navigate to a list of the software packages that are available for each accessible profile manager.

12. Select the required software package state for the dependency from the **Desired state** drop-down list.
13. Click **OK**. The dependency dialog box closes and the dependency is shown in the dependency list for the SWD element.
14. Click **OK**. The new SWD element is added to the reference model.

Assigning Subscribers to a Reference Model

For detailed information about subscribers and subscriber types, see “Subscribers” on page 149. You can assign subscribers to any of the component models of your reference model. To assign subscribers to a model, perform the following steps:

1. In the upper pane of the Change Manager window, select the reference model to which you want to assign subscribers.
2. In the lower pane, select the **Subscribers** tab.
3. Right-click in the Subscribers pane to display the pop-up menu:



4. Select one of the following values and follow the corresponding instructions:
 - Inventory subscriber.
 - Profile manager subscriber.
 - CSV subscriber.
 - Static subscriber.
 - Query Library subscriber.
 - Directory Query Library subscriber.
 - Web User Interface (Web UI) subscriber.

The **Web subscriber** option is available only for Web UI version 4.1.

- Group subscriber
- Pristine target subscriber

Note: All methods of assigning subscribers, other than Static subscriber, are dynamic. When you add the subscriber, Change Manager shows you a list of selected targets. These are the targets that fit the selection criteria at this time.

Assigning an Inventory Subscriber

If you have registered the Inventory data plug-in, you can specify queries which access the Tivoli Inventory configuration database and selects a list of targets as the query result. To specify an inventory subscribers as a target using a Structure Query Language (SQL) query, perform the following steps:

1. Select **Inventory Subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The Inventory subscriber dialog box appears.



2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Choose an inventory table view from the **SQL statement** drop-down list. A list of columns for the selected view is displayed. The available views will vary depending on which subscriber type you selected.
4. Build an expression in the **Where** box to specify what information the query will return. To build a simple query, select a column from the list, select an operator from the panel, and then assign a constant by typing in the **Where** box. You can build more complex queries using the AND and OR operators. If you want a review of the query results, click **Get result**. All the results satisfying that condition are displayed in the third pane.

The query generated changes according to the selected subscriber type. If you select Endpoint or User subscriber types, the query is a selection on the field TME_OBJECT_LABEL. If you select the Device subscriber type, the query is a selection on the field LABEL. Similarly, the list of views available in the drop-down list changes according to the subscriber type you select.

Using Change Manager

5. Select either the **Include** or **Exclude** button and click **OK**. The inventory subscriber is added to the reference model.

Refer to the *User's Guide for Inventory* for more information about SQL queries.

Assigning a Profile Manager Subscriber

You can specify a profile manager as a subscriber to a reference model. The subscribers to the selected profile manager are the targets that are to match the configuration specified in the reference model. To specify a profile manager as a subscriber to a reference model, perform the following steps:

1. Select **Profile manager subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The Profile manager subscriber dialog box appears.

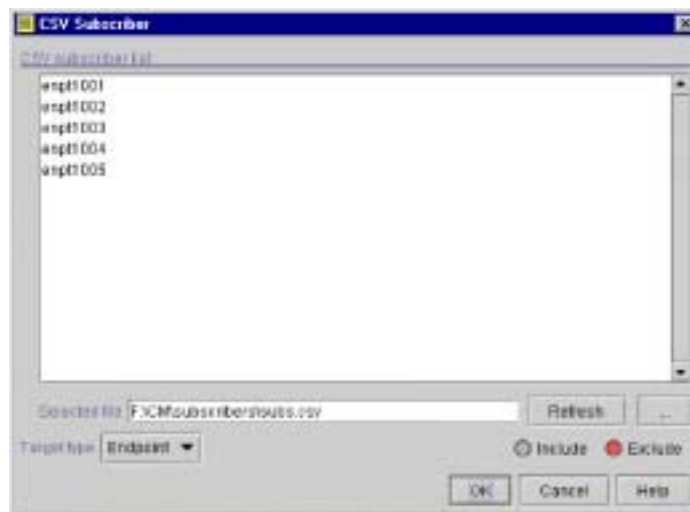


2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Enter the fully qualified name of the profile manager in the Profile manager text box, or click the browse button (...) to navigate to the location of the profile manager and select it.
4. Select either the **Include** or **Exclude** button and click **OK**. The profile manager subscriber is added to the reference model.

Assigning a CSV Subscriber

You can select the subscribers to a reference model by indicating the path to a file containing a list of target names separated by a comma. To specify a file name containing a list of targets, perform the following steps:

1. Select **CSV subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The CSV Subscriber dialog box appears:



2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Select the CSV file you want to include and display its contents in the CSV subscriber list, by doing one of the following:

- Type the fully qualified file name in the Selected file text box and click **Refresh**.
 - Click the browse button (...) to navigate to the file location. The available files will vary depending on which subscriber type you selected. Select the file, and click **Open** to return to the CSV subscriber dialog box.
4. Select either the **Include** or **Exclude** button and click **OK**. The CSV subscriber is added to the reference model.

Assigning a Static Subscriber

You can specify a list of target names to be subscribers to a reference model. To specify the subscribers to a reference model using a list of target names, perform the following steps:

1. Select **Static subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The Static subscriber dialog box is displayed:



2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Enter the name of the target you want to include in the **Subscriber** text box, or click the browse button (...) to navigate to the location of the target and select it. If you selected the endpoint subscriber type, you can browse the endpoints. If you selected the device subscriber type, the navigator is disabled since it is not possible to navigate the individual device instances. If you selected the user subscriber type, the navigator displays the association between a user and the related endpoint.
4. Click **Add**. The target is displayed in the **Selected subscriber list**.
5. Continue building the list by selecting other targets and clicking **Add**.
6. Select either the **Include** or **Exclude** button and click **OK**. The static subscriber is added to the reference model.

Assigning Query Library Subscribers

You can specify a query representing a query library to return a list of subscribers from the Inventory configuration repository database. To select a query library from the Tivoli object database to retrieve subscribers, perform the following steps:

Using Change Manager

1. Select **Query Library subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The Query Library subscriber dialog box is displayed:



2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Select a query library from the pull-down list. The contents of the query library are displayed in the box. If you want a list of the targets that satisfy the selected query library, click **Get result**. All the results satisfying the query library are displayed in the third pane.

The **Device** option is disabled because the Query Library Subscriber is available only for endpoints, which are eventually selected by an association to a user.
4. Select either the **Include** or **Exclude** button and click **OK**. The query library subscriber is added to the reference model.

Assigning Directory Query Library Subscribers

You can specify a query representing a directory query library to return a list of subscribers from the Inventory configuration repository database. To select a directory query library from the Tivoli object database to retrieve subscribers, perform the following steps:

1. Select **Directory query library subscriber** from the **Add** pop-up menu described in “Assigning Subscribers to a Reference Model” on page 180. The Directory query library subscriber dialog box appears:



2. Select the type of subscriber using the **Target Types** pull-down menu.
3. Select a query directory library from the pull-down list. The contents of the query library are displayed in the box. If you want a review a list of the targets that satisfy the selected directory query library, click **Get result**. All the results satisfying the directory query library are displayed in the third pane.
4. Click **Get result** to retrieve the list of targets that satisfy the selected query directory library.
5. Select either the **Include** or **Exclude** button and click **OK**. The query directory library subscriber is added to the reference model.

Assigning Pristine Manager Subscribers

To assign subscribers for a pristine installation to the reference model, perform the following steps:

1. In the upper pane of the Change Manager window, select the reference model.
2. In the lower pane, select the **Subscribers** tab.
3. Right-click in the Subscribers pane to display the pop-up menu and click **Add**. The menu lists various subscriber types, including those that are related to other Tivoli services. However, only the following types are valid for pristine installations:

Group subscriber

Subscribes a group of target machines.

Pristine target subscriber

Specifies one target at a time, similarly to the static subscriber for other Change Manager configuration elements.

4. Follow one of the following procedures:
If you selected **Group subscriber**, perform the following steps:

Using Change Manager

- a. The Group Subscriber dialog is displayed. Click the browse (...) button. The Group Manager dialog lists available groups. Select one or more groups and click **Close**. The Group Subscriber dialog is displayed.



- b. Click **OK**. The group is added to the Subscribers page.
If you selected **Pristine Target subscriber**, perform the following steps:
 - a. The Pristine Target Subscriber dialog is displayed. Click the browse (...) button. The Machine Manager dialog lists available machines. Select one or more machines and click **Close**. The Pristine Target Subscriber dialog displays the machines subscribed.



- b. Click **OK**. The machines are added to the Subscribers page.

Modifying a Reference Model

After you have created and saved a reference model, you can make changes to it to reflect new requirements. For example, once a reference model has been used to generate the actions required to bring subscribers to a required state, you can change the requirements by adding or modifying configuration elements. In

addition, you can use the Change Manager GUI to export the reference model in xml format as described in “Exporting a Reference Model in XML Format” on page 175. Alternatively, you can represent the reference model in xml format as described in “Representing a Reference Model in XML Format” on page 163. In this way, you can reuse the hierarchical structure you have created to make further changes to the subscribers.

To modify a reference model that you have exported to an xml format, you must import the model. When the model is displayed, you use the Change Manager GUI to change it in the following ways:

- Modify the structure of the reference model by moving child reference models to new parents.

To do this, select the reference model you want to move, click the edge of the highlighted area, and drag and drop the model until its line connects to the new parent.

You can also cut and paste reference models by selecting a model and using the **Cut** and **Paste** options on the **Edit** menu.

- Add new reference models to the structure.

See “Creating the Reference Model Structure” on page 172.

- Remove reference models from the structure.

To do this, select the reference model you want to remove, right-click, and select **Remove** from the pop-up menu.

Note: When you remove a parent reference model, the children are automatically removed.

- Modify the properties of a reference model.

To do this, select the reference model you want to modify, right-click, and select **Properties** from the pop-up menu. See “Creating the Reference Model Structure” on page 172, for information about the properties that must be defined for a reference model.

- Add a new configuration element to a reference model.

See “Adding Elements to a Reference Model” on page 176.

- Remove a configuration element.

To do this, select the reference model you want to remove an element from, in the lower pane select the element, right-click, and select **Remove** from the pop-up menu.

Note: The list of elements for a reference model indicates inherited elements by means of an up-arrow icon. Inherited elements can only be removed from the parent model where they are defined.

- Modify the properties of a configuration element, for example to specify a different software package.

To do this, select the reference model for which you want to make changes, in the lower pane select the element, right-click, and select **Properties** from the pop-up menu. See “Adding Elements to a Reference Model” on page 176 for details of the properties of an element.

Note: The list of elements for a reference model indicates inherited elements by means of an up-arrow icon. You can modify an element that is inherited, but you must be aware that the changes you make affect not only the reference model you have selected, but also the parent model where the element is defined and any other child models that inherit the element. To

modify only the selected reference model you must re-insert the same inherited element assigning a new desired state. Because the element is then a new element, the parent model is not affected.

- Assign a new subscriber to a reference model.
See “Assigning Subscribers to a Reference Model” on page 180.
- Remove a subscriber.
To do this, select the reference model you want to remove a subscriber from, in the lower pane select the subscriber, right-click, and select **Remove** from the pop-up menu.
- Modify a subscriber, for example to define a different query.
To do this, select the reference model for which you want to make changes, in the lower pane select the subscriber, right-click, and select **Properties** from the pop-up menu. See “Assigning Subscribers to a Reference Model” on page 180 for details of the properties of an subscriber.

Note: You can restore a reference model from the latest saved version. From the **File** menu, select **Restore**; the reference model is restored to the last saved version.

Change Manager and Activity Plans

Change Manager includes two functions that use a differencing mechanism to produce a list of the actions required to bring subscribers to the desired state defined in a reference model. These functions are **Preview Activity Plan** and **Submit Activity Plan**.

When you preview or submit an activity plan, the differencing mechanism creates an activity plan listing all the actions requested to move the configuration elements to their desired state. The **Preview** function simply displays the activity plan in a window so you can preview the changes that are going to take place. The **Submit** function allows you to set other Activity Plan Manager (APM) specific parameters before it submits the plan to APM for processing.

By default Change Manager synchronizes a reference model to create the associated activity plan by considering all the configuration elements specified in the model and creating the actions related to those elements. However, the full synchronization feature allows you to perform a further step in the synchronization process. This option is available when you preview an activity plan as described in “Previewing an Activity Plan” on page 189, or submit an activity plan as described in “Submitting an Activity Plan” on page 191. Though this option actually applies only to Software Distribution elements, it is general enough to be implemented for every plug-able configuration element.

In general, Change Manager synchronization creates the requested actions related to the listed elements as in the default behavior. When it does this, it also creates a new set of actions related to all the elements already present on the selected subscribers though not listed in the current reference model, in order to “revert” the state of such elements.

In the case of Software Distribution elements, “reverting” means to uninstall, or to remove the software element, or package, from the target machine. This does not necessarily mean that such actions will all involve actual “remove” actions. The required action will depend on the actual state of the package on the target machine. Thus, for a Software Distribution element, when you select the full

synchronization option, Change Manager creates a set of actions to remove from the subscribers all the software packages not listed in the reference model being synchronized.

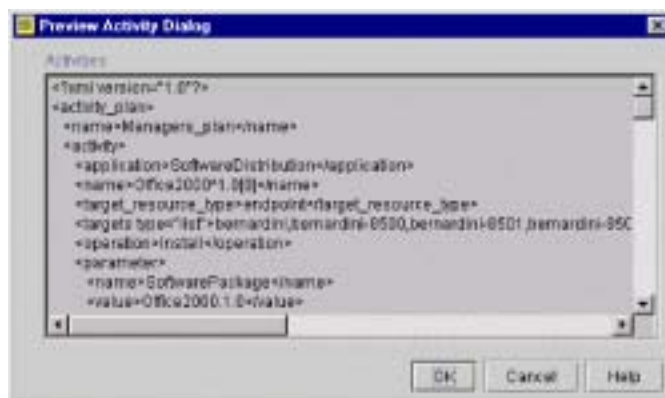
You should preview the activity plan before submitting it to the Activity Planner.

Note: If you are using Web User Interface (WebUI) 4.1 and your model has only Web Interface subscribers, you still need to submit an activity plan to update the Change Manager database; even though the activity plan is empty. In this case, there is no need to preview the plan.

Previewing an Activity Plan

To preview an activity plan, perform the following steps:

1. Select the required reference model and right-click to display the pop-up menu.
2. Select **Preview actions** to display the Preview Activity dialog box:



After selecting **Preview actions**, you are asked to specify whether or not you want the preview performed with full synchronization. If you specify Full synchronization, the activity plan preview will include all activities required for full compliance with the reference model, including such activities as removing software from the targeted subscribers if necessary.

3. You can now return to the reference model to correct any errors, before submitting the activity plan, see “Submitting an Activity Plan” on page 191.

Assigning a Priority to Configuration Elements

Before you synchronize a reference model, you might want to set a priority for the actions, or configuration elements, required to bring the target machine to the desired state. The actions are performed in the order you specify. By default, no priority is defined for the actions when they are generated.

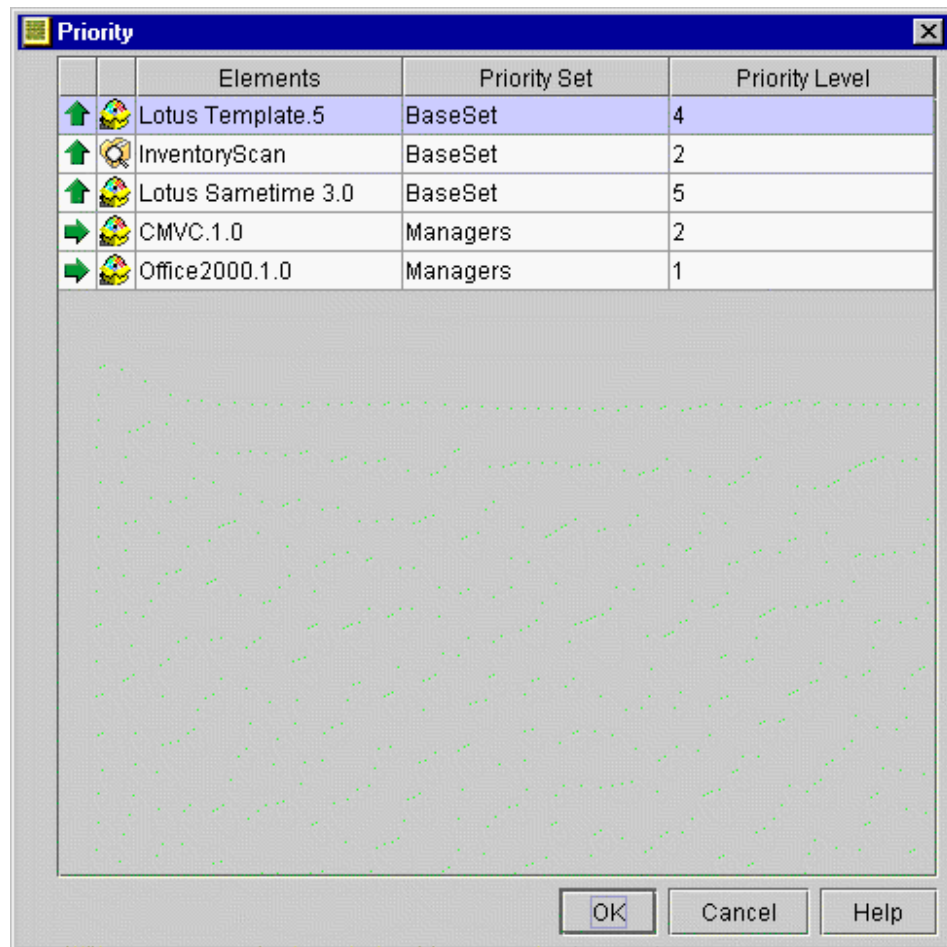
To define a priority, you must first define the priority set the configuration elements belong to. Different priority sets are separate sets of configuration elements with no correlation. Two configuration elements belonging to two different priority sets cannot be dependent on each other.

If you define a priority set for a Pristine Manager element, all other configuration elements present in the reference model must be part of the same priority set and must have a higher priority than the Pristine Manager element. In this way, all other configuration elements are conditioned to the Pristine Manager element, which must always be installed first.

Change Manager and Activity Plans

To define a priority for the configuration elements in the reference model, perform the following steps:

1. Select **Assign Priority** in the Edit menu. The Assign Priority dialog is displayed.



2. In the **Priority Set** column, click the text field next to the specified element and type a new priority set name, or select an existing name from the pull-down menu.
3. In the **Priority Level** column, define the priority for the specified configuration element within its priority set. The default value is zero, which means no priority is specified.
4. When you have finished, click **OK** to save your changes.

In the figure displayed above, the InventoryScan, Lotus Template5.0, and Lotus Sametime3.0 elements belong to the BaseSet priority set, and CMVC.1.0 and Office2000.1.0 belong to the Managers priority set.

As determined by the priority value defined in the **Priority** dialog box, the actions in the BaseSet priority set are performed in the following order:

1. InventoryScan
2. Lotus Template5.0
3. Lotus Sametime3.0

and the elements in the Managers priority set are performed in the following order:

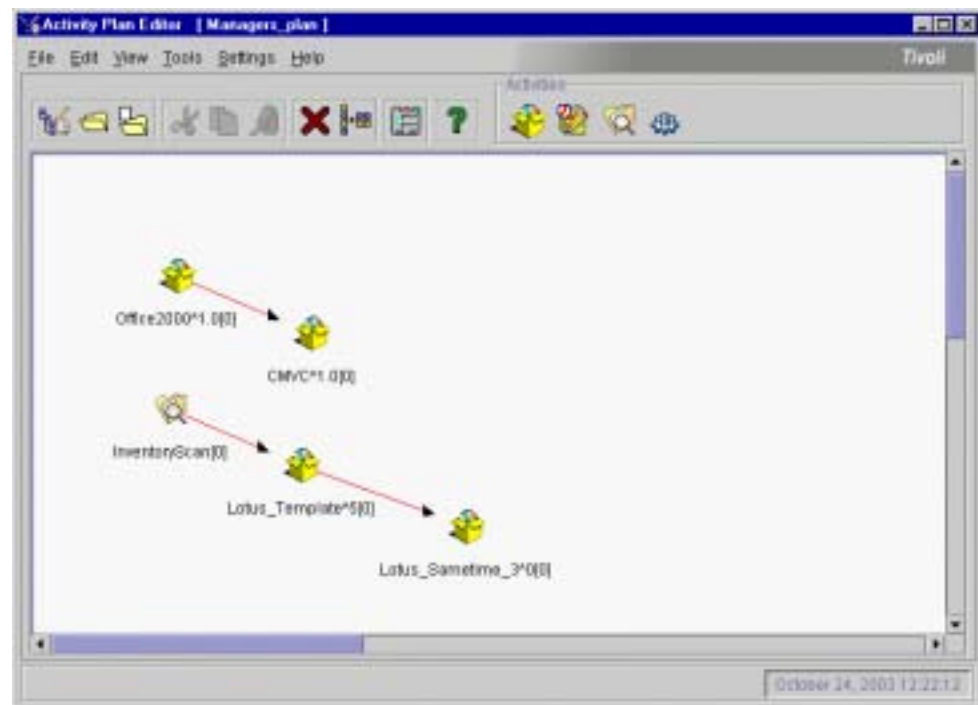
1. Office2000.1.0

2. CMVC.1.0

The two priority sets are independent and are executed in the order they are received from Activity Planner. In Activity Planner, the following conditioning type is applied to render the priority defined in the reference model:

- ST (success target). This conditioning applies to all Software Distribution operations, provided they do not have pristine targets.
- SA (success all). This conditioning applies to all configuration elements and to Software Distribution operations with pristine targets.

The following picture displays the plan created in Activity Planner based on the priority defined in the reference model.



Submitting an Activity Plan

To submit an activity plan, perform the following steps:

1. Select the required reference model and right-click to display the pop-up menu.
2. Select **Submit activity plan** to display the Select activity plan name dialog box.



3. On the Select plan name dialog box, enter the plan name in the Plan name field. Select the **Full synchronization** check box to specify that you want the reference model to be completely implemented on the target, as described in

Change Manager and Activity Plans

“Change Manager and Activity Plans” on page 188. Select the **Start not before** check box to specify the start date and time for the activity plan implementation.

4. Click **OK** to display the Submit dialog box:



5. On the Submit dialog box, select one of the submission options:
 - **Submit activity plan** to submit the plan to the Activity Planner.
 - **Import activity plan** to send the plan to the Activity Planner where it can be scheduled later.
6. If you selected **Submit activity plan**, select or clear the **Delete activity plan after submission** check box. If you select **Delete activity plan after submission**, the activity plan will be deleted from the Activity Plan Manager database.
7. If you selected **Import activity plan**, select or clear the **Overwrite activity plan** check box.

If you selected **Submit activity plan**, this check box cannot be cleared. This is to prevent the submission failing because of a duplicate activity plan name in the Activity Planner database.
8. Click **OK**.

If any of the subscribers you specify are not valid, the operation fails on those subscribers, and continues on the other subscribers. Information about the subscribers on which the operation did not complete is written to the ccmlog0 file in the Change Manager trace files directory.

Creating Change Manager Reports

Change Manager provides reports that provide you with an overview of activity plans related to reference models and targets. The following reports are available:

Reference Model report

This report summarizes progress for all targets that are subscribers to a specified reference model. The model you specify need not be currently displayed.

Target report

This report summarizes progress for all reference models to which a specified target is a subscriber.

Both versions of the report include the following information:

- Target name.
- Type of subscriber, for example Profile Manager.
- Reference model name.
- Activity plan ID.
- Status.
Possible values are **Submitted**, **Successful** and **Failed**.
- Date and time the activity plan was submitted.
- Date and time the result was received.
- Synchronization ID returned by the engine when a synchronization operation is submitted. Use this ID to match the reference model and the related activity plan.

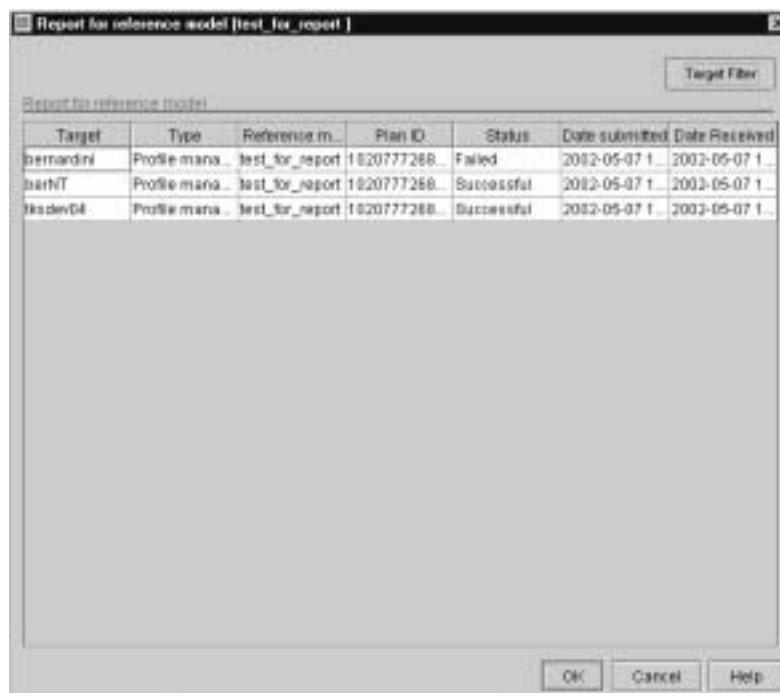
Note: If you have a large number of endpoints (50,000 or more), using Change Manager reports can adversely affect plan execution time. In this case, only use Change Manager reports if really necessary, for example, if you need to collate Change Manager integration or historical information. Otherwise, use other Configuration Manager reports, such as the Activity Plan Monitor, MDist2, and log files.

Reference Model Report

To display the Reference Model report, perform the following steps:

1. In the upper pane of the Change Manager window, select the reference model for which you want a report. From the **Report** menu, select **Report for Ref. Model**. The Report for reference model dialog box appears.

Creating Change Manager Reports



2. The Report for reference model dialog box displays the records for the selected reference model. You can use the **Target filter** button to display the Target filter dialog box and limit the targets listed in the reference model report.



On the Target Filter dialog box, either select the **All targets** radio button, or select the **Select the maximum number of targets to show** radio button and specify a number in the **Maximum targets number** field.

3. Click **OK**.

Target Report

To display the Target report, perform the following steps:

1. From the **Report** menu, select **Report for Target**. The Select target dialog box appears.



2. Specify the name and subscriber type of the targets for which you want to produce a report. If you selected the endpoint subscriber type, you can browse the endpoints. If you selected the device type, the navigator is disabled since it is not possible to navigate the individual device instances. If you selected the user type, the navigator displays the association between a user and the related endpoint.
3. Click **OK**.

Chapter 7. Using the Command Line

This chapter describes the Change Manager component of IBM Tivoli Configuration Manager (Change Manager) command line interface (CLI).

The sections of this chapter include:

- Managing Reference Models
- Command Line Tasks

The CLI interface is a textual interface that you use to manage reference models. A reference model provides a profile of a desired configuration to which targets (workstations and devices) can subscribe.

For more information about reference models, see “Reference Models” on page 161 and “Creating the Reference Model Structure” on page 172.

Managing Reference Models

The Change Manager feature provides a command line interface to manage and control reference models. The commands allow you to perform the following tasks:

- Import a reference model to the database
- Export a reference model from the database
- Synchronize targets to a given reference model
- Generate reports
- List models
- Remove a reference model from the database
- Handle trace information dynamically

You can specify a timeout for CLI commands in Change Manager using the environment variable `_CCM_CLI_TIMEOUT_=<seconds>`. This variable specifies how many seconds the command line must wait for a response to a Change Manager engine request. This is useful when you are using the **wsyncrmod** command to synchronize a reference model, because you can specify a timeout that gives **wsyncrmod** enough time to successfully complete. You can set the `_CCM_CLI_TIMEOUT_` variable in the shell or in the `setup_env` file. The default value is 180 seconds.

To request help from the command line about a particular command, enter the command name followed by a question mark (?). For example, for information about the **wexprmod** command, enter `wexprmod ?`.

The commands shown in Table 46 are used to manage reference models.

Table 46. Commands for managing reference models

Command	Purpose	Details on page
wccmgui	Starts the Change Manager graphical user interface from the CLI.	197
wccmplugin	Registering available plug-ins.	198

Managing Reference Models

Table 46. Commands for managing reference models (continued)

Command	Purpose	Details on page
wdelrmod	Removing a reference model from the Change Manager database.	200
wexprmod	Exporting a reference model.	202
wimprmod	Importing a reference model.	204
	Getting report information for a selected reference model.	208
wlstrmod	Listing all reference models stored in the Change Manager database.	206
wsyncrmod	Synchronizing on a selected reference model.	210
wtrcccm	Dynamically traces reference model activity.	217

Full details of the commands are as follows:

wccmgui

Starts the Change Manager graphical user interface from the CLI.

Note: In a Windows environment the user who starts the GUIs must belong to the Tivoli_Admin_Privileges group.

Syntax

wccmgui

Description

The **wccmgui** starts the Change Manager graphical user interface from the CLI. Because this command is a bash script it must be run in the bash environment on a Windows machine. Precede the command with the string **sh**.

Options

None.

Authorization

view, and any other authority appropriate for the tasks you want to perform with the Change Manager GUI.

Return Values

The **wccmgui** command returns one of the following values:

0 Indicates that **wccmgui** started successfully.

Nonzero

Indicates that **wccmgui** failed due to an error. Ensure that the **wccmgui** script is in the \$(BINDIR)/bin folder. Also, in a Windows environment, be sure to precede the command with the string **sh**.

Examples

1. To start the Change Manager graphical user interface:
`wccmgui`
2. To start the Change Manager graphical user interface using a Windows machine, enter the following command:
`sh wccmgui`

See Also

None

wccmplugin

Registers available plug-ins.

Syntax

wccmplugin {-a | -s } *plugin_name* [-f *config_xml_file*] | -r *plugin_name* /-l

Description

The **wccmplugin** command allows you to register, modify, or remove a plug-in in the database.

Options

a *plugin_name*

The option to register a new plug-in named <plugin_name>.

s *plugin_name*

The option to modify a previously registered plug-in named <plugin_name>.

f *config_xml_file*

The option to specify the xml file containing the plug-in description.

r *plugin_name*

The option to remove a registered plug-in named <plugin_name>.

l

The option to list all the registered plug-ins.

Authorization

One of the following: CCM_Admin or install_product

Return Values

The **wccmplugin** command returns one of the following values:

- 1 Indicates that the usage returned.
- 0 Indicates that the command completed successfully.
- 1 Indicates that insufficient roles were detected.
- 2 Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3 Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4 Indicates a failure in committing transactions.
- 5 Error reading file. An error was detected while reading the xml file containing the model to import. Verify the path and filename.
- 6 Indicates a failure creating the files. The specified path does not exist.
- 9 Indicates that generic error occurred while processing the command.
- 42 Indicates that the plug-in was not found. The specified plug-in cannot be found in the database.
- 43 Indicates that the plug-in is already registered. The specified plug-in is already registered in the database.

Examples

To register the Software Distribution plug-in:

```
wccmplugin -a SoftwareDistribution -f $(BINDIR) /TME/CCM/GUI/swd_plugin.xml
```

To register the Inventory plug-in:

```
wccmplugin -a InventoryScan -f $(BINDIR)/TME/CCM/GUI/invscan_plugin.xml
```

See Also

None

wdelrmod

Removes a reference model from the database.

Syntax

```
wdelrmod [-f ] -n modelName [-v modelVersion] [-P parentRootName [-V parentRootVersion]]
```

Description

The **wdelrmod** command removes a reference model from the database.

Options

f The force option to remove a reference model with all its children models.

n *modelName*

The selected reference model name. Using this option lists all the actions performed on this reference model's subscribers.

v *modelVersion*

The selected reference model version to uniquely identify the reference mode.

P *parentRootName*

The root model name.

V *parentRootVersion*

The root model version.

Authorization

manage role.

Return Values

The **wdelrmod** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 9** Indicates that a generic error occurred while processing the command.
- 30** No reference model found. The database is empty.
- 31** Model not found. The reference model was not found inside the database.
- 34** No removal performed. No reference model was removed from the database. Check that the information you supplied is correct.
- 36** No force option. The addressed reference model contains children models. To force the whole sub tree removal you need to specify a force option.
- 46** Root not unique. The specified root is not uniquely identified: try to supply a version number.

Examples

To remove the DevelopersTree reference model belonging to the RootNode^1.0 tree, including all its children models:

```
wdelrmod -f -n DevelopersTree -P RootNode -v 1.0
```

See Also

None

wexprmod

Exports a reference model in xml format.

Syntax

```
wexprmod -n modelName [-v modelVersion] [-P parentRootName [-V
parentRootVersion]] [[-o] -f xml_file_name] [-i] [-s]
```

Description

The **wexprmod** command exports a reference model from the Change Manager database in xml format.

Options

n *modelName*

The name of the reference model to export.

v *modelVersion*

The parent reference model version.

P *parentRootName*

The root model name.

V *parentRootVersion*

The root model version.

o

The overwrite option if the destination file already exists.

f *xml_file_name*

The full path to the xml format file containing the reference model structure.

i

A flag to include the exported model along with the inherited from the upper model in the hierarchy.

s

A flag to specify the exported model does not contain child models; it is a single model export.

Authorization

view role.

Return Values

The **wexprmod** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 6** Error writing file. An error was detected while exporting the model inside the target xml file.
- 7** Overwrite required. The destination file already exists. The overwrite option is required.
- 9** Indicates that generic error occurred while processing the command.

- 30** No reference model found. The database is empty.
- 31** Model not found. The reference model was not found inside the database.
- 33** Model not unique. The specified model is not uniquely identified: try to specify either a root node or a version number.
- 46** Root not unique. The specified root is not uniquely identified: try to supply a version number.

Examples

To export the `JavaNode^1.0` reference model belonging to the `DevelopersTree` hierarchy including inherited elements:

```
wexprmod -n JavaNode -v 1.0 -P DevelopersTree -o -f $ (TMP_DIR) /export.xml -i
```

See Also

None

wimprmod

Imports a reference model in xml format into the Change Manager database.

Syntax

wimprmod [-o] -f *xml_file_name* [-e *encoding*] [-p *parentModelName* [-v *parentModelVersion*] [-P *parentRootName* [-V *parentRootVersion*]]]

Description

The **wimprmod** imports a reference model in xml format into the Change Manager database.

Options

- o** A flag to force the replacement of a model already present as a child of the selected parent node.

f *xml_file_name*

The full path to the xml format file containing the reference model structure.

e *encoding*

The name of the encoding type that applies for this XML file. When importing an XML file, you can choose to modify its encoding type. The encoding type specifies the code set to use to process the input file. For a complete list of supported encoding types, see the Sun Microsystems Web Site.

p *parent_Name*

The parent reference model name.

v *parent_Version*

The parent reference model version. If no parent model is specified, the imported reference model is added as a new root model.

P *parentRootName*

The root model name.

V *parentRootVersion*

The root model version.

Authorization

manage role.

Return Values

The **wimprmod** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 5** Error reading file. An error was detected while reading the xml file containing the model to import.

- 9 Indicates that generic error occurred while processing the command.
- 31 Model not found. The reference model specified as parent node was not found inside the database.
- 32 Parent node specified is not unique. The reference model specified as parent node is not uniquely identified. The version number must be supplied.
- 33 Model not unique. The specified model is not uniquely identified: try to specify either a root node or a version number.
- 37 No overwrite option specified. A reference model having the same name and version of the imported model already exists in the database as child node of the specified parent model and no overwrite option was specified.
- 38 Model already present. The imported reference model is already stored in the database with a different parent node.
- 45 No instance found for some of the model configuration components.
- 46 Root not unique. The specified root is not uniquely identified: try to supply a version number.

Examples

To import a model inside the DevelopersTree model belonging to the RootNode^1.0 tree:

```
wimprmod -o -f $(TMP_DIR) /export.xml -p DevelopersTree -P RootNode -V 1.0
```

See Also

None

wlstrmod

Lists all reference models in the database.

Syntax

wlstrmod [-a [-l] [-w]]

wlstrmod -t {-a | -n *name* [-v *version*] [-P *parentrootname* [-V *parentRootVersion*]]}

wlstrmod -n *name* [-v *version*] [-P *parentrootname* [-V *parentRootVersion*]] [-l | -s | -e [-i]] [-w]

Description

The **wlstrmod** command lists all reference models in the database by reference model name and version number.

Options

- t An option to display the tree structure.
- a An option to list all available reference models inside the database.
- l An option to specify that the following information is also listed:
 - model description
 - last-update date and time
 - is a leaf reference model
 - is a root reference model
- w An option to specify that output be formatted to screen size.
- s An option to list the subscribers of a specific model.
- e An option to list the configuration elements of a specific model.
 - i An option to display the reference model's inherited elements. This option can be used only with the -e option.
- n *modelName*
The selected reference model name. If no reference model is specified, this option returns all the reference models available in the database.
- v *modelVersion*
The selected reference model version.

Authorization

view role.

Return Values

The **wlstrmod** command returns one of the following values:

- 1 Indicates that the usage returned.
- 0 Indicates that the command completed successfully.
- 1 Indicates that insufficient roles were detected.
- 2 Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3 Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4 Indicates a failure in committing transactions.

- 9** Indicates that generic error occurred while processing the command.
- 30** No reference model found. The database is empty.
- 31** Model not found. The reference model was not found inside the database.

Examples

To list the models in the DevelopersTree in a tree structure:

```
wlstrmod -t -n DevelopersTree
```

See Also

None

wlstsrep

Gets the reporting information for a selected reference model.

Syntax

```
wlstsrep [-l ] [-w]] {-a | -n modelName [-v modelVersion] [-m] | -t targetName }
```

Description

The **wlstsrep** command displays information about the following record entries:

- Model name and version
- Target name and type
- Plan ID
- Plan status
- Synchronization ID

Options

- l** The option to specify that target type, submit time, and receive time, and synchronization ID be listed.
- w** The option to specify that output be formatted to screen size.
- a** A flag to list all the available reports.
- n *modelName***
 The selected reference model name. Using this option lists all the actions performed on this reference model's subscribers.
- v *modelVersion***
 The selected reference model version to uniquely identify the reference model.
- m** Specifies that the search for the report must return an exact match.
- t *targetName***
 The selected target. Using this option lists all the actions performed on the reference models to which this target is subscribed.

Authorization

view role.

Return Values

The **wlstsrep** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 9** Indicates that generic error occurred while processing the command.
- 39** No report info. No report information is available inside the database.

- 40** No report info for target. No report information is available inside the database for the specified target.
- 41** No report info for model. No report information is available inside the database for the specified reference model.

Examples

To list the report information related to model `JavaNode^1.0`

```
wlstprep -l -n JavaNode -v 1.0
```

See Also

None

wsyncrmod

Performs a synchronization on a selected reference model. This command can be used in preview-only mode when you do not have the proper role to perform a synchronization, or in the full-function mode if you do have the proper role to perform a synchronization.

Syntax

The full- function mode syntax follows:

```
wsyncrmod [ -f xml_file_name] | [-n planName] [-s yyyy-MM-dd HH:mm] [ -x
in_xml_file_name] [{-d] | [-i [-o]]}] [-F] -m modelName [-v modelVersion] [-P
parentRootName [-V parentRootVersion]]
```

The preview-only mode syntax follows:

```
wsyncrmod -p -o [ -f xml_file_name] [-F] -m modelName [-v modelVersion] [-P
parentRootName [-V parentRootVersion]]
```

Description

The **wsyncrmod** command synchronizes a reference model. The reference model can be taken either from the Change Manager database or from an xml file.

If any of the subscribers specified in the reference model are not valid, the operation fails on those subscribers, and continues on the other subscribers. Information about the subscribers on which the operation did not complete is written to the ccmlog0 file in the Change Manager trace files directory.

When a synchronization operation is performed, a synchronization ID is returned. Use this ID to match the reference model with the plan created by Activity Planner. To perform this operation, see “Creating Change Manager Reports” on page 192.

Options

- p** The preview option to specify that no action is performed against Activity Planner; the result will be displayed on the current screen or can be redirected to a file using the **-f** option.
- f** *xml_file_name*
The full path to the file containing the preview output.
- n** *planName*
The option to specify an activity plan name different from the default one.
- s** The option to specify a value for the “start not before” activity plan attribute.
- x** The option to specify an input file containing the reference model to be synchronized.
- d** The delete-after-submission option to specify that the related plan has to be deleted from the database once it has been submitted.
- i** The import option specifying that no submit action will be performed. The action plan generated will be imported into the Activity Planner database and can be scheduled for future submission.
- o** The overwrite option if a plan with the same name already exists inside the Activity Planner database. This option is used by default when the plan is submitted immediately to Activity Planner.

- F** The option to specify full synchronization. This means that the reference model will be fully implemented on the targets even if it means that some software must be removed in order to conform to the model.
- m** *modelName*
 The name of the reference model to synchronize.
- v** *modelVersion*
 The version of the reference model to synchronize.
- P** *parentRootName*
 The root model name.
- V** *parentRootVersion*
 The root model version.

Authorization

The following authorization roles are required depending on the mode you are using the command:

- edit role for full-function mode
- view role for preview-only mode

Return Values

The **wsyncrmod** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for Activity Planner: the Activity Planner object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 9** Indicates that generic error occurred while processing the command.
- 31** Model not found. The reference model specified for the synchronization was not found inside the database.
- 33** Model not unique. The reference model specified for the synchronization was not uniquely identified. Supply the version information.
- 45** No instance found for some of the model configuration components.
- 46** Root not unique. The specified root is not uniquely identified: try to supply a version number.
- 51** No action required. The current configuration does not require any action to be performed.

Examples

1. To synchronize the `JavaNode^1.0` reference model belonging to the `DevelopersTree` hierarchy:

```
wsyncrmod -n test_plan -d -m JavaNode -v 1.0 -P DevelopersTree
```
2. To preview the actions related to the `JavaNode^1.0` reference model belonging to the `DevelopersTree` hierarchy:

```
wsyncrmod -p -m JavaNode -v 1.0 -P DevelopersTree
```

wsyncrmod

Chapter 8. Troubleshooting

This section gives an overview of the troubleshooting process and provides descriptions of sources of information that will help you in solving problems with Change Manager operations. It includes the following topics:

- “Checking the Change Manager Configuration File” on page 215
- “Change Manager Logs”
- “Change Manager Traces”
- “Change Manager GUI Trace” on page 216
- “Managing Different Domains” on page 219
- “Distributing to a Large Number of Targets” on page 219

Change Manager Logs

The Change Manager core log is called **ccm.log**. When the log is activated, it can be found in one of the following locations:

Windows operating systems

%SystemDrive%

Unix operating systems

/tmp

The log traces the history of all calls made to the IDL interface, for example operations performed by the C code of the CCM_core application when other applications interact with Change Manager.

The log can be activated by adding the environment variable **__CCM_DEBUG__**, using the **odadmin environ set** command, or as a system variable.

In addition, Change Manager includes the **ccmlog** log. This log contains the key history of Change Manager operations. The technical details of Change Manager operations are contained in trace files. For information about trace files, see “Change Manager Traces.”

The **ccmlog** log is controlled by the <log_data> stanza in the Change Manager configuration file, **confccm.xml**. For information about the **confccm.xml** file, see “The Change Manager Configuration File **confccm.xml**” on page 153.

Each line in the **ccmlog** file describes a single operation and provides the following information:

- Date and time the operation occurred.
- Message code.
- Description of the operation being performed.

Change Manager Traces

Change Manager provides traces to record information about Change Manager operations. You can set the trace level to determine the level of detail recorded for each operation. Trace level values are numbers between 0 and 5, with the default being 0, with no logs or traces set.

Change Manager Traces

To set the trace level for command line operations, use the `wtrccm` command. If you change the trace setting in the command line on the Tivoli server, the update is recorded in the GUI after 60 seconds.

To set the trace level for GUI operations, choose **Settings » Trace Level** in the Change Manager GUI. For information about trace levels, see “`wtrccm`” on page 217. If you change the trace setting in the GUI on the Tivoli server, the update is recorded in the command line after 60 seconds.

Change Manager has the following available traces:

- `ccmxxx.trc`
- `ccm_apmxxx.trc`
- `ccm_webxxx.trc`
- `ccm_clixxx.trc`

The main Change Manager trace file is **`ccmxxx.trc`**.

It traces the history of all operations performed by the Change Manager code, both for the GUI and for the engine. For example, it records trace information for the following:

- Managing a reference model and its components.
- Performing persistence operations.
- Synchronizing the reference model.
- Accessing the reference model and target reports.

The following keys in the `trace_data` stanza of the `confccm.xml` file control the main Change Manager trace, `ccmxxx.trc`:

trace_level	The trace file is only created if this is set to greater than zero (default is zero). Trace values are as follows: <ul style="list-style-type: none">• 0 (none)• 1 (fatal)• 2 (error)• 3 (warning)• 4 (info)• 5 (verbose)
trace_size	Maximum number of records that can be written to the file (default is 1 000 000)

Other Change Manager traces are described below:

- **`ccm_apmxxx.trc`**
Contains the history of all operations performed by the Change Manager Java code of the CCM_core application when other applications interact with Change Manager. It records trace information for the following Change Manager operations:
 - A submit action for the Activity Planner.
 - The response to a Web UI request.
 - A synchronization operation for pristine systems.
- **`ccm_webxxx.trc`**
Contains the history of all operations performed by the Change Manager engine when interacting with the new Web UI 4.2 on the Application server.
- **`ccm_clixxx.trc`**

Contains the history of all the operations performed using the CM command line as well as the operations performed when Change Manager interacts with the Activity Planner to download the plug-ins information.

Checking the Change Manager Configuration File

The Change Manager is configured using the `confccm.xml` file, which can be found in one of the following locations:

Windows

`%SystemDrive%`

Unix

`/etc/Tivoli`

Note: For information about how the `confccm.xml` file is backed-up when upgrading from Change Configuration Manager 4.1 to Change Manager 4.2 or to Configuration Manager Version 4.3.1, see “The Change Manager Configuration File `confccm.xml`” on page 153.

The file is organized in stanzas, each one of which is responsible for configuring a different aspect of Change Manager. For an example of the `confccm.xml` file, see “The Change Manager Configuration File `confccm.xml`” on page 153. The `confccm.xml` file contains information about the following:

- Persistence

The RIM name is stored inside the `<url>` tag of the `<persistence data>` stanza. The default RIM name is `ccm`.

- Trace

You can customize the following values:

- `trace_level` : [0...5]
 - 0 (none)
 - 1 (fatal)
 - 2 (error)
 - 3 (warning)
 - 4 (info)
 - 5 (verbose, default)
- `trace_size` : default is 1000000
- `working_dir` : default `$BINDIR\..\ccm`

- Logs

You can customize the following values:

- `log_level` : {0...5}
 - 0 (none, default)
 - 1 (fatal)
 - 2 (error)
 - 3 (warning)
 - 4 (info)
 - 5 (verbose)
- `logfile_size` : default is 1000000
- `logfile_name` : default is `ccmlog`
- `log_working_dir` : default `$BINDIR\..\ccm`

- Plug-in download enablement

Using the appropriate stanza in the `confccm.xml` file, you can specify whether Change Manager should use the local plug-in information or download it from

Checking the Change Manager Configuration File

the server when the GUI starts. If you want to use the server plug-in information, you must enable the server tags on both the client, or GUI, and on the server because the download mechanism is bi-directional. When you enable the download mechanism on the client, it means Change Manager can send local plug-in information stored in the confccm.xml file to the server. When you enable the download mechanism on the server, the Change Manager engine can compare plug-in information received from the client with plug-in information stored in the Change Manager database and update this information on the client.

You can enable or disable the download mechanism by using the `<plugin_download>` stanza of the confccm.xml file.

For more information about downloading plug-ins, see “The Change Manager Configuration File confccm.xml” on page 153.

Change Manager GUI Trace

The Change Manager GUI trace file is called **ccmxxx.trc**. It is controlled by the same keys as the main trace file. It traces all actions performed by the Change Manager GUI. It is written on either Server or Managed Node, whenever the Change Manager GUI is opened.

wtrcccm

Manages trace options.

Syntax

```
wtrcccm { -a | -d | -v | -q key | -s key=value }
```

Description

The **wtrcccm** command enables or disables dynamic trace and log handling.

Options

- a** A flag to enable both traces and logs (level 5).
- d** A flag to disable both traces and logs (level 0).
- v** A flag to view all the available keys with the current values:
 - trace_level;

Note: If a wrong value is inserted, this key is set to 2 by default

 - trace_size;
 - working_dir;
 - log_level;
 - log_size;
 - log_file_name;
 - log_work_dir
- q** A flag to get the current value for a specific key.
- s** A flag to set a particular key/value pair.

Authorization

view role.

Return Values

The **wtrcccm** command returns one of the following values:

- 1** Indicates that the usage returned.
- 0** Indicates that the command completed successfully.
- 1** Indicates that insufficient roles were detected.
- 2** Indicates that the command was unable to look-up for APM: the APM object was not found.
- 3** Indicates that the timeout expired: the timeout fixed for the command execution has expired: the communication channel has been closed.
- 4** Indicates a failure in committing transactions.
- 9** Indicates that a generic error occurred while processing the command.
- 18** The supplied level is not allowed.
- 44** Invalid key: an invalid key has been selected either for setting or for viewing.
- 48** Invalid level: The supplied trace or log level is not valid.

wtrcccm

Examples

To set the trace level to 4:

```
wtrcccm -s trace_level=4
```

See Also

None

Managing Different Domains

To start the Change Manager on a managed node if the managed node is installed on a different network domain from the Tivoli Management region domain, you start the Change Manager in one of the following ways:

- Use the `wccmogui` command.
- On the Tivoli management region, specify the domain where the managed node resides. To specify a new domain on a UNIX Tivoli management region, add an entry to the `/etc/resolv.conf` file. To specify a new domain on a Windows Tivoli Management region, add the required DNS to the DNS list for the TCP/IP settings.

Distributing to a Large Number of Targets

When you distribute a reference model to a very large number of targets, for example to 20,000 targets, the versioning and dependency checking operations performed by Software Distribution can take a long time to complete.

In this case, you can index the following columns in the `COMPUTER` table stored in the Inventory database:

- `TME_OBJECT_ID`
- `COMPUTER_SYS_ID`

Perform the indexing only when the content of the database is stable, and frequent scans are no longer required, because it can slow down insert and update operations. For more information on the Inventory database, refer to *IBM Tivoli Configuration Manager: Database Schema Reference*.

Specific Problems and Workarounds

The following list details some common problems and workarounds for Change Manager:

The Change Manager GUI does not start when launched from the Tivoli desktop

Check whether the `odadmin odlist` command returns the fully qualified hostname. If the fully qualified hostname is not returned, add it to the aliases list using the `odadmin add_hostname_alias` command. For more information on the `odadmin odlist` command, refer to *Tivoli Management Framework: Reference Manual*.

Specific Problems and Workarounds

Part 3. Installing an Operating System on Pristine Machines

Chapter 9. Overview: Installing an Operating System on Pristine Machines	223
Advantages of Using Pristine Manager	223
Pristine Manager Concepts	223
Prerequisites.	224
Configuring Pristine Manager	224
Enabling Pristine Manager Databases	224
Configuring Pristine Manager for Tivoli Enterprise Console	226
Enabling Integration with the Tivoli Enterprise Console	226
The tecad_pristine.conf File.	227
Creating the Pristine Manager Console	229
Pristine Manager Classes	230
Installing on Pristine Machines in Interconnected Regions	230
Chapter 10. Installing on Pristine Machines	233
Setup Tasks	233
Defining Servers	233
Defining Machines	236
Filtering Machines in the List	242
Adding Machines to a Group	243
Updating Multiple Machines	243
Creating Groups	245
Creating an Operating System Element.	248
Defining Environment Variables	252
Defining Your Own Environment Variables	254
Modifying the Installation Job for ADS Servers	255
Running Commands on Pristine Machines	256
Preparing and Submitting a Plan	256
Using Activity Planner	256
Using Change Manager	257
Chapter 11. Using the Command Line	259
woselement	260
wpristine	262
wpristineexport	264
wpristinegroup	265
wpristinemachine	267
wpristinesrv	270
Chapter 12. Troubleshooting	273
Gathering Trace Information	273
PRISTINE_TMA_SETUP_PATH and Sharing Access to Endpoint Setup Files	274
Installing from a RIS Server without Sharing the Folder.	274
Installing from an ADS Server without Sharing the Folder	274
Adding Software Distribution or Inventory Activities to a Plan	276
Configuration Changes after Installing Pristine Manager	277
Failure of Installation.	277
Avoiding Failure	277
ADS Certificate.	277
Installing in Interconnected Regions.	277
Installing Microsoft Windows 2003 from an ADS Server.	278
Installation from RIS Server Successful but Activity Plan Fails.	278
Specifying a Static IP Address	278

This part describes how to install an operating system on pristine machines using the Pristine Manager.

Locating Related Information

Information related to this tool is available from the following sources:

- *Planning and Installation Guide*. This includes information about how you install the service.
- *Messages and Codes*. This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 9. Overview: Installing an Operating System on Pristine Machines

In your environment, the Microsoft Automated Deployment Services (ADS) or Microsoft Remote Installation Services (RIS) servers are configured to install images on machines. Pristine Manager integrates these capabilities with Tivoli functions. Pristine Manager uses the images that are on your RIS and ADS servers to install them on the target machines. At the same time, Pristine Manager includes the machines in the Tivoli environment as endpoints: it takes the label that you define and makes it the endpoint label of the new machine. This enables Change Manager and Activity Plan Monitor to work with the machines as if they are Tivoli endpoints after the pristine installation.

In Pristine Manager, you define the servers that have the images and the pristine machines on which they will be installed. You specify the operating system element and the targets. This creates a connection among the operating system to be installed, the target pristine machines, and the servers. Then, this information is used to create an activity plan that is used by Activity Planner to do the pristine installation.

Advantages of Using Pristine Manager

The advantages of using the tool are as follows:

- You can perform remote, unattended pristine installations without having to use a boot diskette on site.
- You can also install on machines that are already configured, for example, to reinstall the operating system or a new operating system from scratch.
- A Tivoli endpoint is automatically installed on the target machine.

Pristine Manager Concepts

Many of the concepts used in Pristine Manager are derived from Change Manager.

machine manager

Lists the machines on which the operating system is installed. It contains the server that has the images for each machine. You can group the machines and specify details for the Tivoli endpoint that is also installed on the target machine.

pristine server

An ADS or RIS server that contains the operating system images. It is also a Tivoli endpoint.

server manager

Lists pristine servers.

operating system element

Contains images that are on the pristine servers. It can contain only one image per server.

pristine target subscriber

A subscriber type that can be included in a reference model. Like the static subscriber for other Change Manager configuration elements, the pristine target subscriber specifies one target machine at a time.

group manager

Lists groups of machines.

group subscriber

A subscriber type that can be included in a reference model. Use this subscriber to specify a group of targets. Unlike a profile manager, the group subscriber contains machines that are not yet endpoints.

operating system element manager

Lists operating system elements.

Prerequisites

In order to use Pristine Manager successfully, ensure that you have the following prerequisites:

- Your RIS or ADS server is already installed, configured, and working.
- There is a Tivoli endpoint on the machines where the RIS or ADS server is installed.
- You have already created the images on your RIS or ADS server. Refer to the appropriate documentation for instructions.
- On the ADS or RIS server, you have created a directory with the binary files to install the Tivoli endpoint. The directory must be shared and have full-control permissions.

Configuring Pristine Manager

Before you begin, you need to configure Pristine Manager.

Enabling Pristine Manager Databases

Pristine Manager organizes the machines, servers, operating system elements, and groups that you define into databases. When you install Pristine Manager, you must configure the RIM object for the database that you use. After installation of the Pristine Manager server component, the \$BINDIR\TME\PM\SCRIPTS directory contains an SQL script for each database vendor to create databases and users and to grant the appropriate authorities to use the database. It also includes an SQL script, which creates the tables in the Pristine Manager databases. You must run these script files before you can start creating the Pristine Manager objects.

To run the scripts, perform the following steps:

1. Refer to the following table to identify the names of the script and schema files for your database and locate the files in the \$BINDIR\TME\PM\SCRIPTS directory:

Table 47. Pristine Manager Script Files

Database Vendor	Script File	Schema File
DB2	pristine_db2_admin.sql	pristine_db2_schema.sql
Informix	pristine_infx_admin.sql	pristine_infx_schema.sql
Microsoft SQL	pristine_ms_sql_admin.sql	pristine_ms_sql_schema.sql
MVS	pristine_db2_mvs_admin.sql	pristine_db2_mvs_custom_schema.sql
Oracle	pristine_ora_admin.sql	pristine_ora_schema.sql
Sybase	pristine_syb_admin.sql	pristine_syb_schema.sql

2. Edit the `pristine_vendor_admin.sql` file, and ensure that the `filename=` statements list the correct directory in which the database is installed, for example, if you are using Microsoft SQL, `filename='c:\Program Files\Microsoft SQL Server\MSSQL\data\@DBNAME@.mdf'`.
3. On a client machine of your vendor database, run the script files: first, the `pristine_vendor_admin.sql` file and then, the `pristine_vendor_schema.sql` file.

Note: If you have already created the tables and you want to recreate and overwrite them, uncomment the ALTER and DROP statements by removing the dashes preceding them. However, be aware that by doing this, you will lose the data that is stored in the existing tables.

The `pristine_vendor_schema.sql` file creates the following tables:

PRISTINE_ACTIVITY

Contains information about the activity plan, the subscribers, and the operating system elements.

IMAGE

Contains information about the images and the server on which they are stored.

MACHINE

Contains the properties of the machines.

MACHINE_ENV

Contains the settings specific to the pristine machine, for example, the screen width.

OSELEMENT

Contains information about the operating system element, including the operating system name, version, and architecture.

OSELEMENT_IMAGE

Contains the association between image and operating system element.

PLUGIN

Contains the server technologies that are supported by Pristine Manager.

PLUGIN_CREATE_KEYS

Contains a list of environment keywords that are required for each supported server technology.

PLUGIN_ENVIRONMENT

Contains environment variables, determined by the type of pristine server, that configure the settings of the machine.

PRISTINE_ROLE

Contains the groups of machines.

ROLE_SUBSCRIBER

Contains the machines that are part of each group.

SERVER

Contains the properties of the servers.

SERVER_ENVIRONMENT

Contains the settings defined on a server that can be inherited by associated pristine machines when the operating system is installed.

SUPPORTED_OS

Contains the long and short names of operating systems that can be installed, supported versions, and the family to which the operating system belongs.

TMR_NAME

Contains a list of interconnected regions where the Pristine Manager daemon has been installed and run at least once.

Configuring Pristine Manager for Tivoli Enterprise Console

You can configure Pristine Manager to send events that track the progress of the pristine installation until its completion. The following sections provide a description of the Pristine Manager event configuration file and event classes. You must be familiar with the Tivoli Enterprise Console application. Refer to the *IBM Tivoli Enterprise Console: User's Guide* and the *IBM Tivoli Enterprise Console: Adapters Guide* for details.

To use Tivoli Enterprise Console integration, perform the following steps:

- Install Pristine Manager, Version 4.3.1.
- Install versions 3.7.1 or later of the Tivoli Enterprise Console, and the Tivoli Enterprise Console server in your Tivoli management region.
- Ensure the integration is enabled by using the **wpristine enable tec** command. See “wpristine” on page 262 for details.
- Register the Pristine Manager event classes on the Tivoli Enterprise Console server.
- Configure the event server, if necessary.
- Create the Pristine Manager Console.

Refer to the *IBM Tivoli Enterprise Console: User's Guide* for more information about installing and using the event server and console.

Enabling Integration with the Tivoli Enterprise Console

The following procedure demonstrates the steps required to enable Pristine Manager to send events to the Tivoli Enterprise Console server in an environment with several Tivoli management region servers and Tivoli Enterprise Console event servers, but where the Pristine Manager server does not reside on the same Tivoli management region server as the Tivoli Enterprise Console server.

1. Before connecting the Tivoli management regions, run the **wregister** command to register the resource for the event server (EventServer) on all Pristine Manager servers from which you want events sent and where you want the EventServer class visible. Run the command as follows:

```
wregister -i -v EventServer
```

Refer to *IBM Tivoli Enterprise Console: Installation Guide* for more information about registering the resource and the *Tivoli Management Framework: Reference Manual* for more information about the **wregister** command.

2. To enable integration with Tivoli Enterprise Console, run the **wpristine enable tec** command. See “wpristine” on page 262 for details.
3. Pristine Manager event classes are defined in the `tecad_pristine.baroc` file which is found in the `$BINDIR/TME/PM/SCRIPTS` directory on the Pristine Manager server. See “Pristine Manager Classes” on page 230 for more information about Pristine Manager event classes.

To set a rule base to manage events and to install the Pristine Manager event classes to the event server, the `pristine_tec_inst` script file is provided and is

located in the \$BINDIR/TME/PM/SCRIPTS directory. Copy the following files from the Pristine Manager server to a temporary directory (for this example, /tmp) on the Tivoli Enterprise Console server:

```
$BINDIR/TME/PM/SCRIPT/pristine_tec_inst.sh  
$BINDIR/TME/PM/tecad_pristine.baroc
```

4. Run the pristine_tec_inst.sh script as follows, specifying the path (/tmp) of the tecad_pristine.baroc file using the -w option:

```
sh /tmp/pristine_tec_inst.sh -b <your_rule> -s aix270 -u root -p <your_password>  
-t <your_console> -w /tmp/tecad_pristine.baroc
```

5. Specify the event server where Pristine Manager events should be sent. The event server is specified by defining the SeverLocation key in the tecad_pristine.conf file located in the following path:

```
$BINDIR/TME/PM/SCRIPT/tecad_pristine.conf
```

The tecad_pristine.conf file is discussed in greater detail in the “The tecad_pristine.conf File.”

6. Complete the process by creating the Pristine Manager console. See “Creating the Pristine Manager Console” on page 229.

The tecad_pristine.conf File: The tecad_pristine.conf configuration file for the Pristine Manager Tivoli Enterprise Console Integration component initially has the following default entries:

```
ServerLocation=@EventServer  
ConnectionMode=connection_oriented  
RetryInterval=1  
NO_UTF8_CONVERSION=NO  
ServerPort=0
```

The keywords are as follows:

ServerLocation

Specifies the keyword @EventServer or a fully qualified identifier if multiple event servers are running in an interconnected Tivoli management region environment. You must set the value of this field after installing Pristine Manager. Events are sent to the event server specified. To obtain the event server identifier, enter the following command from the command line:

```
wlookup -ar EventServer
```

The following is an example of the output of the command:

```
EventServer#red-region      1723798822.2.28#Tec::Server#  
EventServer#blue-region    1104217136.1.932#Tec::Server#  
EventServer#yellow-region  1272881041.1.1161#Tec::Server#
```

One of the event servers in this output example assumes the value of ServerLocation as follows:

```
ServerLocation=@EventServer#blue-region
```

Refer to the *Tivoli Management Framework: Reference Manual* for more information about the **wlookup** command.

ConnectionMode

Specifies the connection mode to use to connect to the event server. Valid values are connection_oriented (or its abbreviations CO and co) and connection_less. The default value is connection_oriented, which specifies that the connection with the event server is maintained for all events sent. A new connection is established only if the initial connection is lost.

When `connection_less` is specified or used by default, a new connection is established (and discarded) for each event that is sent.

RetryInterval

When `ConnectionMode=connection_oriented`, and the connection to the event server is lost, an adapter waits the specified number of seconds before connecting to a secondary server or buffering the events. While the adapter is waiting for the expiration of this interval, no new events are processed by the adapter.

This option allows an adapter to send all events to the primary event server even if the primary event server is stopped briefly, such as when loading a new rule base. If you use this option to wait for restarting an event server, the value should be set for a period of time longer than necessary for the event server to be stopped and then restarted.

The `RetryInterval` keyword is optional. The default is 120 seconds.

NO_UTF8_CONVERSION

When this option is set to YES, the Java Event Integration Facility (EIF) does not encode event data in UTF-8, the data is assumed to already be in UTF-8 encoding when passed to the EIF. It will, however, prepend the flag indicating that the data is in UTF-8 encoding if the flag does not exist at the beginning of the event data. The default value is NO.

ServerPort

Specifies the fixed reception port number on which the Tivoli Enterprise Console server listens for connection and adapter input. This keyword value should be set to 0 (the default value) on UNIX systems unless the portmapper is not available on the server.

If the Tivoli Enterprise Console server is a Windows NT system, the `ServerPort` should be set to the value of the `tec_rcv_agent_port` entry in the `tec_config` file. Refer to the *IBM Tivoli Enterprise Console: Adapters Guide* for more information about the server port specification on Windows NT machines. If the port number is not specified, the value is retrieved by calling the portmapper.

`ServerPort` can contain up to eight values, separated by commas. Specify one port number regardless of the number of `ServerLocation` values specified; however, if you specify more than one port number, you must specify a corresponding port number for each `ServerLocation` value. The default is 0.

By default, Pristine Manager sends all events to the event server. You can optionally specify event filters that indicate which events should not be sent. You can specify as many event filter entries as needed. To do so, you can include a `Filter` entry in the `tecad_pristine.conf` file, as follows:

Filter

Specifies how events are filtered. Filter statements are used by the `FilterMode` entry to determine which events should be discarded. An event matches a Filter statement if each `slot=value` in the Filter statement is identical to the corresponding `slot=value` in the event. A Filter statement must contain the event class and can include any other `slot=value` that is defined for the event class. The format of the Filter statement is:

```
Filter:Class=classname;slot=value;...;slot=value
```


Each Filter statement must be on, and contained in, a single, 512-character (maximum) line. The Filter statement cannot contain spaces. The class name specified for an event filter entry must match a defined class name. See “Pristine Manager Classes” on page 230 for a listing of classes for Pristine Manager.

For example, you can include the following entry in the `tecad_pristine.conf` file to discard all events generated when the removal operation begins.

```
Filter:Class=Remove_Start;
```

The filtering behavior can be modified by using the environment variable `FilterMode`. By default, `FilterMode` is set to `OUT`. By adding `FilterMode=IN` to the configuration file, only the events that match the filters are delivered to the event server. Refer to the *IBM Tivoli Enterprise Console: Adapters Guide* for detailed information about filtering.

If you make any changes to the `tecad_pristine.conf` file, you must stop and restart the event server for the changes to take effect.

Creating the Pristine Manager Console:

1. From the Tivoli Enterprise Console - Configuration window, right-click **Event Console Configuration** and select **Create Console** to create a new console. The Create Console dialog is displayed.
 - a. Enter the name and a description of the console in the **Name** and **Description** text boxes.
 - b. To assign the Pristine Manager event group to the console, select **Event Groups** from the left pane of the Create Console dialog. Select **Assign Groups** and the Assign Event Group to Console dialog is displayed.
 - c. Ensure that the Pristine Manager event group is listed in the Available Event Groups scrolling list. Highlight the **Pristine Manager** event group in the list.
 - d. You can also assign administration roles for the Event Group from the Assign Event Groups to Console dialog. Assign a role by selecting the appropriate check box. Click **OK** to assign the Pristine Manager event group to the console and the selected administration role.
 - e. To assign operators to the new console, select **Operators** in the left pane of the Create Console dialog. Select an operator and use the left-arrow button to move the selected operator from the Available Operators scrolling list to the Current Operators scrolling list.

Refer to the *IBM Tivoli Enterprise Console: User's Guide* for more information about performing tasks from the Tivoli Enterprise Console - Configuration window.
2. To view the event notices for the event group, select **Windows->Summary Chart View** from the Tivoli Enterprise Console - Configuration window. Double-click the summary chart and the Event Viewer window is displayed. The Event Viewer window lists all events for the Pristine Manager event group:



Pristine Manager Classes: The following table lists the class name of all events defined by Pristine Manager. Specifically, this class is defined in the tecad_pristine.baroc file. When you enable the Pristine Manager Tivoli Enterprise Console Integration and run the swdistecsrvr_inst.sh script, this class is compiled and included in the event server's rule base. Use this listing to understand how Pristine Manager event is mapped to Tivoli Enterprise Console events.

Note: Do *not* edit the tecad_pristine.baroc file.

Table 48. Tivoli Enterprise Console event class for Pristine Manager

Event	Attribute	Notes
PRISTINE_installation_event	machine_label os_element result	By default, severity = HARMLESS and sub_source = PM.

The Tivoli Enterprise Console event attributes relative to Pristine Manager are defined as follows:

Table 49. Tivoli Enterprise Console event attributes for Pristine Manager

Attribute Name	Attribute Value/Description
machine_label	Label of the target machine
os_element	A list of successful installation target machines
result	Result of the installation, displayed as severity. Values are: 0 or -1. 0=success -1=failure

Installing on Pristine Machines in Interconnected Regions

You can install on pristine machines in interconnected regions if the following prerequisites are satisfied:

- In all of the interconnected regions, there must be a RIM connection to the same database for both the Pristine Manager and the Inventory databases.

- The endpoint that is installed on the pristine machine must log in to the same region as the one that is specified for the pristine machine. Otherwise, the activity plan will time out.
- The same Tivoli Enterprise Console server must be used in all of the interconnected regions.

When an activity plan for a pristine installation is submitted, it uses the pristine daemon that is located in the same region as the new endpoint on the pristine machine. The pristine daemon manages the pristine installation without interacting with pristine daemons in the interconnected regions. When the installation ends, regardless of the outcome, the pristine daemon notifies the Activity Plan Monitor that launched the installation. The interconnected regions must be updated to make the new endpoint ready for future operations.

Chapter 10. Installing on Pristine Machines

You can use Pristine Manager either from Change Manager or the Activity Planner. However, the plan that is generated must be run from the Activity Planner GUI or CLI. Therefore, ensure that you have these services installed and configured correctly if you plan to use them to work with Pristine Manager. The following sections outline the main tasks required to do a pristine installation using either service.

Before you start, ensure that you have performed the following steps:

- Registered the plugin for Pristine Manager and that you have created the tables that contain the objects that you will create during the setup tasks. See “Enabling Pristine Manager Databases” on page 224 for details.
- Been assigned the correct authorization role in the Tivoli Management Framework. Otherwise, you cannot access the Pristine Manager objects at all. The following list shows the roles:

Pristine_Write

To create the Pristine Manager objects (servers, machines, operating systems, groups, and so on).

Pristine_Execute

To launch the plan to install on pristine machines.

Pristine_Read

To browse the Pristine Manager objects in read-only mode. You open the object as if you wanted to update it by clicking **Change**.

To install an operating system on pristine machines, you must do the following main tasks:

1. “Setup Tasks.” You create the building blocks of pristine machines, servers, operating system elements, and groups in the Pristine Manager databases.
2. “Preparing and Submitting a Plan” on page 256

Setup Tasks

The setup tasks prepare the components needed to create the reference model for a pristine installation. Although the instructions and the screens are taken from Change Manager, the steps are the same using Activity Planner. The setup tasks consist of the following actions:

- Defining Servers
- Defining Machines
- Creating Groups (optional)
- Creating an Operating System Element

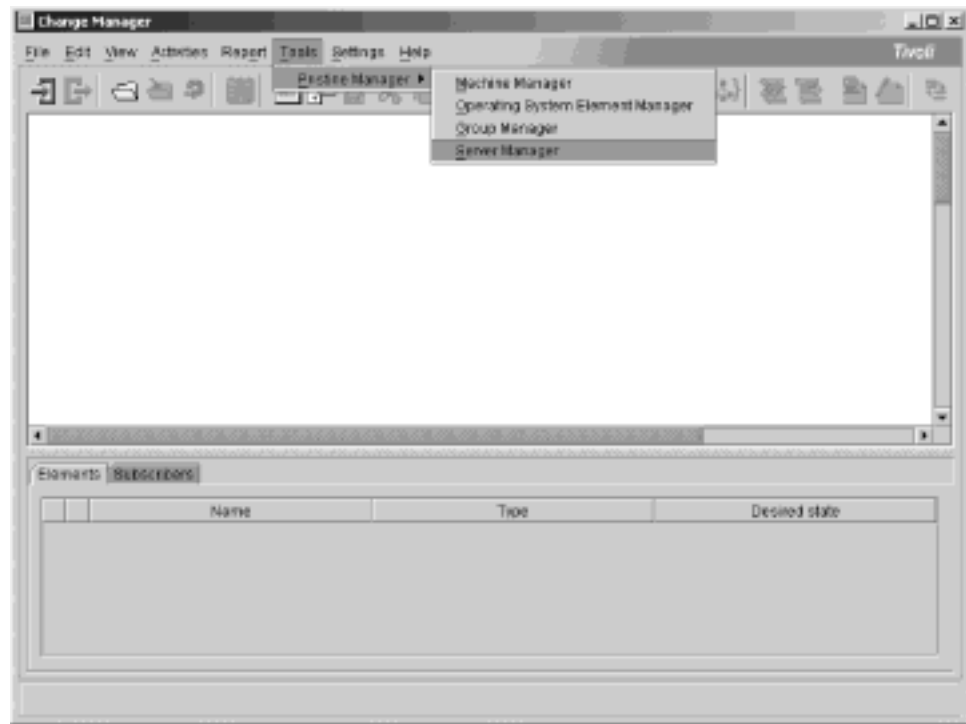
Defining Servers

You define the servers on which the operating system images are located.

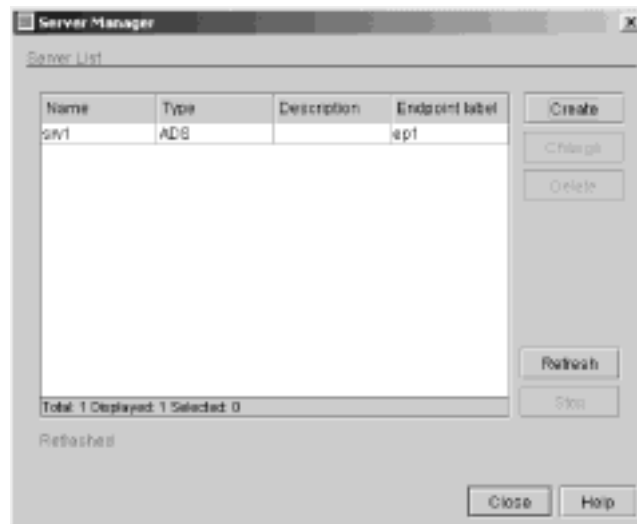
Tip: From the command line, you can perform the same task using the **wpristinesrv** command (see page 270.)

To define servers, perform the following steps:

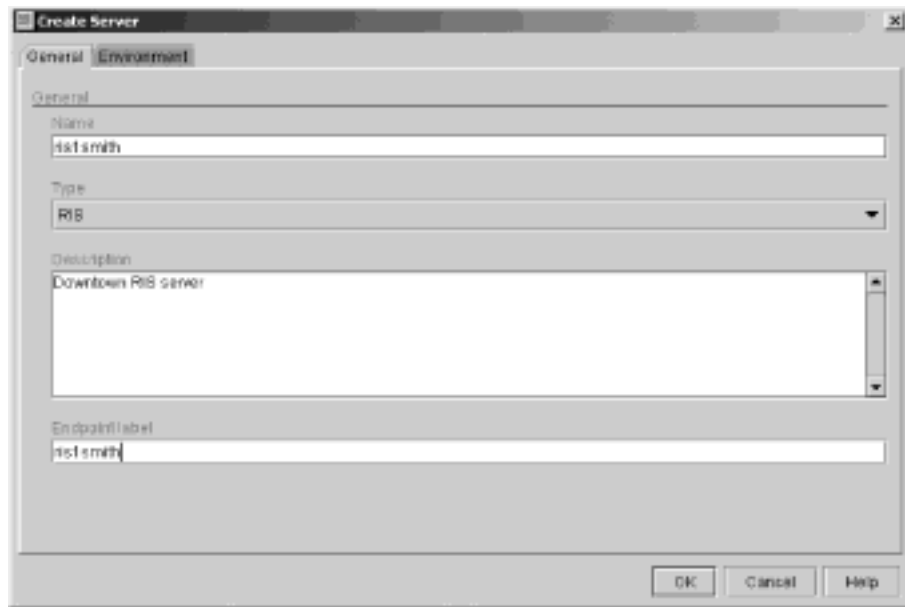
1. From the Tools menu on the Change Manager main window, select Pristine Manager and choose **Server Manager**.



The Server Manager dialog is displayed.



2. Click **Create**. The Create Server dialog is displayed.



3. On the **General** page, complete the following fields:

Label The server name.

Type Select the server type: **ADS** or **RIS**.

Description
Enter a description. Optional.

Endpoint label
Enter the Tivoli endpoint label of the server.

On the **Environment** page, you can set environment variables, which can be inherited as default settings by the machines associated with the server. See “Defining Environment Variables” on page 252 for more information about these settings.

Click **Add**. The Environment Variables dialog is displayed.



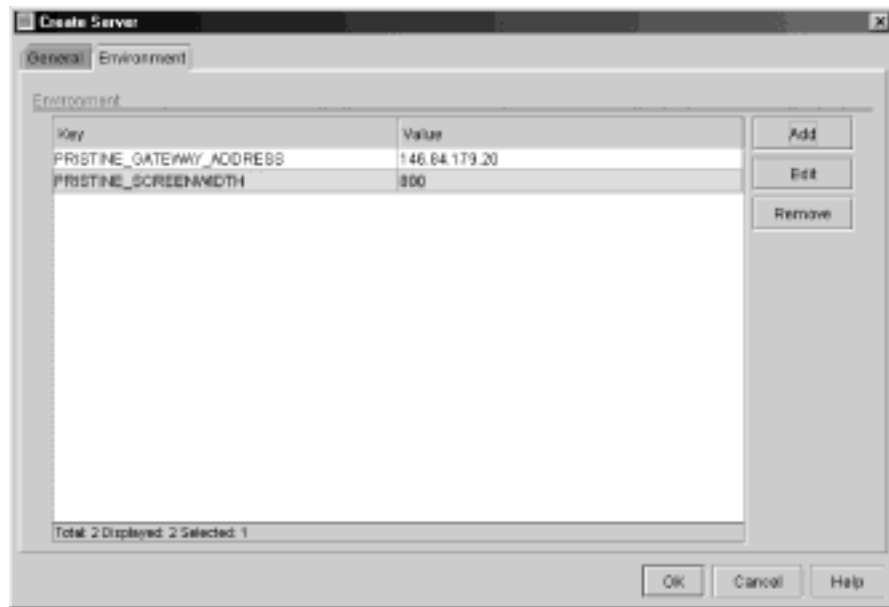
Complete the following fields:

Key Select a keyword from the list. The list is predefined by Pristine Manager, but you can specify your own keyword strings that you already defined in a response file. Optional.

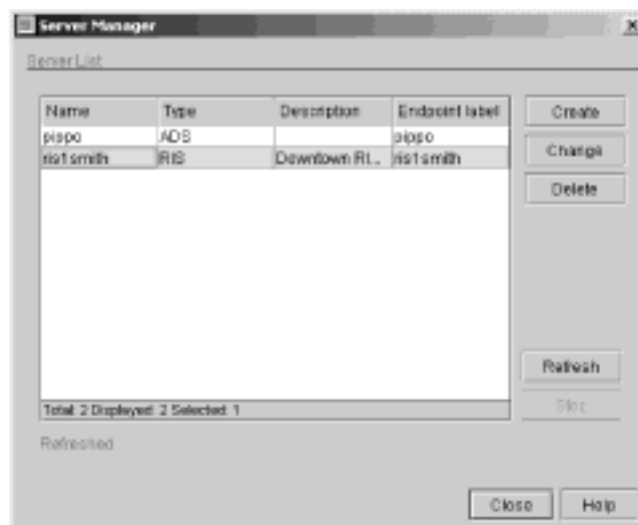
Value Enter a value for the keyword, for example, refresh=80 or server_userid=root.

Click **Help** on this dialog to display a list of predefined variables.

Click **OK**. The Environment page displays your input.



Click **OK**. The Server Manager window shows the newly defined server.



Defining Machines

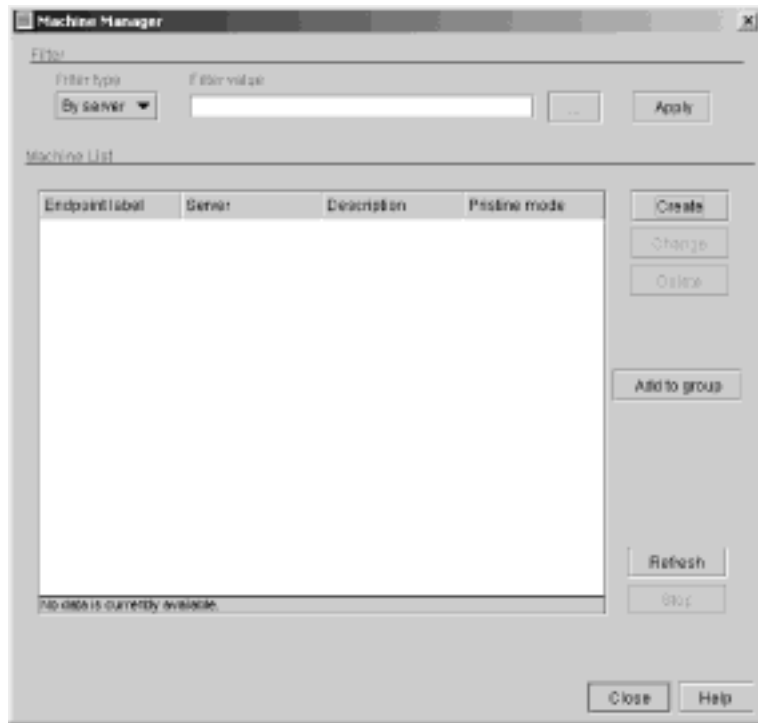
You define the pristine machines on which the operating system is to be installed. Pristine Manager defines these machines and enables Change Manager to work with them as if they were already Tivoli endpoints. As part of the installation, you can configure the machines either with settings that you specify here in the machine properties or that are inherited from the server that contains the images.

Tip: From the command line, you can perform the same task using the **wpristinemachine** command (see page 267.)

Note: You can create machines with a minimum of information and then return to modify them later. However, for a fully unattended installation, the values that are described as required must be specified before you submit the plan.

To define machines, perform the following steps:

1. From the Tools menu on the Change Manager main window, select Pristine Manager and choose **Machine Manager**. The Machine Manager dialog is displayed.



2. Click **Create**. The Create Machine dialog is displayed.



3. On the **General** page, complete the following fields:
Server Select the server that will perform the installation. Required.

Endpoint label

Specify a name for the machine. This will become the label when the machine becomes a Tivoli endpoint. Required.

Description

Enter a description.

MAC address

Enter a 12-character hexadecimal MAC address. Required for installing from an ADS server. For RIS servers, it is recommended that you specify this setting, because it could be used to verify the success of installation on the pristine machine.

SMBIOS GUID

Enter the 32-character hexadecimal System Management BIOS Globally Unique identifier that has been assigned to the machine. It can also include dashes. Required for installing from a RIS server.

Pristine mode

Displays how the pristine installation should proceed if the operating system is already installed on the machine.

Note: For the If different and If newer options, ensure that the machine has been scanned by Inventory.
It can be one of the following:

Ignore

The machine is not checked for an operating system and the installation is not performed. Default.

Force The operating system is installed in all cases.

If different

The operating system is installed only if it is different from the one already installed, for example, to replace Microsoft Windows 2000 with Microsoft Windows XP Professional.

If newer version

The operating system is installed only if the images contain a newer version within the same family than the one already installed, for example, to replace Microsoft Windows NT with Microsoft Windows XP Professional. However, Microsoft Windows 2000 is not newer than Microsoft Windows 95 because they belong to different families.

On the **Network** page, some of the fields might display default values that are inherited from the server that you specify on the General page. To override them, enter a different value.

Create Machine

General Network **Advanced**

Network

Computer name

Host name

Domain
☒ Inherit from server variable PRISTINE_DOMAINNAME

IP Address
☒ Inherit from server variable PRISTINE_IPADDRESS
☐ Obtain an IP address from a DHCP server
☐ Specify an IP address

Subnet mask
☒ Inherit from server variable PRISTINE_SUBNETMASK

Default gateway
☒ Inherit from server variable PRISTINE_GATEWAY_ADDRESS

DNS server address
☒ Inherit from server variable PRISTINE_DNSERVER_ADDRESS

OK Cancel Help

Complete the following fields. After you complete a successful pristine installation of this machine, you cannot change the computer name or the host name. To change them, you must delete the machine and recreate it.

Computer name

By default, the name that you specify for the endpoint label on the General page is displayed. To change it, enter a name for the machine. On a RIS server, the pristine machine will have the name that you specify here. Required.

Host name

Enter a host name for the machine. On an ADS server, the pristine machine will have the name that you specify here. Required.

Domain

Enter the domain on which the machine will be or mark the check box to inherit the domain from the pristine server.

IP address

Specify an IP address for the machine or mark the appropriate check box.

Subnet mask

Specify the subnet mask of the machine or mark the check box to inherit the value from the pristine server.

Default gateway

Enter the name of the default network gateway.

DNS address

Enter the address of the DNS server for the machine or mark the check box to inherit the value from the pristine server.

On the **Tivoli** page, enter information that will be used to include the machine in the Tivoli environment. Some of the fields might display default values that are inherited from the server that you specify on the General page. To override them, enter a different value.

The screenshot shows a 'Create Machine' dialog box with four tabs: 'General', 'Network', 'Tivoli', and 'Environment'. The 'Tivoli' tab is selected. It contains several fields: 'Endpoint label' with the value 'Mail', 'Tivoli region' with a dropdown menu showing 'lanc-0000', 'Endpoint port' with a checked box 'Inherit from server variable PRISTINE_ENDPOINT_PORT', 'Gateway port' with a checked box 'Inherit from server variable PRISTINE_ENDPOINT_GWPORT', 'Endpoint options' with a checked box 'Inherit from server variable PRISTINE_ENDPOINT_OPTIONS', and 'Endpoint setup path' with a checked box 'Inherit from server variable PRISTINE_TMA_SETUP_PATH'. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

Complete the following fields:

Endpoint label

This field is read-only and displays the label that you specified on the General page.

Tivoli region

The drop-down list shows the interconnected Tivoli regions on which a pristine server has been installed. Select the region to which the endpoint on the machine will belong.

Endpoint port

Enter the port of the endpoint that is installed on the machine or mark the check box to have the machine inherit the value from the server.

Gateway port

Enter the port for the Tivoli gateway to which endpoint logs in or mark the check box to have the machine inherit the value from the server.

Endpoint setup path

Specify the path from which the binary installation files for the endpoint are taken or mark the check box to have the machine inherit the value from the server. The directory must be shared with full-control permissions. To avoid this, see “PRISTINE_TMA_SETUP_PATH and Sharing Access to Endpoint Setup Files” on page 274. Required.

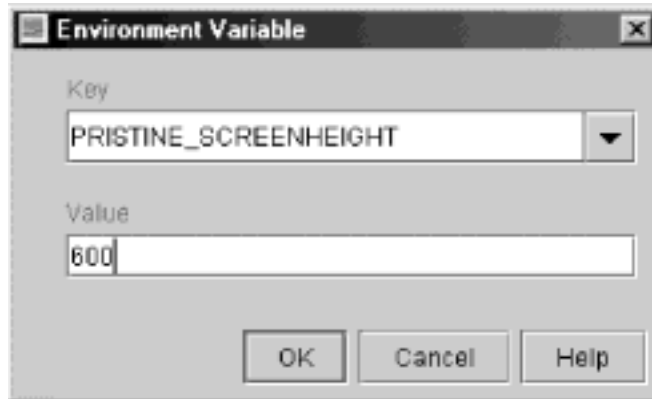
Endpoint options

Enter the options used during the setup of the endpoint or mark the

check box to have the machine inherit the value from the server. You must have the `-gateway+gw_port` option and you must not use the `-n` option. Required.

On the **Environment** page, the Inherited environment list displays settings that the machine can inherit from the server by default. They are displayed as key-value pairs, such as `_PRISTINE_COLORDEPTH=16`. In the Machine-defined environment list, you can define additional settings that are specific to the machine. See “Defining Environment Variables” on page 252 for more information about these settings.

To add a setting, click **Add**. The Environment Variable dialog is displayed.

The image shows a dialog box titled "Environment Variable". It has two main input sections. The first section is labeled "Key" and contains a text box with the text "PRISTINE_SCREENHEIGHT" and a small downward-pointing arrow to its right. The second section is labeled "Value" and contains a text box with the text "600". At the bottom of the dialog, there are three buttons: "OK", "Cancel", and "Help".

Complete the following fields:

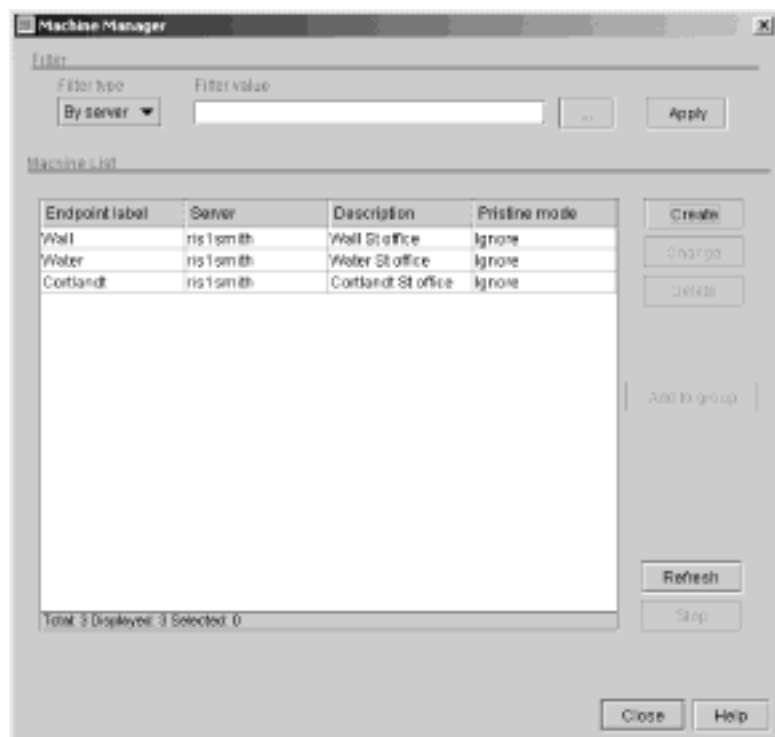
Key Select a keyword from the list. Pristine Manager lists keywords depending on the installation server.

Value Enter a value for the keyword.

Click **OK**. The Environment page displays your input in the Machine-defined environment list. Click **OK**.



The Machine Manager dialog displays the new machine in the Machine list.



Filtering Machines in the List

On the Machine Manager dialog, you can list the machines in any of the following ways:

- List all of the machines.

- Filter the machines by group
- Filter the machines by server

To list *all* of the machines, click **Refresh**. The machines are displayed under Machine List.

To filter the machines *by group*, perform the following steps:

1. Select **By group** from the Filter type drop-down list.
2. Enter a group name in the Filter value field, or click the browse (...) button and select a group from the Group Manager dialog and click **OK**.
3. Click **Apply**. The machines that belong to the group are displayed in the machine list.

To filter the machines *by server*, perform the following steps:

1. Select **By server** from the Filter type drop-down list.
2. Enter a server name in the Filter value field, or click the browse (...) button and select a server from the Server Manager dialog and click **OK**.
3. Click **Apply**. The machines that belong to the server are displayed in the machine list.

Adding Machines to a Group

You can make a machine part of a group. A machine can be part of any number of groups. First, ensure that the group has been created. To create a group, see “Creating Groups” on page 245.

To make the new machine part of a group, perform the following steps:

1. Select the machine in the list on the Machine Manager dialog and click **Add to group**. The Group Manager dialog is displayed.
2. Select a group and click **Close**. To make the machine part of another group, repeat these steps.

Updating Multiple Machines

If you want to provide the same value for the same setting of several machines, you can do a multiple update. Perform the following steps:

1. From the Tools menu on the Change Manager main window, select **Pristine Manager** and choose **Machine Manager**. The Machine Manager dialog is displayed.
2. Select all of the machines to be updated and click **Change**. The Update Machines dialog is displayed.

Update Machines

General Network Tools

General

Server
r1s1amth

Endpoint label

Description

MAC address

gmplog GUID

Pristine mode
Ignore

OK Cancel Help



The **Network** page shows how multiple machines can inherit defaults from the pristine server.

Note: The **Environment** page is not available. Environment variables must be modified either by passing them on from a pristine server or individually by updating one machine at a time.

3. After you make your updates and click **OK**. The changes are applied to the properties of all of the machines that you selected.
4. Click **OK** on the Machine Manager window.

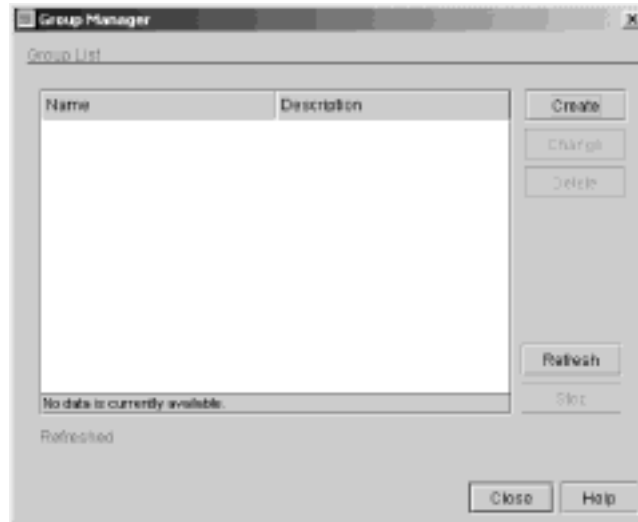
Creating Groups

You can group target machines, based on the needs of the user group, and install software that is tailored for that group. Create a group only if you want to subscribe a group of machines to reference model.

Tip: From the command line, you can perform the same task using the **wpristinegroup** command (see page 265).

To create a group, perform the following steps:

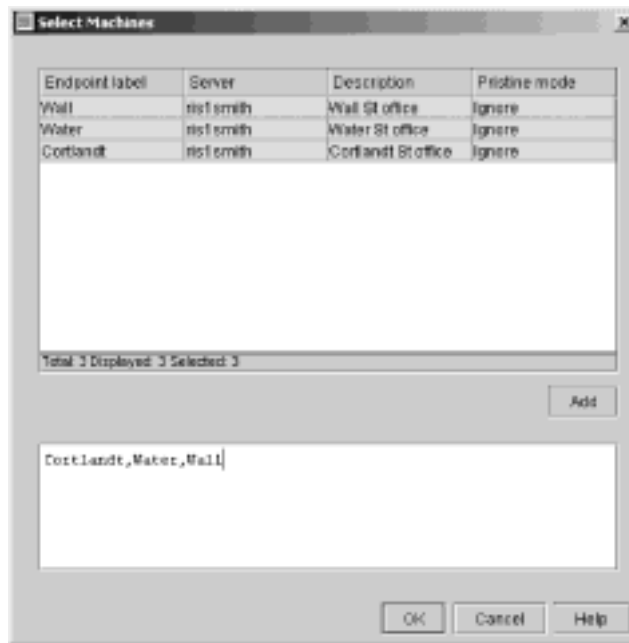
1. From the Tools menu on the Change Manager main window, select Pristine Manager and choose **Group Manager**. The Group Manager dialog is displayed.



2. Click **Create**. The Create Group dialog is displayed.



3. Complete the following fields:
 - Name** Enter a name for the group.
 - Description** Enter a description of the group.
4. Machines must already be defined. Click **Add** to add machines to the group and select the machines on the Select Machines dialog and click **Add** and **OK**.

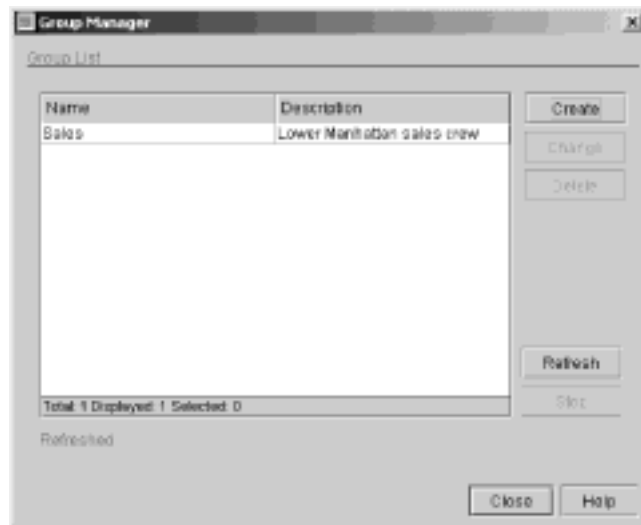


Any machines that you added appear in the Members list on the Create Group dialog.



Notes:

- a. If you do not have machines defined yet, you can add them to a group after you define them.
 - b. A machine can belong to more than one group.
5. Click **OK** on the Create Group dialog.



The Group Manager displays the newly created group. Click **Close**.

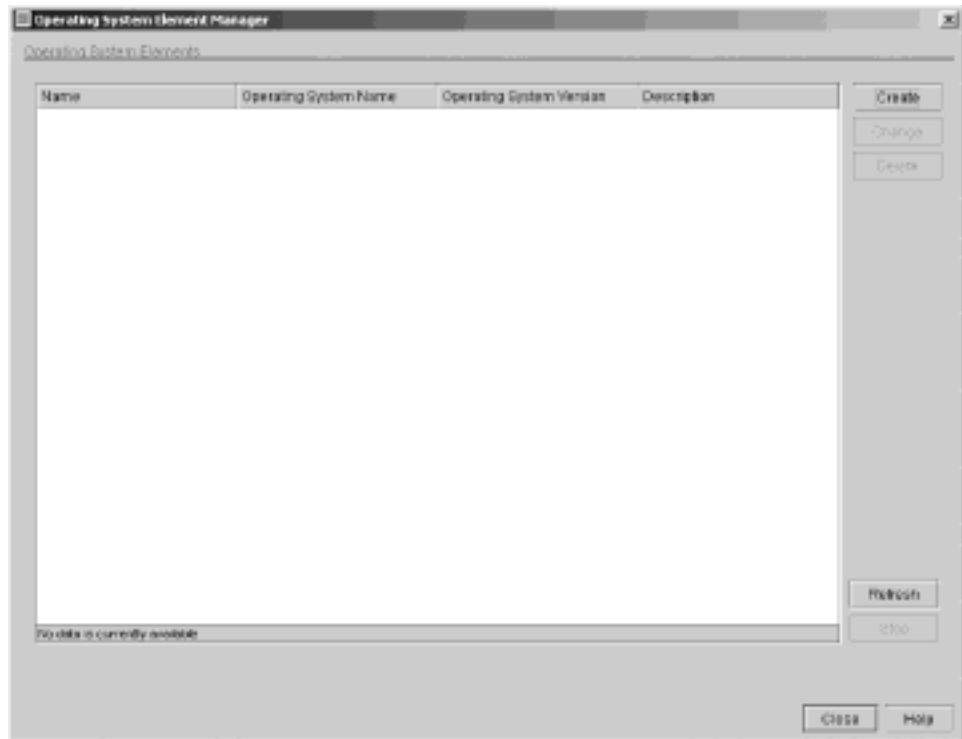
Creating an Operating System Element

An operating system element can contain any number of images, but only one per server.

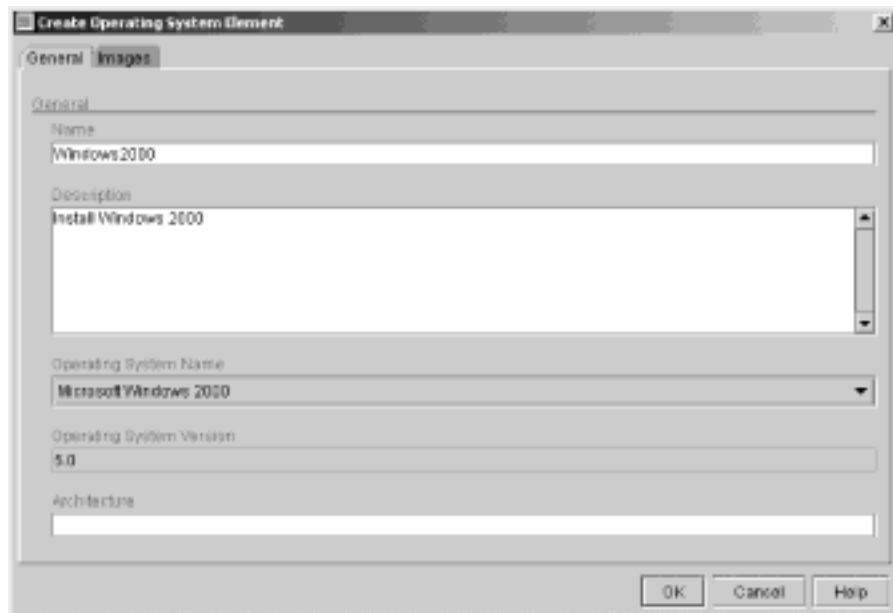
Tip: From the command line, you can perform the same task using the **woselement** command (see page 260).

To create an operating system element, perform the following steps:

1. From the Tools menu on the Change Manager main window, select **Pristine Manager** and choose **Operating System Element Manager**. The Operating System Element Management dialog is displayed.



2. Click **Create**. The New Operating System Element dialog is displayed.



3. On the **General** page, complete the following fields:

Name

Enter a name to identify the operating system element.

Description

Enter a description that helps to distinguish the operating system element.

Operating system name

Select the operating system to be installed by this element from the drop-down list.

Operating system version

The version of the operating system is selected automatically when you select the operating system name.

Architecture

Enter the architecture used by the operating system.

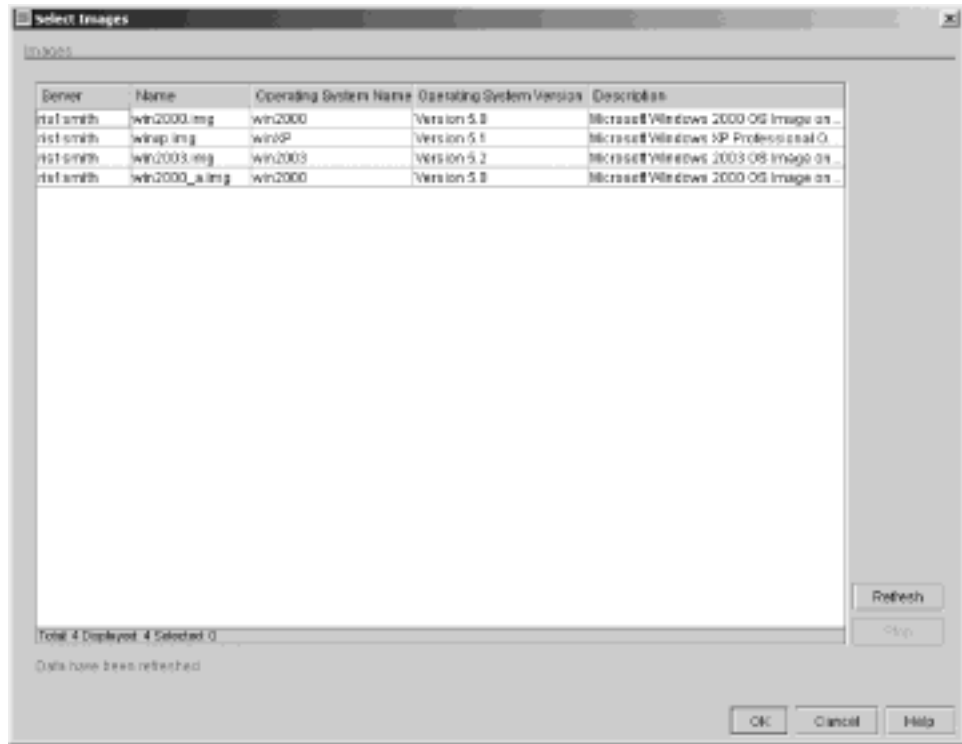
Note: The operating system name and version are compared to the COMPUTER Inventory table to determine the outcome of the pristine mode option selected on the General page.

4. On the Images page, click **Add**.



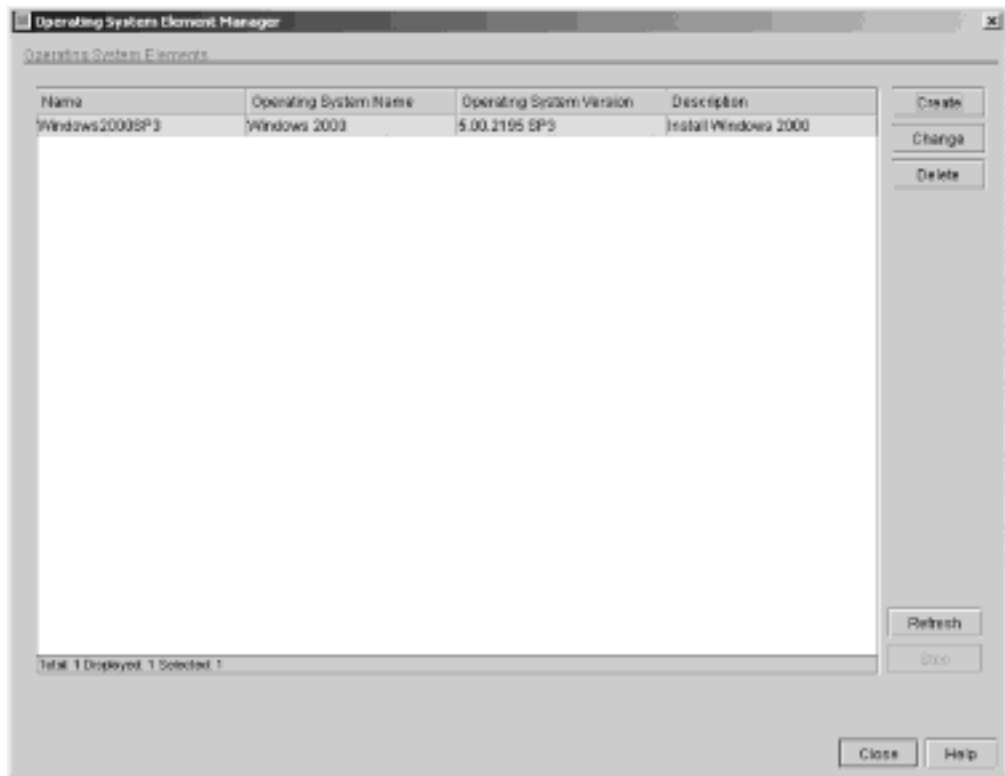
The Select Images dialog is displayed and lists the images that are available from the servers that you defined.

Because the list is being retrieved from the installation servers, it might take a while.



Select the images to be associated with the operating system element and click **OK**. The selected images are listed on the Images page. Note that you can select only one image per server. Click **OK** on the New Operating System Element dialog.

The new element is displayed in the Operating System Management window.



Defining Environment Variables

You can define environment variables either in the server properties or in the machine properties and either by using the Pristine Manager GUI or the CLI. The variables defined on the server are inherited by the pristine machines associated with it. In the machine properties, you see both the variables inherited from the server and those defined for the machine. By defining environment variables for the machine, you do one of the following:

- Override a default setting inherited from the server by specifying a different value for the same keyword
- Add other settings from the predefined list of keywords
- Add a setting for which you have defined a variable in the installation response file. See “Defining Your Own Environment Variables” on page 254 for details.

Pristine Manager automatically includes the following variables in the prefilled list of environment variables:

PRISTINE_COLORDEPTH

The display color depth in bits per pixel, for example, 24.

PRISTINE_COMPUTERNAME

The computer name assigned to the machine on which the operating system is installed, for example, cortlandt. For RIS servers, it is also used to create the computer in the Active Directory. If you do not specify this value, your input for the endpoint label of the machine is used.

In the Pristine Manager GUI, you can also specify this name in the Computer name field on the Network page of the machine properties.

PRISTINE_DNSSERVER_ADDRESS

The address of a DNS server (used mostly for static IP addresses).

In the Pristine Manager GUI, you can also specify this name in the DNS Address field on the Network page of the machine properties.

PRISTINE_DOMAINNAME

The name of the domain in which the computer is defined, for example, salescompany.com (usually appended to the host name).

In the Pristine Manager GUI, you can also specify this name in the Domain field on the Network page of the machine properties.

PRISTINE_ENDPOINT_GWPORT

The port of the gateway to which the endpoint connects, for example, 9494.

In the Pristine Manager GUI, you can also specify this name in the Gateway port field on the Tivoli page of the machine properties.

PRISTINE_ENDPOINT_LABEL

The label that the machine has when it is defined as a Tivoli endpoint, for example, cortlandt.

In the Pristine Manager GUI, you can also specify this label in the Endpoint label field on the Tivoli page of the machine properties. After you define the machine, you cannot change this value. To change it, you must delete the machine and redefine it.

PRISTINE_ENDPOINT_OPTIONS

The endpoint options, for example, -gnewyork+9491.

In the Pristine Manager GUI, you can also specify this name in the Endpoint options field on the Tivoli page of the machine properties.

PRISTINE_ENDPOINT_PORT

The endpoint port, for example, 9495.

In the Pristine Manager GUI, you can also specify this name in the Endpoint port field on the Tivoli page of the machine properties.

PRISTINE_GATEWAY_ADDRESS

The address of the default gateway (used mostly for static IPs).

In the Pristine Manager GUI, you can also specify this name in the Default Gateway field on the Network page of the machine properties.

PRISTINE_GUIRUNONCE

Specified for the machine, one or more lines of instructions that are run when the operating system restarts on the machine after it is installed. See “Running Commands on Pristine Machines” on page 256 for more information.

PRISTINE_HOSTNAME

The host name of the computer on which the operating system is installed, for example, cortlandt. For ADS servers, it is used only to create the device on the ADS console. If you do not specify a value for this keyword, the string for the endpoint label of the machine is used.

In the Pristine Manager GUI, you can also specify this string in the Host name field on the Network page of the machine properties.

PRISTINE_IPADDRESS

The IP address of the computer on which the operating system is installed. Use a dot format if the machine has a static address, for example, 192.17.24.223. Otherwise, use the string DHCP to indicate that the machine obtains its address from a DHCP server.

In the Pristine Manager GUI, you can also specify this name in the IP Address field on the Network page of the machine properties.

PRISTINE_MAC_ADDRESS

The MAC address of the primary network adapter. For ADS server, you must specify it. For RIS servers, it is recommended that you specify this setting, because it is used to verify the success of installation on the pristine machine.

In the Pristine Manager GUI, you can also specify this value in the MAC address field on the General page of the machine properties.

PRISTINE_SCREENHEIGHT

The display height in pixels, for example, 600.

PRISTINE_SCREENWIDTH

The display width in pixels, for example, 800.

PRISTINE_SERVER_GUIRUNONCE

Inherited from the server, one or more lines of instructions that are run when the operating system restarts on the machine after it is installed. See “Running Commands on Pristine Machines” on page 256 for more information.

PRISTINE_SMBIOS_GUID

The machine System Management BIOS Globally Unique ID (SMBIOS GUID), a 32-character hexadecimal value. Required for installation from a RIS server. For ADS servers, it is recommended that you specify this setting, because it is used to verify the success of installation on the pristine machine.

In the Pristine Manager GUI, you can also specify this value in the SMBIOS GUID field on the General page of the machine properties.

PRISTINE_SUBNETMASK

The network subnet mask, for example, 255.255.0.0 (used mostly for static IPs).

In the Pristine Manager GUI, you can also specify this name in the Subnet Mask name field on the Network page of the machine properties.

PRISTINE_TMA_SETUP_PATH

The directory that contains the files required to install the Tivoli endpoint on target machines (for example, setup.exe, setup.iss and related files). For more information about this variable, see “PRISTINE_TMA_SETUP_PATH and Sharing Access to Endpoint Setup Files” on page 274.

In the Pristine Manager GUI, you can also specify this name in the Endpoint setup path field on the Tivoli page of the machine properties.

PRISTINE_TMR_SERVER

The region to which the endpoint on the machine will belong.

In the Pristine Manager GUI, you can also specify this name in the Tivoli region field on the Tivoli page of the machine properties.

Defining Your Own Environment Variables

You define your own environment variables, in two steps:

1. Define a keyword in the installation response file. This step is described in the following paragraphs.
2. From the Pristine Manager GUI or CLI, specify the keyword that you define in the response file and a value.

For ADS servers only: Ensure that you copy the sysprep.inf file of the operating system image to the %WINDIR%\TEMP\PM\sysprep directory and that you rename the copy of the INF file to the same name as the image file, for example, for an image file named win2k.img, rename the INF file as win2k.inf. If your Tivoli endpoint is running as a service, the %WINDIR%\TEMP directory is the system environment variable of your ADS server. If you are running the Tivoli endpoint as a console, the %WINDIR%\TEMP directory is the user environment variable of your ADS server. To verify your settings, open the System Properties window of the computer and select the Environment variables button on the Advanced tab.

To define a keyword in the installation response file, perform the following steps:

1. Edit the INF file for ADS servers or the SIF file for RIS servers, and add the new keyword to the file. For example, to create the keyword REFRESH for the refresh setting, change the DISPLAY section of the file the line in bold from the following section:

```
[Display]
ConfigureAtLogon = 0
BitsPerPel =16
XResolution =1024
YResolution =768
VRefresh = 60
AutoConfirm = 1
```

As follows:

```
[Display]
ConfigureAtLogon = 0
BitsPerPel =16
XResolution =1024
YResolution =768
VRefresh = ^REFRESH^
AutoConfirm = 1
```

Note: You must put your keyword between the caret (^) symbols.

Then, to set a refresh rate of 60, for example, define the keyword REFRESH and specify a value of 60.

Modifying the Installation Job for ADS Servers: Installation from ADS servers enables you to customize the configuration of the machine, for example, partition the hard disk. Pristine Manager provides a template file to include this type of customization in the environment variables. You change a task in the installation job in two steps:

1. Edit the tivdeployimage.xml file, which is located in the Tivoli\bin\lcf_bundle.40\bin\w32-ix86\PM directory of the Pristine Manager gateway component.
2. From the Pristine Manager GUI or CLI, specify the keyword that you define in the XML file and a value.

For example, to partition the disk, edit the task in the tivdeployimage.xml file, which originally appears as follows:

```
<!-- Create a single partition on the disk -->
<task description="Partition the disk">
  <command>/bmonitor/bmpart.exe</command>
  <parameters>
    <parameter>\device\harddisk0</parameter>
    <parameter>/init</parameter>
  </parameters>
</task>
```

```

        <parameter>/C</parameter>
        <parameter>/A</parameter>
    </parameters>
</task>

```

And modify it as follows (changes in **bold**):

```

<!-- Create a single partition on the disk -->
<task description="Partition the disk">
    <command>/bmonitor/bmpart.exe</command>
    <parameters>
        <parameter>\device\harddisk0</parameter>
        <parameter>/init</parameter>
        <parameter>/C:$CDISK_SIZE$</parameter>
        <parameter>/A</parameter>
    </parameters>
</task>

```

Note that the variable that you specify in the file must appear between dollar sign delimiters.

Then, to partition the C drive with a size of 6 MB, for example, define the keyword `CDISK_SIZE` and specify a value in KB of 6000.

Running Commands on Pristine Machines

When Pristine Manager installs the Tivoli endpoint, it reboots the pristine machine. If you need to run other commands on the machine before it is rebooted, define an environmental variable with the commands. Use the `PRISTINE__SERVER_GUIRUNONCE` keyword for the pristine server or use the `PRISTINE_GUIRUNONCE` keyword for the machine and separate commands with the pipe character (`|`). The following example shows the values to specify to run a net use on the machine `Cortlandt\share` by the user `smith` with the password and then a net share command of `c:\shares.txt`:

```
net use * \\Cortlandt\share /user:smith password | net share >c:\shares.txt
```

If you specify values for the server keyword `PRISTINE__SERVER_GUIRUNONCE` and for the machine-defined keyword `PRISTINE_GUIRUNONCE`, the commands defined at the machine are run after those at the server.

Preparing and Submitting a Plan

After you have completed the setup tasks, you must pull the Pristine Manager objects into a plan and then submit the plan. You do this in different ways in Activity Planner and Change Manager.

Using Activity Planner

If you are using Activity Planner, perform the following tasks:

1. “Defining an Activity” on page 20, including “Defining a Pristine Manager Activity” on page 25.
2. “Selecting Targets for an Activity” on page 26, including “Specifying a Pristine Target Subscriber” on page 31.
3. “Saving the Activity Plan” on page 43. Save the plan as a template if you want to run it.
4. “Submitting an Activity Plan” on page 49.

Using Change Manager

If you are using Change Manager, you must perform the following tasks:

1. “Creating the Reference Model Structure” on page 172.

This includes the following subtasks:

- “Adding an Operating System Element” on page 177
 - “Assigning Pristine Manager Subscribers” on page 185
2. Submitting the reference model to the Activity Planner. See “Submitting an Activity Plan” on page 191.

Chapter 11. Using the Command Line

This chapter describes how to work with Pristine Manager using the command line interface (CLI).

For information about syntax conventions and getting help for the command line, see “Using the Command Line” on page xix.

Table 50. Commands for working with Pristine Manager

Command	Purpose	Details on page
woselement	Works with operating system elements	260
wpristine	Works with the Pristine Manager execution engine	262
wpristinegroup	Works with groups	265
wpristineexport	Exports the machines from the ADS or RIS server database	264
wpristinemachine	Works with pristine machines	267
wpristinesrv	Works with servers	270

woselement

Creates, modifies, lists, and deletes operating system elements.

Syntax

```
woselement create -l label -o os_name [-d description] [-p os_architecture] [-a server:image]...
```

```
woselement delete -l label
```

```
woselement edit -l label {[-d description] [-o os_name] [-p os_architecture] [-a server:image]... [-r server:image]...}
```

```
woselement list [-l label] [-m]
```

Description

The **woselement** command creates operating system elements, modifies the properties of the operating system element, deletes operating system elements, and lists operating system elements and the image-server pairs.

Options

The following lists the subcommands of the **woselement** command:

- create** Creates an operating system element.
- delete** Deletes an operating system element.
- edit** Modifies the properties of the operating system element.
- list** Lists the operating system elements in the operating system element database or the image-server pairs in the specified operating system element.

The following lists the options and variables belonging to the subcommands:

- a *server:image***
Adds an image to the operating system element. The server name and image must be separated by a colon. You can specify more than one image but only one image per server.
- d *description***
Provides the specified description of the operating system element.
- l *label*** Provides the specified label for the operating system element or specifies to which operating system element the subcommand applies.
- m** Lists the image-server pair of a specified operating system element.
- o *os_name***
Specifies the operating system to be installed. You can specify the operating system names in either long or short form, regardless of case, as follows: Microsoft Windows 2000, Win2000, Microsoft Windows XP, WinXP, Microsoft Windows 2003, Win2003, Microsoft Windows NT, WinNT, Microsoft Windows 95, Win95, Microsoft Windows 98, Win98, Microsoft Windows Millennium, WinME. However, you can abbreviate any of the preceding strings, such as entering XP, but you must use the case that is specified here.

- p *os_architecture***
Specifies the architecture used by the operating system.
- r *server:image***
Removes the specified server and image from the operating system element. The server name and image must be separated by a colon.

Authorization

The **admin** role is required for all **woselement** commands.

Examples

1. To create an operating system element called **Windows2000SP3** with a description, enter the following command:

```
woselement create -l Windows2000SP3 -o Win2000 -d "Win2000 \
Service Pack 3 for Sales reps"
```
2. To create an operating system element called **Windows2000SP3** with the images **win2000** and **win2k** of Microsoft Windows 2000 on the servers **ris1smith** and **ris2jones**, respectively, enter the following command:

```
woselement create -l Windows2000SP3 -o "Microsoft Windows 2000" \
-a ris1smith:win2000 -a ris2jones:win2k
```
3. To change the images on server **ris1smith** from **win2000** to **win2000a** for the operating system element called **Windows2000SP3**, enter the following command:

```
woselement edit -l Windows2000SP3 -a ris1smith:win2000a
```
4. To list existing operating system elements, enter the following command:

```
woselement list
```
5. To list the image-server pairs that are included in the operating system element called **Windows2000SP3**, enter the following command:

```
woselement list -l Windows2000SP3 -m
```

See Also

None.

wpristine

Works with the Pristine Manager execution engine.

Syntax

wpristine start

wpristine stop

wpristine restart

wpristine status

wpristine config

wpristine enable tec

wpristine disable tec

wpristine set trace_size *value*

wpristine set trace_level *value*

wpristine set timeout *value*

Description

The **wpristine** command manages the Pristine Manager daemon, which is installed and started automatically when you install Pristine Manager plug-in. It also provides information about and enables you to configure options for the pristine installation. Configuration changes become effective when you restart the Pristine Manager daemon.

Options

The following lists the subcommands of the **wpristine** command:

start Starts the daemon for Pristine Manager.

stop Stops the daemon for Pristine Manager.

restart Restarts the daemon for Pristine Manager.

status Lists information about the operating system installations that are running at the moment you launch the command. It creates a table that lists the status for each target machine. The status can be: canceled, completed, failed, running, or timedout.

config Provides information about how the daemon for Pristine Manager is configured. This includes trace settings, a timeout limit, and whether or not notification from Tivoli Enterprise Console is enabled.

enable tec

Enables notification from Tivoli Enterprise Console about completion of the operating system installation.

disable tec

Disables notification from Tivoli Enterprise Console.

set trace_size *value*

Sets the maximum size (in megabytes) of the trace file. To allow a file of any size, enter **0**.

set trace_level *value*

Sets the level for the trace. This affects how much detail is reported during a trace. Enter one of the following values:

- 0** Reports only fatal errors.
- 1** Reports all errors.
- 3** Reports all errors and informational messages.
- 4** Verbose, reports all messages generated.

set timeout *value*

Sets the maximum time, in hours, for an operating system installation to be completed before it times out and is considered unsuccessful.

Authorization

The **admin** role is required for the **wpristine** command.

Examples

1. To allow for a trace file of up to 2 MB, enter the following command:
`wpristine set trace_size 2`
2. To have a trace file that details all the messages generated during an installation, enter the following command:
`wpristine set trace_level 4`
3. To time out the installation after 24 hours, enter the following command:
`wpristine set timeout 24`

See Also

None.

wpristineexport

Exports machines from RIS and ADS servers.

Syntax

```
wpristineexport -s server_name [-f file] [-header]
```

Description

The **wpristineexport** command exports machines from the specified server from RIS and ADS servers.

Options

The following lists the options and variables belonging to the subcommands:

- f *file*** Exports the list of machines to the specified file in CSV format. See the description of the **import** option on page 267 for a description of the CSV file.
- header** Provides column titles for the fields as a header in the CSV file.
- s *server*** Specifies the server from which the machines are retrieved.

Authorization

The **admin** role is required for the **wpristineexport** command.

Examples

To export the machines defined on a server called **ris1smith** to a file named **winmachines**, enter the following command:

```
wpristineexport -s ris1smith -f winmachines
```

See Also

None.

wpristinegroup

Creates, modifies, lists, and deletes groups.

Syntax

```
wpristinegroup create -l label [-d description] [-a member]...
```

```
wpristinegroup delete -l label
```

```
wpristinegroup edit -l label [-d description] [-a member]... [-r member]...
```

```
wpristinegroup list [-l label] [-m]
```

Description

The **wpristinegroup** command creates groups, modifies the list of pristine target machines in the group, deletes groups, and lists groups and their members.

Options

The following lists the subcommands of the **wpristinegroup** command:

create Creates a group.

delete Deletes a group.

edit Modifies the properties of the group.

list Lists the groups in the group database or the pristine target machines in the specified group.

The following lists the options and variables belonging to the subcommands:

-a *member*

Adds the specified pristine target machine to the group.

-d *description*

Provides the specified description of the group.

-l *label* Provides the specified label for the group or specifies to which group the action applies.

-m Lists the members of a specified group.

-r *member*

Removes the specified member from the group.

Authorization

The **admin** role is required for all **wpristinegroup** commands.

Examples

1. To create a group called **LowerManhattan** with a description, enter the following command:

```
wpristinegroup create -l \
LowerManhattan -d "Wall St offices"
```
2. To create a group called **LowerManhattan** with the machines **WallSt**, **Cortlandt**, and **WaterSt**, enter the following command:

```
wpristinegroup create -l LowerManhattan -a WallSt -a Cortlandt -a WaterSt
```

wpristinegroup

3. To add a machine called **WallSt** and remove a machine called **Cortlandt** from the group called **LowerManhattan**, enter the following command:

```
wpristinegroup edit -l LowerManhattan -a WallSt -r Cortlandt
```
4. To list existing groups, enter the following command:

```
wpristinegroup list
```
5. To list the machines that belong to the group called **LowerManhattan**, enter the following command:

```
wpristinegroup list -l LowerManhattan -m
```

See Also

None.

wpristinemachine

Creates, modifies, lists, and deletes target machines.

Syntax

```
wpristinemachine create -s server_name -l endpoint_label [-t region_name] [-d description] [-h hostname] [-n computer_name] [-mac mac_address] [-guid smbios_id] [-m ignore | force | if_diff | if_newer] [-u endpoint_setup_path] [-p endpoint_port] [-g gateway_port] [-o endpoint_options] [-k key=value]... [-a group]...
```

```
wpristinemachine delete -l endpoint_label
```

```
wpristinemachine edit -l endpoint_label [-s server_name] [-t region_name] [-d description] [-h hostname] [-n computer_name] [-mac mac_address] [-guid smbios_id] [-m ignore | force | if_diff | if_newer] [-u endpoint_setup_path] [-p endpoint_port] [-g gateway_port] [-o endpoint_options] [-k key=value]... [-z key]... [-a group]... [-r group]...
```

```
wpristinemachine export [-s server_name] [-f file] [-header]
```

```
wpristinemachine import [-f file] [-auto]
```

```
wpristinemachine list -l endpoint_label
```

Description

The **wpristinemachine** command creates target machines, modifies the properties of target machines, deletes target machines, and lists target machines. It also imports and exports target machines in a CSV format.

Options

The following lists the subcommands of the **wpristinemachine** command:

create Creates a target machine.

delete Deletes a target machine.

edit Modifies the properties of the target machine.

export Exports target machines from the Pristine Manager. Contrast with “wpristineexport” on page 264.

import

Imports the machines listed in a CSV file into the machine database. The CSV file lists the following values. Values in bold are required. Even though you must list all the fields, you can leave them blank unless otherwise noted. Repeat the format for each machine.

```
label,region_name,machine_data,server_name,description,  
mac_address,smbios_guid,hostname,computer_name,  
ip_addr,subnet_mask,gateway_addr,  
dns_addr,ep_port,ep_options,ep_gwport
```

Notes:

1. The endpoint label is required but can be assigned automatically by using the **-auto** option.
2. The value for *machine_data* is reserved for Pristine Manager and should not be changed.

3. Some of the values correspond to keywords that are listed under “Defining Environment Variables” on page 252.

list Lists the properties of the specified machine.

The following lists the options and variables belonging to the subcommands:

- a group**
Adds the machine to a group. You can specify multiple groups.
- auto** If you do not specify a machine label when you import from a CSV file, the default is the value specified for the hostname or the computer name.
- d description**
Provides the specified description of the machine.
- f file** Imports or exports the list of machines to the specified file in CSV format. If you do not specify a file, the standard output is used to export and the standard input is used to import the list of machines.
- g gateway_port**
Specifies the port for the Tivoli gateway to which machine endpoint logs in. Default: 9494.
- guid sbmbios_id**
System Management BIOS Globally Unique identifier.
- header**
Provides column titles for the fields as a header in the CSV file.
- h hostname**
Specifies the hostname of the machine. Default: endpoint label.
- k key=value**
Specifies a keyword and value. For example, `-k computer_name=cortlandt`. For a descriptive list of predefined keywords, see “Defining Environment Variables” on page 252.
- l endpoint_label**
Provides the specified label for the machine. Required.
- m** Displays whether the pristine installation should proceed if the operating system is already installed on the machine. It can be one of the following:
 - ignore** The machine is not checked for an operating system and the installation is not performed. Default.
 - force** The operating system is installed in all cases.
 - if_diff** If different. The operating system is installed only if it is different from the one already installed.
 - if_newer**
If newer version. The operating system is installed only if the images contain a newer version than the one already installed.
- mac mac_address**
Of network adapter used for pristine installation.
- n computer_name**
Specifies the name of the machine.
- o endpoint_options**
Specifies options used when setting up a Tivoli endpoint. For example, `-o -gpriestine_gw+9694`.

- p *endpoint_port***
Specifies the endpoint port for the machine. Default: 9495.
- r *group***
Removes the machine from specified group.
- s *server_name***
Specifies the server with which the machine or machine list is associated.
- t *region_name***
Specifies the Tivoli region to which the endpoint installed on the pristine machine will belong. The region must be installed on a machine on which a pristine server has been installed.
- u *endpoint_setup_path***
The path on the server where the endpoint setup files are stored.
- z *key*** Removes a keyword definition.

Authorization

The **admin** role is required for all **wpristinemachine** commands.

Examples

1. To create a machine called **WaterSt**, associated with the server **ris1smith**, that is part of the group **lowermanhattan** and part of the region **newyork-region**, enter the following command:

```
wpristinemachine create -s ris1smith -l WaterSt -t newyork-region \
-a lowermanhattan
```
2. To create a machine called **WaterSt**, associated with the server **ris1smith**, with an endpoint port of 9496 and a screen 800x600 pixels, enter the following command:

```
wpristinemachine create -s ris1smith -l WaterSt -p 9496 \
-k PRISTINE_SCREENWIDTH=800 -k PRISTINE_SCREENHEIGHT=600
```
3. To change the server with which the machine **WaterSt** is associated to **ris2jones**, enter the following command:

```
wpristinemachine edit -l WaterSt -s ris2jones
```
4. To export the list of machines associated with the server **ris2jones** into a CSV file called **nymachines.csv**, enter the following command:

```
wpristinemachine export -s ris2jones -f nymachines.csv
```
5. To list the properties of the machine **WallSt**, enter the following command:

```
wpristinemachine list -l WallSt
```

See Also

None.

wpristinesrv

Works with servers.

Syntax

```
wpristinesrv create -s server -e endpoint_label -t ADS | RIS [-d description] [-u endpoint_setup_path] [-k key=value]
```

```
wpristinesrv delete -s server
```

```
wpristinesrv edit -s server [-d description] [-u endpoint_setup_path] [-e endpoint_label] [-k key=value]... [-t ADS | RIS] [-z key]...
```

```
wpristinesrv list [-s server] [-e endpoint_label] [-t ADS | RIS] [[-env] [-i] | [-csv [-header]]]
```

Description

The **wpristinesrv** command creates, modifies, deletes, and lists servers.

Options

The following lists the subcommands of the **wpristinesrv** command:

create Creates a server.

delete Deletes a server.

edit Modifies the properties of a server.

list Lists servers and their associated machines or images.

The following lists the options and variables belonging to the subcommands:

-csv Displays output in CSV format. If you specify this option, the **-env** and **-i** options are ignored.

-d *description*

Provides the specified description of the server.

-e *endpoint_label*

Provides the specified label for the server. Required.

-env Lists the environment variables defined for the servers.

-header

Provides column titles for the fields as a header in the CSV file.

-i Lists images defined on the server

-k *key=value*

Specifies a keyword and value. For example, **-k PRISTINE_COLORDEPTH=24**. The variables defined for the server can be passed onto the machines associated with the server. For a descriptive list of predefined keywords and an explanation about how to define your own keywords, see “Defining Environment Variables” on page 252.

-s *server_name*

Specifies the name of the server.

-t **ADS | **RIS****

Specifies the server type: **ADS** or **RIS**.

-u *endpoint_setup_path*

Specifies the path on the server where the endpoint setup files are stored.

-z *key* Removes a keyword definition.

Authorization

The **admin** role is required for the **wpristinesrv** command.

Examples

1. To create an RIS server called **ris1smith** with an endpoint label of the same name, enter the following command:

```
wpristinesrv create -s ris1smith -e ris1smith -t RIS
```
2. To specify variables (a screen that is 800 x 600 pixels) that will be passed down to the machines associated with the server **ris1smith**, enter the following command:

```
wpristinesrv edit -s ris1smith -k PRISTINE_SCREENWIDTH=800 \  
-k PRISTINE_SCREENHEIGHT=600
```
3. To list the environment variables defined on the server **ris1smith**, enter the following command:

```
wpristinesrv list -s ris1smith -env
```

See Also

None.

wpristinesrv

Chapter 12. Troubleshooting

This chapter provides information to help you do the following when using the Pristine Manager tool:

- Gather information to analyze and solve problems
- Avoid problems
- Work around problems

Gathering Trace Information

The trace file `pdaemon.log` is located in the `$DBDIR` directory of the Tivoli region where the Pristine Manager daemon is running. It provides information about installations anytime from when you submit the plan to when the pristine machine logs in as a Tivoli endpoint. To gather trace information, you must specify the maximum size of the trace file and the level of detail that the file should contain.

In the same directory, you can find the `pmanager.log` trace file where all the Pristine Manager actions are logged. The trace level of the log file cannot be customized. In the `msg_cmosep.log` and `trace_cmosep.log` files, which are located in the `c:\Program Files\IBM\Tivoli\common\CBI\logs` directory of the endpoint that is installed on the pristine server, you can find all the information about the Pristine Manager activities performed on the pristine server.

The trace level can be customized by editing the file `C:\Program Files\ibm\tivoli\common\cfg\cmosep.properties` and by changing the level values for the `cmosep.loggers.msgLogger.level` and `cmosep.loggers.trcLogger.level` parameters.

To specify the maximum size of the `pdaemon.log` file, use the **wpristine set trace_size** command. To specify the level of detail, use the **wpristine set trace_level** command. See “wpristine” on page 262 for details about both subcommands.

Trace information about the pristine installation when it is part of a reference model or an activity plan is included in the trace files associated with the Change Manager and Activity Planner, respectively. See “Change Manager Traces” on page 213 and “Activity Planner Logs and Traces” on page 131.

You can also enable tracing for the Pristine Manager RIM object typing the following command:

```
wrimtrace RIM_object_label ERROR|INFORMATION
```

where

RIM_object_label

Specifies the RIM object you want to trace. The default name of the Pristine Manager RIM object is `pristine`.

For more information on the **wrimtrace** command, refer to *Tivoli Management Framework: Reference Manual*.

The tracing function is intended for debugging purposes. If enabled for extended periods of time, tracing can decrease performance and slow the processing of the product considerably.

PRISTINE_TMA_SETUP_PATH and Sharing Access to Endpoint Setup Files

The reserved variable PRISTINE_TMA_SETUP_PATH is the path where the installation files of the endpoint are located. For successful installation of the endpoint, this path must be a network share to everyone with full-control access, because the pristine engine copies all of the endpoint response files for each pristine machine there. For example, if you copy all of the endpoint installation files, such as setup.exe, setup.iss and so on, to the c:\lcf directory of a pristine server named ris1jones, you must share this folder to everyone with full-control access and define the variable in Pristine Manager as follows:

```
PRISTINE_TMA_SETUP_PATH=\\ris2jones\lcf
```

However, if you do not want to share the folder with full-access control, use one of the alternatives in the following sections.

Installing from a RIS Server without Sharing the Folder

If you are installing from a RIS server, perform the following steps:

1. Copy the endpoint installation files to a subdirectory, as shown in the following example:

```
\\ris2jones\reminst\setup\lcf
```

2. Define the variable with the same subdirectory as follows:
PRISTINE_TMA_SETUP_PATH=\\ris2jones\reminst\setup\lcf

Note: At the pristine machine site, as soon as the operating system is installed, you are prompted to log on to the network share before the endpoint setup program starts. Specify a user that is a member of the Administrator group, which is the authorization required for installing endpoints.

To avoid the network logon and install unattended, set the PRISTINE_GUIRUNONCE environment variable as follows:

```
PRISTINE_GUIRUNONCE=net use \\< ris2jones >\reminst \  
user_password /USER:user_name
```

Installing from an ADS Server without Sharing the Folder

If the operating system image has already been captured, perform the following steps:

1. Copy the endpoint installation files to the c:\windows\temp directory of the ADS server (or c:\winnt\temp if the pristine machine will be a Microsoft Windows 2000 machine).
2. Define the variable as follows:
PRISTINE_TMA_SETUP_PATH=c:\windows\temp

(or c:\winnt\temp)
3. Modify the tivdeployimage.xml file in the cache directory (or at the Tivoli region site directly) by adding the following tasks after the Download image task:

```

<task description="copy endpoint installation file1 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
    <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\%PRISTINE_DEVICEID%.iss</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\%PRISTINE_DEVICEID%.iss </parameter>
    </parameters>
</task>
<task description="copy endpoint installation file2 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\data1.hdr</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\data1.hdr</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file3 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\data2.cab</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\data2.cab</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file4 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\ikernel.ex_</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\ikernel.ex_</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file5 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\layout.bin</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\layout.bin</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file6 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\setup.bmp</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\setup.bmp</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file7 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\Setup.exe</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\Setup.exe</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file8 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>

```

```

        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\Setup.ini</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\Setup.ini
</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file9 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\setup.inx</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\setup.inx
</parameter>
    </parameters>
</task>
<task description="copy endpoint installation file10 to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\%PRISTINE_WINDOWSDIR%\temp\data1.cab</parameter>
    <parameter>\device\harddisk0\partition1\%PRISTINE_WINDOWSDIR%\temp\data1.cab
</parameter>
    </parameters>
</task>

```

If the operating system image has *not* yet been captured, perform the following steps:

1. Copy the endpoint installation files to the c:\lcf directory of your master device before capturing the image.
2. Define the variable as follows:
PRISTINE_TMA_SETUP_PATH=c:\lcf
3. On the ADS server, create the directory c:\lcf and copy only the endpoint setup.iss file to it.
4. Modify the tivdeployimage.xml file in the cache directory by adding the following task after the Download image task:

```

<task description="copy endpoint installation file to the target">
    <command>bmonitor/bmfilexfer.exe</command>
    <parameters>
        <parameter>-d</parameter>
        <parameter>C:\LCF\%PRISTINE_DEVICEID%.iss</parameter>
        <parameter>\device\harddisk0\partition1\LCF\%PRISTINE_DEVICEID%.iss
    </parameter>
    </parameters>
</task>

```

Adding Software Distribution or Inventory Activities to a Plan

If you add a Software Distribution or Inventory activity to a plan with a Pristine Manager activity for a target whose endpoint label matches the label of an endpoint that is already present in an interconnected region, or that will log onto an interconnected region, the pristine installation succeeds and does not fail. However, the Software Distribution or Inventory activities that follow and the plan itself fail. To avoid this problem, ensure that you do not assign duplicate endpoint labels in interconnected regions or that you put the Software Distribution or Inventory activity in a separate plan, when the pristine target endpoint has to log onto an interconnected region.

Configuration Changes after Installing Pristine Manager

If, after you install Pristine Manager, you reinstall Activity Plan Monitor or you change the RIM, the Tivoli Enterprise Console EventServer, or any Pristine Manager daemon configuration details, for example, enabling Tivoli Enterprise Console notification or lowering the trace level, you must restart the Pristine Manager daemon. See a description of the **wpristine restart** (“wpristine” on page 262).

Failure of Installation

If the RIS or ADS server fails to install an operating system, no error message is reported to the Activity Plan Monitor. The plan fails when the pristine installation times out.

To set the timeout period, in hours, use the **wpristine set timeout** command. See “wpristine” on page 262 for details.

Avoiding Failure

For images on ADS servers, ensure that the image name includes a substring of either 2000 or 2k.

ADS Certificate

If you move the ADS certificate file `adsroot.cer` from the default directory `c:\Program Files\Microsoft ADS\certificate` on the sever, you must define an environment variable named `PRISTINE_ADSCERTIFICATE` with the directory in which it is located. You can define this environment variable either as a server-inherited variable or as a machine-defined variable. For example, if the `adsroot.cer` file is located in `c:\ads\certificate` and you want to configure all the machines to locate the file in that directory, define an environmental variable in the server properties as follows:

1. In the Key field, enter **PRISTINE_ADSCERTIFICATE**.
2. In the Value field, enter **c:\Program Files\Microsoft ADS\certificate**. The `adsroot.cer` file will be installed on all machines associated with this server in the specified directory.

Installing in Interconnected Regions

If a Software Distribution or Inventory activity follows a pristine installation activity in a plan launched from a hub Tivoli region of interconnected regions, the Software Distribution or Inventory activity fails. The Tivoli server cannot access the endpoint in the interconnected region until you run the **wupdate** command. To avoid this problem, enable the read-through feature using the **wregister -fr** command. This turns an exchangeable resource type (the endpoint) into a remote resource type which is visible to the server in the hub region. The endpoint on the new machine can log in and the plan and the pristine installation end successfully. For more information on hub and spoke Tivoli regions, refer to *Tivoli Management Framework: Planning for Deployment Guide*. For more information about the read-through feature, see the **wregister** command in the *Tivoli Management Framework: Reference Manual*.

Installing Microsoft Windows 2003 from an ADS Server

If you install a Microsoft Windows 2003 operating system on a pristine machine from an ADS server and you assign the machine a static IP address, the final reboot of the installation job fails with the following message:

Device or service connection does not exist.

The machine completes the job by connecting to the Deployment Agent. It connects to the ADS server with a dynamic IP, which is different from the static one that was assigned and the reboot task fails.

To avoid this problem, besides assigning a static IP address, have the DHCP server reserve the static IP address for the MAC address of the pristine machine before the installation is launched. The DHCP server will assign the IP address to the machine that connects with the specified MAC address. The Deployment Agent will reconnect to the same IP address without an error.

Installation from RIS Server Successful but Activity Plan Fails

Problem: The activity plan fails even though the operating system and endpoint are installed successfully on the pristine target from a RIS server. You receive the following symptoms:

- The pdaemon.log contains a message, reporting that the machine was not successfully checked.
- A trace of the endpoint reveals that the SMBIOS ID could not be identified.
- If you are using Tivoli Enterprise Console, the event CBIPD0020I Machine endpoint logged in successfully is sent; however, you also receive the event CBIPD0015E Unable to verify machine (the miniscan downcall failed).

Solution: In the Pristine Manager machine database, modify the machine to include a MAC address as well as an SMBIOS GUI ID.

Specifying a Static IP Address

If you are installing from a machine running Microsoft Windows 2000 Server, Microsoft Windows 2000 Advanced Server, or Microsoft Windows 2000 Professional, all with Service Pack 1 and you specify a static IP address and subnet mask, the IP address for the machine is assigned dynamically from the DHCP server.

Part 4. Web Interface

Chapter 13. Administering the Web Interface	281	Configuring Trace Settings	301
Setting up the Web Interface	281	Configuring Trace Parameters	302
Adding the WebUI_Admin Role	281	wwebcfg	303
Registering Plug-ins	282	Troubleshooting	305
wwebplugin.	283	Administrator on the Tivoli Server	305
Enabling Java Plug-in Automatic Download	285	Web Interface User	306
Making Web Objects Available.	285	Chapter 14. Using the Web Interface	309
Publishing and Unpublishing Web Objects	286	Starting the Web Interface	309
wwweb	288	Web Interface Window	311
Unrestricted Access to Web Objects	292	Operations Console	311
Determining What Has Been Published.	292	Using Web Objects	312
Example of		Software Packages.	313
PUBLISHED_PACKAGES_QUERY Output	293	Installing Software Packages	313
Example of PUBLISHED_REFMODS_QUERY		Verifying Software Packages	314
Output	294	Uninstalling Software Packages	315
Providing Information and Assistance for Web		Running Inventory Scans	316
Interface Users	294	Reference Models	317
Configuring the Web Interface Window	295	Viewing Reference Models	317
Enabling User Context Switch.	296	Applying (Synchronizing) Reference Models	319
Monitoring the Results of Operations	298	Troubleshooting	321
Troubleshooting	300		
Tivoli Web Gateway Logs and Traces for the			
Web Interface	300		

This part describes the Web Interface which provides, for users who can access a Tivoli region network only temporarily, using a Web browser, a method by which they can perform some of the activities of Software Distribution, Inventory, and Deployment Services. This Web Interface documentation is divided into two chapters.

- Chapter 13 is written for the administrator. It describes how the administrator can publish and unpublish Web objects such as software packages, inventory profiles, and reference models. The administrator also has tasks to set up the Web Interface and to perform diagnosis.
- Chapter 14 is written for the Web user accessing Web Interface by means of a browser. It explains how the Web user can use these Web objects to maintain the software and provide information about the user's computer.

Locating Related Information

Information related to this service is available from the following sources:

- *Introducing IBM Tivoli Configuration Manager*. This provides an overview of the service.
- *Planning and Installation Guide*. This includes information about how you install the service.
- *Messages and Codes*. This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 13. Administering the Web Interface

This chapter describes the administrator's tasks for setting up the Web Interface and maintaining it. A description of how the Web user uses the interface to perform selected inventory and software distribution activities is in Chapter 14, "Using the Web Interface," on page 309.

Using the Web Interface, the administrator can:

- Make Web objects, such as software packages, inventory profiles, and reference models, available to Web Interface users in a secure or unrestricted environment
- Monitor the results of the operations performed by the Web users
- Carry out troubleshooting to resolve problems
- Enable Web users to perform install, uninstall, and verify operations on a software package even if they are not logged on as administrators, for the supported Windows operating systems. For a list of supported operating systems, refer to *IBM Tivoli Configuration Manager: Release Notes*.

For more information about Web objects, refer to the documentation on how to create and maintain them:

Software packages

See *IBM Tivoli Configuration Manager: User's Guide for Software Distribution*.

Inventory profiles

See *IBM Tivoli Configuration Manager: User's Guide for Inventory*.

Reference models

See Part 2, "Modeling the Enterprise Configuration," on page 145.

Setting up the Web Interface

If you did not install the Web Interface on the Tivoli server using the InstallShield wizard, you need to ensure that:

- The **WebUI_Admin** role has been added on the Tivoli server
- The correct plug-ins have been registered

Adding the WebUI_Admin Role

If you intend to use the **wweb** command to publish and unpublish Web objects, ensure that the **WebUI_Admin** role has been added on the Tivoli server. To do this perform the following steps:

1. Open the Tivoli Desktop.
2. Double-click the **Administrators** icon. The Administrator window is displayed.
3. Right-click the **Root Administrator** icon and select **Edit TMR Roles** from the pop-up menu. The **Available Roles** and **Current Roles** lists are displayed.
4. To add the **WebUI_Admin** role, select the role from the **Available Roles** list.
5. Double-click the left arrow button. The role moves from the **Available Roles** list to the **Current Roles** list.
6. Click **Change & Close** to add the **WebUI_Admin** role for the administrator. The Administrators window is displayed again.

Registering Plug-ins

If you install a component (Software Distribution or Inventory) on the Tivoli server after you install the Web Interface, the respective Web Interface plug-in is registered automatically. However, if you install a component before installing the Web Interface, you must register the plug-in manually by using the **wwebplugin** command (there is no GUI equivalent) with the following syntax, depending on the component:

Software Distribution:

```
wwebplugin -register -c SoftwarePackage -p publish
```

Inventory:

```
wwebplugin -register -c InventoryConfig -p ic_publish
```

Full details of the command syntax are given in the following pages.

wwebplugin

Registers or unregisters a Configuration Manager component plug-in with the Tivoli server.

Syntax:

wwebplugin -register -c *profile_class_name* -p *publish_method*

wwebplugin -unregister -c *profile_class_name*

Description: Manually registers or unregisters a Configuration Manager component plug-in such as for Software Distribution or Inventory.

Options:

{ -register | -unregister }

Registers or unregisters the plug-in.

Note: **-register** or **-unregister** must be the first option.

-c *profile_class_name*

The profile class name of the plug-in, as follows:

Software Distribution	SoftwarePackage
Inventory	InventoryConfig

-p *publish_method*

The name of the function that implements the publish operation, as follows:

Software Distribution	publish
Inventory	ic_publish

Note: Specify the publish method correctly, because no checks are made on the validity of the entry. If you make a mistake you will register a non-existent publish method, and your subsequent Web Interface operations on the class you intended to register will fail.

This is required for **-register**.

Authorization: None.

Return Values: The **wwebplugin** command returns one of the following values:

- 0** The operation is successful.
- 1** The operation is unsuccessful.

Examples:

1. To register a profile class name called SoftwarePackage with a publish method of publish:

```
wwebplugin -register -c SoftwarePackage -p publish
```

The output is:

```
IWGSR0009I Application class name <SoftwarePackage> - Registration as Web Interface application succeeded.
```

2. To unregister a profile class name called SoftwarePackage:

```
wwebplugin -unregister -c SoftwarePackage
```

wwebplugin

The output is:

```
IWGSR0011I Application class name <SoftwarePackage> - Unregistration as Web  
Interface application succeeded.
```

3. To register a profile class name called InventoryConfig with a publish name method called ic_publish:

```
wwebplugin -register -c InventoryConfig -p ic_publish
```

The output is:

```
IWGSR0009I Application class name <InventoryConfig> - Registration as Web  
Interface application succeeded.
```


Enabling Java Plug-in Automatic Download

You can enable the Java plug-in automatic download feature on web user machines by editing the `webconsole.properties` file stored on the Web Gateway machine in the following path: `$WAS_HOME/installedApps/node_name/WebConsole.ear/WebUI.war/WEB-INF/classes`, where:

`$WAS_HOME`

is the WebSphere installation directory

`node_name`

is the name of the node where WebSphere is installed

In this file, perform the following operations:

1. Set the `PLUGIN_AUTH_DOWNLOAD.ENABLED` key to `true`
2. Set the `PLUGIN_DOWNLOAD_URL.os_name` key to the http URL where the installable binaries for the java plug-in are located, where

`os_name`

is the name of the operating system you are using.

3. Stop and restart the WebUI_AppServer.

The automatic download feature is supported on the following browsers:

- Internet Explorer
- Mozilla

The automatic installation of the Java plug-in is available only with Internet Explorer. The minimum Java Virtual Machine required version is 1.4.x.

Making Web Objects Available

The Configuration Manager Web Interface allows Web users to perform operations using Web objects. Web objects can be:

- Software packages
- Inventory profiles
- Reference models

Before a Web object can be used, it has to be published. This process grants access rights to those users who want to access the object, and copies it to a server from where it can be accessed by means of the Web. The Web Interfaces guarantees data and access security for Web objects by means of the IBM Tivoli Access Manager and IBM Tivoli Access Manager WebSEAL security mechanisms. For more information on using the Web Interface with IBM Tivoli Access Manager, refer to *IBM Tivoli Configuration Manager: Planning and Installation Guide*.

You can also use the Web interface in an unrestricted environment, so that no authentication is performed when accessing Web objects. When publishing Web objects, the administrator can choose to not specify user information, so that the objects published can be accessed by all users. For more information about publishing objects with unrestricted access, see “Unrestricted Access to Web Objects” on page 292.

This section explains how to publish and unpublish Web objects, and how to find out what is currently published. It also discusses the steps necessary to inform the users of the availability of the Web objects.

Publishing and Unpublishing Web Objects

To publish and unpublish Web objects use the **wweb** command (there is no GUI equivalent). This command allows you to give access to a specified Web object to one user, several users, a list of users, or to grant unrestricted access to all users.

Note: Access is given to users not computers. A user can access the Web Interface from more than one computer and perform different or the same activities at each computer, provided that the same user name is used.

A Web object is published on one or more endpoints where the Web Infrastructure is installed, as described in *IBM Tivoli Configuration Manager: Planning and Installation Guide*. The Web Interface user accesses this server using a Web browser that supports Java applets, and logs on using an ID and password that you provide.

The command allows you to choose whether, when publishing, you want both to give access rights and to copy the Web object onto the depot at the server. For example, the first time you publish an object you must do both. Subsequently, if you want to add users you would use the command without copying the object to the server depot. However, if you wanted to replace the object, you would force download the object (using the **-f** option); you would still have to specify the access rights (using the **-u** option) for all the users who needed to access that object, if you want to maintain user authentication restrictions. Otherwise, you can omit the **-u** option, to grant unrestricted access to Web objects.

Software packages intended for use with the WebUI should be created taking into account that user-defined variables, or command-line variables, are not supported by the WebUI.

The WebUI searches for any variable in the package among the embedded Software Distribution variables in the order specified in "Using Variables" in the *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*. If the variable is not among the embedded Software Distribution variables, it is not resolved. In this case, the default value defined at creation time, if any, is used when the software package is installed.

When using the unpublish option, you can opt to unpublish selected users without removing the object, or to remove an object completely.

When publishing reference model objects, you must ensure that all Software Distribution and Inventory objects referenced in the reference model are also published in the same way to the same users. In particular, with respect to Software Distribution, this means considering all possible software packages that might be required to upgrade the user's software objects from their current state to the desired state. In addition, check that published reference models do not contain references to operations that are not Software Distribution or Inventory operations.

To manage reference models in a secure environment, you need to perform the following steps:

Note: The procedure described below applies to Internet Explorer. If you use a different browser, the panels described might be different.

1. Ensure that the junction on the WebSEAL workstation was created using the **server task create** command with the **-k** option.

2. When connecting to the WebSEAL server, click the **View Certificate** button in the Security Alert dialog box displayed. The Certificate dialog box is displayed.
3. Select the **Details** tab.
4. Click **Copy to File**. The Certificate Export Wizard starts. Click **Next**.
5. Select the **Base-64 encoded X.509 (.CER)** radio button. Click **Next**.
6. Specify a location for the file to be exported. You must save the file to the workstation where WebSphere is installed.
7. Click **Finish**.
8. Run the following command on the workstation where WebSphere is installed to import the WebSEAL certificate into the WebSphere keystore (cacerts):

```
$WAS_HOME\java\jre\bin\keytool -import -file certificate -keystore  
$WAS_HOME\java\jre\lib\security\cacerts
```

where

\$WAS_HOME

Is the WebSphere installation directory

certificate

Is the name of the file in which you saved the certificate

9. Restart the WebUI_AppServer application server to make the changes effective.

Note: The sec_master user, which is created when configuring the WebSEAL server, does not have the authority to publish software packages, because it is an administrative user. To publish software packages, create a different user using the pdadmin command. For more information on this command and on configuring the WebSEAL server, refer to *IBM Tivoli Access Manager for e-business Command Reference*.

Full details of the **wwweb** command are given in the following pages:

wweb

Publishes a software package, reference model, or inventory scan as a Web object.

Syntax:

```
wweb -publish -p publicName -v version {-i all | -i interp } ... \
-w app_server_ep_label [-w app_server_ep_label] ... [-c connspeed] \
  [-u all | [-u user ] ... ] \
  [-U file] ... [-f | -n] \
@profile
```

```
wweb -unpublish -p publicName -v version -w app_server_ep_label \
  [-w app_server_ep_label]...
[-u all | [-u user ] ... ] [-U file] ... [-f | -n] @profile
```

Description: Provides users or groups of users with access rights to one or more Web objects, and optionally copies the object onto the identified WebSphere Application server or servers.

Options:

{ -publish | -unpublish }

Publish or unpublish the profile. Required.

-p *publicName*

The public name of the published Web object. It must start with a forward slash (/). Required.

-v *version*

The version of the published Web object. Required.

{-i all | -i interp } ...

The platforms (interps) on which the Web object works (-publish operation only, at least one is required).

-w *app_server_ep_label* [**-w** *app_server_ep_label*] ...

The labels of the endpoints running the WebSphere Application Server where the Web object is to be published. At least one endpoint is required.

[-c connspeed]

The minimum connection speed in Bps (bytes per second) suggested for the client to download the profile (-publish operation only).

[-u all | [-u user] ...]

Users allowed to access the Web object from the Web, where a user is an Access Manager account name created using WebSEAL. The default value is **- u all**.

Note: If you use the **all** attribute to publish a Web object, you *must* use the **all** attribute when you unpublish it.

[-U file] ...

Absolute path to one or more files containing a list of users.

[-f | -n]

These options change in meaning according to whether the **-publish** or **-unpublish** option has been specified:

-publish

The **-f** option forces the Web object to be downloaded to the server. The **-n** option ensures that the Web object is *not* downloaded to the server.

Default: If neither option is supplied, the behavior depends on the type of object:

Software object

A software package Web object is downloaded *only if it is not already present* at the server.

Inventory scan

An inventory scan Web object is *always* downloaded to the server.

Reference model

A reference model Web object is downloaded *only if it is not already present* at the server.

-unpublish

The **-f** option forces the Web object to be unloaded from the server. The **-n** option ensures that the Web object is *not* unloaded from the server.

Default: If neither option is supplied, the behavior depends on the type of object:

Software object

A software package Web object is *not* unloaded from the server.

Inventory scan

An inventory scan Web object is *always* unloaded from the server.

Reference model

A reference model Web object is *not* unloaded from the server.

@profile

The name of the profile to be published as a Web object. The full format of the profile name is as follows:

@profile_class:profile_name#region

where:

@ Atsign; required

profile_class:

The name of the profile class. Supported values are as follows:

- SoftwarePackage
- InventoryConfig
- ReferenceModel

Only required to resolve ambiguity, for example, if there are two profiles in different classes with the same name. If a *profile_class* is specified, it must be separated from the *profile_name* by the colon (:) delimiter.

profile_name

Name of profile being published. Required. If the profile is a

Reference Model, separate the name from the version number by one caret (^) in a UNIX environment and two carets (^^) in a Windows environment.

#region The name of the region in which the profile was created. Only required to resolve ambiguity, for example, if there are two profiles of the same class with the same name in two different regions. If a region is specified, it must be separated from the profile_name by the number sign (hash) delimiter (#).

Some examples of profile specifications:

@test.1.0

Simple profile definition with no ambiguity.

@SoftwarePackage:test.1.0

Profile definition specifying profile class, to resolve ambiguity with an Inventory profile with the same profile name.

@test.1.0#my_region

Profile definition specifying region, to resolve ambiguity with a profile with the same profile name in a different region.

@SoftwarePackage:test.1.0#my_region

Profile definition specifying profile class and region, to resolve ambiguity with an Inventory profile with the same profile name in the same region, and another profile with the same name in a different region.

@test^^model.1.0

Reference Model profile definition in a Windows environment.

@test^model.1.0

Reference Model profile definition in a UNIX environment.

Authorization: WebUI_Admin

Return codes: The **wwweb** command returns one of the following values:

- 0** The operation is successful.
- 1** The operation is unsuccessful.

Examples:

1. To publish to an endpoint called name1-ep, a profile called test.1.0, in a Web category called cat1, with a public name of test, and a version of 1.0, for all Web clients interps, and for all users, with a minimum connection speed of 9600 bps:

```
wwweb -publish -p /cat1/test -v 1.0 -w name1-ep -i all -u all -c 9600 @test.1.0
```

If the profile is a software package, the output is:

```
DISSE0074I Operation successfully submitted. Distribution ID is 1198304159.12.
```

Note: If the **-f** option is not used, the profile is sent to the endpoint only if it has not been sent before.

2. To perform the same operation as previously, but in force mode:

```
wwweb -publish -p /cat1/test -v 1.0 -w name1-ep -i all -u all -c 9600 -f
@test.1.0
```

Note: The **-f** option is used here to force the distribution, even if the profile has been sent to the same endpoint previously.

3. To perform the same operation, but to two endpoints, name1-ep and name2-ep:

```
wwweb -publish -p /cat1/test -v 1.0 -w name1-ep -w name2-ep -i all -u all -c 9600 @test.1.0
```

If the profile is a software package, the output is:

DISSE0074I Operation successfully submitted. Distribution ID is 1198304159.12.

4. To publish to an endpoint called name1-ep, a profile called test.1.0, in a Web category called cat1, with a public name of test, and a version of 1.0, for all Web clients interps with a minimum connection speed of 9600 bps with unrestricted access:

```
wwweb -publish -p /cat1/test -v 1.0 -w name1-ep -i all -c 9600 @test.1.0
```

If the profile is a software package, the output is:

DISSE0074I Operation successfully submitted. Distribution ID is 1198304159.12.

Note: If the -f option is not used, the profile is sent to the endpoint only if it has not been sent before.

5. To publish to an endpoint called name1-ep, a profile called test.1.0, in a subcategory called subcat1, of a main category called cat1, with a public name of profileTest, and a version 1.0, for the w32-ix86 and Solaris2 Web clients interps, and for a list of users (a text file with the name of users separated by blanks), with a minimum connection speed of 9600 bps:

```
wwweb -publish -p /cat1/subcat1/profileTest -v 1.0 -w name1-ep -i w32-ix86 -i solaris2 -U /tmp/users -c 9600 @test.1.0
```

6. To unpublish to an endpoint called name1-ep, a profile called test.1.0, published in a Web category called cat1, with a public name of profileTest, and a version 1.0, for the users user1 and user2:

```
wwweb -unpublish -p /cat1/profileTest -v 1.0 -w name1-ep -u user1 -u user2 @test.1.0
```

7. To publish to an endpoint called name1-ep, a reference model called managers 1.0, in a Web category called cat1, with a public name of managers, and a version 1.0, for Solaris Web clients interps, and for a user manager1, with a minimum connection speed of 9600 bps:

```
wwweb -publish -p /cat1/managers -v 1.0 -i Solaris2 -w name1-ep -u manager1 -c 9600 @managers^1.0
```

Note: To publish a reference model, Software Distribution and Change Manager must be installed on the Tivoli server, but the administrator does not need to register a plug-in for Change Manager.

Unrestricted Access to Web Objects

You can also use the Web Interface component in a non-secure environment, that is, without installing and configuring the IBM Tivoli Access Manager and IBM Tivoli Access Manager WebSEAL security mechanisms. Running the Web Interface in a non-secure environment means that no authentication is performed when accessing Web objects. When publishing Web object, the administrator can choose to not specify user information so that the object published can be accessed by all users. For more information about publishing objects with unrestricted access, see “wweb” on page 288.

In an existing environment with the IBM Tivoli Access Manager and IBM Tivoli Access Manager WebSEAL security mechanisms configured and installed, you can switch to an unrestricted environment where all Web objects already published and any new objects published become accessible for all users:

1. Change the setting of the `USERS_WEBSEAL_ENABLED` key to false in the `twgConfig.properties` file on the machine where the Web Infrastructure is installed.
2. Stop and restart the `DMS_AppServer`.
3. Stop and restart the `WebUI_AppServer`.
4. Change the Web address used to start the Web Interface, to the unrestricted address as follows:

```
http://hostname/WebConsole/com.tivoli.webui.servlets.ConsoleMain
```

where:

hostname

is the hostname of the machine where Web Infrastructure has been installed. This information is supplied by your system administrator. In this way, you bypass the authentication on the WebSeal machine.

Determining What Has Been Published

To determine which software packages or reference models have already been published in a policy region, you are supplied with the following queries:

PUBLISHED_INV_PROFILES_QUERY

This lists all published Inventory profiles. It runs against the view `PUBLISHED_PROFILES`. It is automatically created by the script `inventory_query.sh`, which is normally run as part of the installation and configuration of Inventory (see *IBM Tivoli Configuration Manager: Planning and Installation Guide*).

PUBLISHED_PACKAGES_QUERY

This lists all software packages that have been published. It is automatically created by the script `inventory_query.sh`, which is normally run as part of the installation and configuration of Inventory (see *IBM Tivoli Configuration Manager: Planning and Installation Guide*).

PUBLISHED_REFMODS_QUERY

This lists all reference models that have been published. It is automatically created when you install Change Manager (see *IBM Tivoli Configuration Manager: Planning and Installation Guide*).

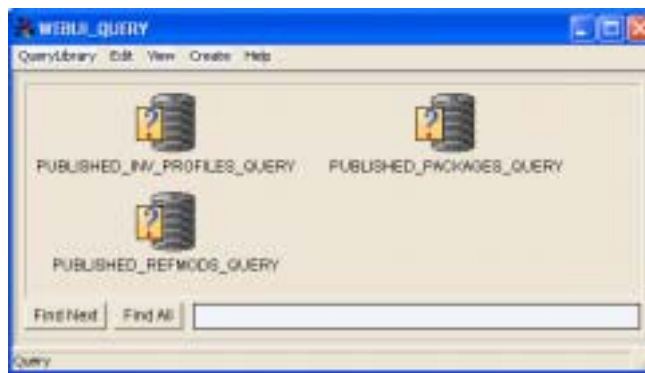
All these queries are created in a query library called `WEBUI_QUERY`.

To run either of these queries, perform the following steps:

1. Open the Policy Region window:



2. Double-click the **WEBUI_QUERY** library icon. The WEBUI_QUERY window is displayed:

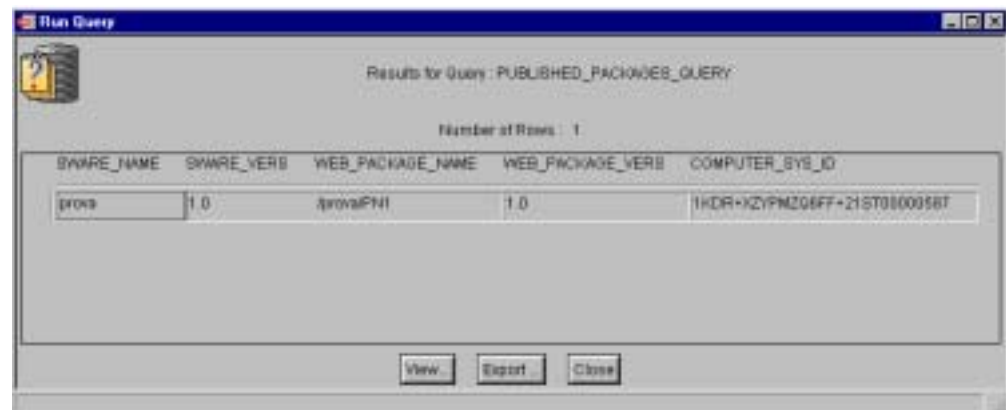


This window shows one or both of the Web Interface queries.

3. To run a query, right-click the required query icon and select **Run Query...**

Example of **PUBLISHED_PACKAGES_QUERY** Output

The following is an example of the output from the **PUBLISHED_PACKAGES_QUERY** query:



The information available for each package is as follows:

- Software package name
- Software package version
- Web package name
- Web package version
- Computer system ID of the endpoint or endpoints running the WebSphere Application Server where the package has been published (as identified in the

Determining What Has Been Published

wwweb command using the **-w** option). Look up this value in the **COMPUTER** table using Inventory to identify the endpoint label.

Example of PUBLISHED_REFMODS_QUERY Output

The following is an example of the output from the **PUBLISHED_REFMODS_QUERY** query:



REF_MODEL_NAME	REF_MODEL_VERSION	WEB_PACKAGE_NAME	WEB_PACKAGE_VERSION	COMPUTER_SYSTEM_ID
WXYZ	1.0	xyzwPnt	1.0	1H0R+XZ7PMZG6FF+21ST0000000T

The information available for each package is as follows:

- Reference model name
- Reference model version
- Web package name
- Web package version
- Computer system ID of the endpoint or endpoints running the WebSphere Application Server where the reference model has been published (as identified in the **wwweb** command using the **-w** option). Look up this value in the **COMPUTER** table using Inventory to identify the endpoint label.

Providing Information and Assistance for Web Interface Users

As administrator, you should provide the following information to your Web Interface users:

Server URL

The user requires the URL of the WebSphere Application Server or servers that can be accessed to perform Web Interface operations in the form:

`https://hostname/junction/WebConsole/com.tivoli.webui.servlets.ConsoleMain`

where:

hostname

Is the host name of the machine where WebSEAL has been installed and configured. This information is supplied by your system administrator.

junction

Is the name of the junction that has been configured in WebSEAL to redirect that Web address to WebSphere. This information is supplied by your system administrator.

Notes:

1. If your environment is configured with unrestricted access to Web objects, type the Web address for the Web Interface server in the form:

`http://hostname/WebConsole/com.tivoli.webui.servlets.ConsoleMain`

where:

hostname

Is the host name of the machine where WebSphere has been

installed and configured. This information is supplied by your system administrator. In this way, you bypass the authentication on the WebSEAL workstation.

No logon dialog is displayed, because no authentication is required.

2. If secure access to Web objects is enabled, the value used to indicate the WebSEAL workstation must exactly match the value of the key `USERS_WEBSEAL_HOST_NAME` contained in the file `twgConfig.properties` stored on the Web UI endpoint and specified at installation time. For example, if you have used the hostname of the WebSEAL workstation in the `twgConfig.properties` file, you cannot use the IP address of the WebSEAL workstation in the URL.

Username and password

Each user requires a username (access ID) and password that you create using WebSEAL.

Note: The `sec_master` user, which is created when configuring the WebSEAL server, does not have the authority to install software packages, because it is an administrative user. To install software packages, create a different user using the `pdadmin` command. For more information on this command and on configuring the WebSEAL server, refer to *IBM Tivoli Access Manager for e-business Command Reference*.

Web objects available

When the user connects to the Web Interface, a list of all available Web objects is displayed. However, you may decide to create a procedure of notification to users, especially in the case where the installation or upgrade of a software item is a matter of enterprise policy.

User assistance

You may wish to provide the pages of Chapter 14, “Using the Web Interface,” on page 309 to the user as an aid when using Web Interface.

Troubleshooting support

If anything goes wrong while the user is performing Web Interface operations, messages or html screens containing information about what has gone wrong are displayed. You may wish to provide an access route for support to which the user can send this information.

Configuring the Web Interface Window

To change the default setting for the **View web objects for all platforms** check box, perform the following steps:

1. Locate the `webconsole.properties` file stored on the Web Gateway machine in the `$WAS_HOME/installedApps/node_name/WebConsole.ear/WebUI.war/WEB-INF/classes` path where:

`$WAS_HOME`

Is the WebSphere installation directory

`node_name`

Is the name of the node where WebSphere is installed

2. Open the `webconsole.properties` file using a text editor.
3. Change the value for the `VIEW_FOR_ALL_INTERPS` key to `true`. Supported values are `true` and `false`.
4. Stop and restart the `WebUI_AppServer` for this change to be effective.

Determining What Has Been Published

You can edit the company information displayed in the Web Interface at startup as described below:

To change the company information, perform the following steps:

1. Open the `webconsole.properties` using a text editor.
2. Add the text you want displayed after the `BANNER.TEXT` key.
3. Change the file name after the `BANNER.LOGO` key to point to a `.gif` file to be displayed in the upper right corner of the Web Interface window.
4. Change the file name after the `HELP.WINDOW` key to point to a `.html` file containing the text to be displayed in the Web Interface window at startup.
5. Stop and restart the `WebUI_AppServer` for this change to be effective.

Note: If the text you specify is entered in a double-byte language, the `webconsole.properties` file must be saved in ANSI format. To perform this operation, specify ANSI in the Encoding pull-down menu in the Save dialog in the text editor.

Enabling User Context Switch

On supported Windows operating systems, to enable the Web user to perform the install, uninstall, and verify operations on a software package even if the web user is not logged on as administrator, the administrator must perform one of the following procedures:

- Log on as Administrator on the Web user workstation and perform the following steps:
 1. Open a browser and connect to the Web interface as described in “Starting the Web Interface” on page 309.
 2. Wait for the Operation Console to initialize. When the initialization completes, the following message is displayed in the Operation Console report area:
Initializing... done
 3. Close the browser.
 4. Edit the `webui.properties` file located in the `%USERPROFILE%` directory.
 5. Change the value for the `ENABLE_UCS_AUTOINSTALL` key to `TRUE`.
 6. Open the browser and connect to the Web interface.
 7. During the Operation Console initialization steps, the following message confirms the correct installation of the User Context Switch functionality:
Downloading UCS bundle ...done
- From the Tivoli server, or from any available managed node for the Web user workstation, perform the following steps:
 1. From the Tivoli desktop, import the `WebUIService_w32-ix86.spb` software package as described in the *User's Guide for Software Distribution*. Import the software package in the profile manager that contains the Web user endpoint as subscriber. The `WebUIService_w32-ix86.spb` software package is located on the Web UI endpoint in the `LCF_ROOT\web\ucs` directory.
 2. From the Tivoli desktop, install the `WebUIService_w32-ix86.spb` software package on the Web user endpoint where you want to enable the User Context Switch service.
 3. On the Web user endpoint, verify that the `WEBUI 4.2 UCS` service is listed in the Services window with the following settings:
 - **Started** in the Status column

- **Automatic** in the Startup Type column
- **LocalSystem** in the Log On As column.

Note: To prevent incorrect use of the Software Distribution disconnected command line by Web users, ensure that the file system of the Web user client is NTFS and that only members of the Administrators group can write to the %WINDIR% directory.

Monitoring the Results of Operations

When a user performs a Web Interface operation, the results of the operation are sent back to the Tivoli WebSphere Application Server. The log file is updated in the working directory of the Tivoli server and the following tables are updated on the workstation where you have installed the Inventory database, as defined by the RIM object:

- SD_INST
- SD_H_INST
- COMPUTER

The software packages distributed using Software Distribution are tracked in the Inventory database by the endpoint label, while the software packages distributed using the Web Interface are tracked by the workstation hostname. This can lead to a misalignment of information in the Inventory database if you install an endpoint on the Web Interface client workstation and distribute software packages to this workstation using Software Distribution.

To avoid this problem, you should install the endpoint with the same name as the workstation hostname or, if the endpoint is already installed, change its name to match the workstation hostname.

If an operation is performed on a workstation that is only a Web Interface client, not an endpoint, only the following fields are populated in the COMPUTER table:

- COMPUTER_SYS_ID
- TME_OBJECT_LABEL
- RECORD_TIME

This item...	Is stored in this field...	Of this table...
Host name	TME_OBJECT_LABEL	COMPUTER
Administrator ID	TME_ADMIN_ID	SD_INST
Execution time	EXEC_TIME	SD_INST SD_H_INST
Record time	RECORD_TIME	SD_INST SD_H_INST
GUID	GUID	COMPUTER

Notes:

1. The execution time is the time it takes for the operation to run on the Web Interface client.
2. The record time is the time when the result of the operation is returned to the Tivoli server.
3. The GUID is a unique numeric string that identifies a machine. If a machine has not previously been scanned, the Web Interface client generates the GUID and the result of the distribution.

The result of the distribution is returned to the Tivoli server within the time interval specified by the Web Gateway component. The default value is five minutes. If, after this time, the log file or the tables are not updated, see “Troubleshooting” on page 300.

For software packages containing restart actions, the report containing the result of the distribution is sent to the Tivoli server after a second operation is performed on the target.

Troubleshooting

This section describes how to configure the trace parameters, and provides a series of possible problems and their solutions, categorized by the activity being performed. The Web Interface provides a variety of logs and trace files.

Tivoli Web Gateway Logs and Traces for the Web Interface

Tivoli Web Gateway provides internal services for the Web Interface. Information concerning problems and errors for the Web Interface is written in the Tivoli Web Gateway logs.

The following log files are available for Tivoli Web Gateway on the application server:

SystemOut.log

The SystemOut.log file gathers standard out information from the WebUI_AppServer application server. Use this log file to determine if WebUI_AppServer was started without exceptions and to view trace messages when tracing is active. This file is located in `$WAS_HOME/logs/WebUI_AppServer`

where

`$WAS_HOME`

is the WebSphere installation directory

SystemErr.log

The SystemErr.log file gathers standard error information from the WebUI_AppServer application server. Use this log file to view exceptions that were written to the standard error stream. This file is located in `$WAS_HOME/logs/WebUI_AppServer`

where

`$WAS_HOME`

is the WebSphere installation directory

The following trace file is available for the Tivoli Web Gateway on the endpoint:

wctrace

The wctrace file contains trace information to be used for debugging purposes. The trace file is located in `$LCF_ROOT`

where

`$LCF_ROOT`

is the endpoint root directory

wclog

The wclog file contains trace information to be used for debugging purposes. The trace file is located in `$LCF_ROOT`

where

`$LCF_ROOT`

is the endpoint root directory

debug.log

The debug.log file contains trace information to be used for debugging purposes. The trace file is located in `$LCF_ROOT`

where

\$LCF_ROOT

is the endpoint root directory

Configuring Trace Settings

The `traceConfig.properties` file allows you to configure the trace settings for Tivoli Web Gateway. It contains the following parameters:

TraceLevel

Specifies the level of detail for trace files. Supported values are:

- 0 (none)
- 1 (fatal)
- 2 (fatal and error)
- 3 (fatal, error, and warning)

DISPLAY_ENTRY_EXIT

Specifies whether exit and entry events for Tivoli Web Gateway methods must be logged. Supported values are **true** and **false**. The default value is **true**.

MaxFileSize

Specifies the maximum size for each trace file. The default value is 512 KB.

MaxTraceFiles

Sets the maximum number of trace files to be created. When this number is reached, the oldest file is deleted.

Setting the keywords listed below to **true**, you can enable the tracing function for the corresponding components:

- `component.console`
- `component.dmsserver`
- `component.event`
- `component.notification`
- `component.plugins`
- `component.resultscollector`
- `component.twgapi`
- `component.database`
- `component.enrollserver`
- `component.datconverter`
- `component.mcollect`
- `component.notificationhandler`
- `component.api`
- `component.userservices`

This file is located in `$WAS_HOME/installedApps/hostname/DMS_WebApp.ear/dmsserver.war/WEB-INF/classes`

where

\$WAS_HOME

is the WebSphere installation directory

Troubleshooting

The tracing function is intended for debugging purposes. If enabled for extended periods of time, tracing can decrease performance and slow the processing of the product considerably.

The `webconsole.properties` file defines the trace settings for the `debug.log` file. The `webconsole.properties` file is located in the following path:

`$WAS_HOME/installedApps/node_name/WebConsole.ear/WebUI.war/WEB-INF/classes,`

where

`$WAS_HOME`

the WebSphere installation directory

`node_name`

is the name of the node where WebSphere is installed

To enable tracing to the `debug.log` file, set the `TRACE.ENABLED` key to **true**. The default value is **false**.

Configuring Trace Parameters

To enable further traces, you can use the **wwebcfg** command, described on the following pages:

wwebcfg

Sets trace parameters.

Syntax: **wwebcfg** {-s [*key*[=*value*]] | -d *key*}

Description: Sets parameters for trace level, trace directory, temporary working directory, and trace size.

Options:

-s Sets the value of a parameter specified by its *key* and with the supplied *value*, or, if no value is specified, displays the values of all parameters.

Note: If you enter a key incorrectly, no error is given but the incorrect key is added to the configuration file with the value that you specified.

The keys to configure the trace are as follows:

trace_level

The trace levels you can set are as follows:

- 0** No trace. This is the default value.
- 1** Fatal only
- 2** Level 1 plus errors
- 3** Level 2 plus warnings
- 4** Level 3 plus informational messages
- 5** Level 4 plus verbose
- 6** Maximum

This option enables the logging of trace information to the \$DBDIR/webui/webuin.trc files, where *n* indicates the file number.

product_dir

Use this key to change the trace directory. The default values are:

UNIX \$DBDIR/webui

Windows

%DBDIR%\webui

working_dir

Use this key to change the directory where temporary files are created. The default values are:

UNIX \$DBDIR/webui/work

Windows

%DBDIR%\webui\work

trace_size

The size in bytes of the trace file. The default value for trace_size is 1000000. You can increase or decrease it. When the file size is equal to the configured value, a new file is created with a new filename.

-d *key* Deletes the specified key.

Authorization: None.

Return Values: The **wwebcfg** command returns one of the following values:

0 The operation is successful.

-1 The operation is unsuccessful.

Examples:

1. To set the maximum trace level, issue the following command:
2. To display all parameters and their values for trace configuration, issue the following command:

```
wwebcfg -s trace_level=6
```

```
wwebcfg -s
```

An example of the information that is displayed is as follows:

```
product_dir=/Tivoli/db/helsinki.db/webui  
working_dir=/Tivoli/db/helsinki.db/webui/work  
trace_level=6  
trace_size=1000000
```

3. To delete the trace_size key, issue the following command:

```
wwebcfg -d trace_size
```

Troubleshooting

The problems and solutions listed here are categorized by the activities being performed.

Administrator on the Tivoli Server

Problem: Authorization error using **wweb** command

The **wweb** command returns the following error:

```
FRWSL0007E An authorization error of type
"o_errs:0022 insufficient authorization" occurred ...
```

Solution: You must add the WebUI_Admin role on the Tivoli server (see “Adding the WebUI_Admin Role” on page 281).

Problem: Profile not found error using **wweb** command

When publishing a reference model, the **wweb** command returns the following error:

```
Profile profile_name not found
```

Solution 1 : Ensure you have registered the plug-in for the profile (see “Registering Plug-ins” on page 282).

Solution 2: Ensure the syntax of the reference model is correct and that there is one (UNIX) or two (Windows) carets (^) separating the name of the reference model and the version. For example: @managers^1.0 (UNIX)
@managers^^1.0 (Windows)

Problem: A reference model profile gives a “not found” error on publishing, but is available with the correct name

In a scenario with interconnected Tivoli management region server, when publishing an existing reference model, the **wweb** command returns the following error:

```
Profile profile_name not found
```

You have checked that the profile exists.

Solution: With interconnected Tivoli management region servers, on the server from which you issue the publish command you must have previously installed the Change Manager component. In addition, the profile DataMovingRequests.1 profile must be present. If it does not exist, create it, using the **wspmvdata** command (see *Reference Manual for Software Distribution* for details of the command).

Problem: Results not returned

The result of an operation on a profile made on a Web Interface client is not returned to the Tivoli server within the set time interval (the default is five minutes, set by the Web Gateway). Neither is any result given in the Software Distribution log file or a related update made to the Inventory tables (SD_INST, SD_H_INST, COMPUTER). However, you should allow time for a completion message being held in a notification queue.

Solution: If, after a significant time interval, the notice has not arrived, check that the profile was not deleted on the Tivoli management region server between the start of the Web Interface operation and before the receipt of the results. If it was deleted, there will be no log file and the database updates on the Tivoli server will not be correct, even though the notice is still available and is waiting to be delivered. You can resolve this problem by creating an empty package of the same name as the deleted

one on the Tivoli management region server, to allow the notice to be recognized. Once this has happened and the log files or Inventory databases updated, the empty package can be deleted.

Problem: Information concerning software packages installed on a Web Interface client is not consistent when an endpoint is installed on the same workstation.

The software packages distributed using Software Distribution are tracked in the Inventory database by using the endpoint label, while the software packages distributed using the Web Interface are tracked by using the workstation hostname. This condition can lead to a misalignment of information in the Inventory database if you install an endpoint on the Web Interface client workstation and distribute software packages to this workstation using Software Distribution.

To avoid this problem, you should install the endpoint with the same name as the workstation hostname or, if the endpoint is already installed, change its name to match the workstation hostname.

Problem: Distributions to a large number of users fail

Distributions to a large number of users might fail because of a transaction timeout. If the SystemOut.log and TWGapi.log files contain a message stating that the transaction has timed out, perform the following steps:

1. Stop the DMS_AppServer.
2. Start the WebSphere Administration Console.
3. Select **Servers » Application Servers**.
4. Select **DMS_AppServer » Transaction Service**.
5. Set the **Total transaction lifetime timeout** property to 0. This sets the timeout to infinite.
6. Apply and save the new configuration.
7. Restart the DMS_AppServer.
8. Stop the IBM HTTP server.
9. Modify the Timeout configuration parameter in the httpd.conf file as necessary depending on your environment.
10. Restart the IBM HTTP server.
11. Perform the distribution again.

For more information on the SystemOut.log file and TWGapi.log, see “Tivoli Web Gateway Logs and Traces” on page 405, for more information on the httpd.conf file, see “Tips to Improve Web Server Performance on AIX and Solaris Systems” on page 409.

Web Interface User

Problem: User reports no Web objects visible

A user cannot see any profiles in the Web Interface.

Solution:

- Check that the ID that the user is using to access the Web Interface has been associated with the Web objects in a **wweb** command.
- Check that profiles have been published on the server that the user is accessing.
- Check that the published profiles were published as being valid for the interp of the user's platform.

Problem: Unsuccessful log on for authorized user

A user logs on to the Web Interface using the correct user name and password, yet receives the following error message:

Your login was unsuccessful

Solution: The internet security options in the Internet Explorer browser are too strict. Instruct the user to lower the security level to medium.

Problem: The user cannot apply a reference model

Solution 1: You have published a reference model, but the software packages or Inventory scans that are required to satisfy the model have not been published. Ask the user to identify the pending operations generated from the reference model and listed in the Operations Console. Check that they have all been published on the appropriate server, for the correct interps and for the user in question.

Solution 2: You have published a reference model, that contains elements other than Software Distribution and Inventory elements. Ask the user to identify the pending operations generated from the reference model and listed in the Operations Console. Check that they are all either Software Distribution or Inventory operations.

Problem: Users cannot see a published software package or reference model profile

A software package is published for some users and some interps, but no-one can see the profile.

Solution: Ensure that the distribution is complete. Use the **wmdist** command to check the MDist 2 queue (refer to the *Tivoli Management Framework: Reference Manual*). If the operation is complete, check the Software Distribution log file for software distribution operations and the data moving log file for change management operations. For more information on these log files, refer to *IBM Tivoli Configuration Manager: Reference Manual for Software Distribution*. If the log file does not provide any information and the MDist 2 queue is empty, set traces for Software Distribution on the Tivoli server and on the endpoint.

Chapter 14. Using the Web Interface

The Web Interface is a Web-based tool that allows you to maintain the correct levels of your system and application software on a workstation that is not permanently connected to your enterprise network. It also allows you to satisfy enterprise requests for information about such a workstation.

Using a suitable Web browser you access a server from which you can perform operations that will do one or more of the following:

- Install, upgrade, or uninstall individual software objects
- Satisfy enterprise requests for information about the software and hardware used with your computer
- Bring your computer into line with a pre-determined enterprise standard in terms of software installed

To perform these activities you select *Web objects* and run them on your workstation. Web objects are of three types:

Software packages

These packages contain a software object, and the associated instructions to tell your computer how to install the software and configure it. Packages can be for new software or for upgrading existing software.

Inventory scans

These scan your computer and send a list of the hardware, or software, or both, to a central enterprise database.

Reference models

These are structured lists that contain the desired state of the software on your computer (which software objects should be installed and at which level). These interact with your computer to determine what is the current state of the software, and then make available a set of software packages that will bring your computer's software to the desired level. Software that is not mentioned in the model is not touched. Reference models often end with an inventory scan to report the final state back to the central database.

This chapter tells you how to start the Web Interface, how to use it to run the operations described above, and how to resolve any problems that might occur.

Starting the Web Interface

Before starting, you require the following information from your system administrator:

- The Web address of the Web Interface server
- A username and password
- An explanation of what specific options are available to you
- A point of reference in case of problems

Depending on the security settings implemented in your environment, you might have to accept several certificates before the Web Console is displayed. After verifying all certificates the first time you log in, you can click the **Grant all** button to accept all certificates in subsequent sessions.

Starting the Web Interface

To start the Web Interface perform the following steps:

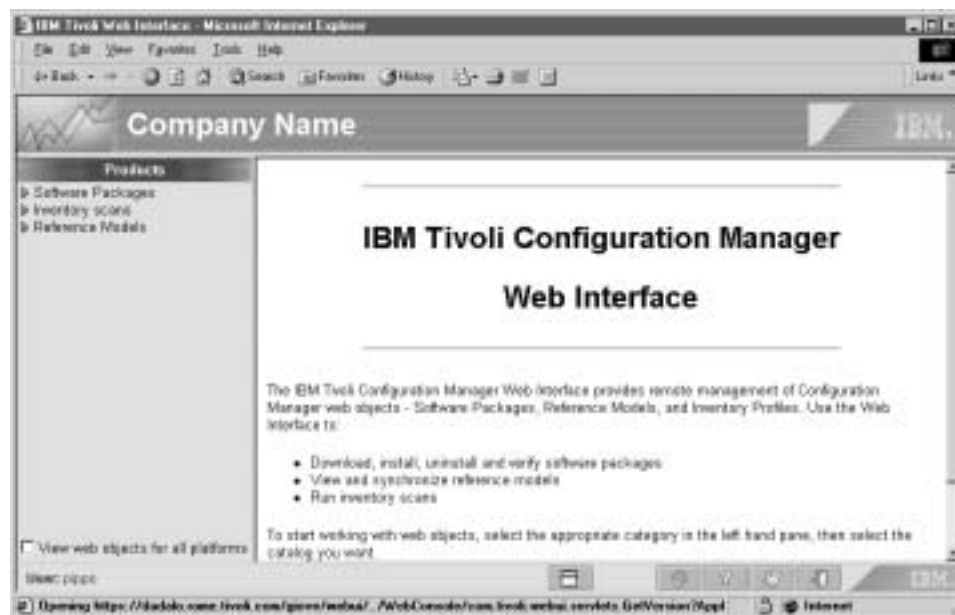
1. Open a browser window and ensure that Java and JavaScript options are enabled. The browser should support Java applets (your system administrator will tell you which browsers you can use).
2. Type the Web address for the Web Interface server provided by your administrator.

A logon window is displayed:



3. Bookmark the page for future reference.
4. Enter your username and password, then click the **Login** button. If you do not have a username and password, obtain them from your administrator.

The Web Interface starts and a window similar to the one below is displayed.



Note: If an error icon is displayed in the browser page while initializing the WebUI on the Web user machine, check the following points:

- Java plug-in is not installed.
- You did not press the Grant button on the certificate validation panel of the Operations Console.
- The Operations Console encountered an error while initializing the client.

In this case, you cannot use the Operations Console. Contact your system administrator for further information.

Web Interface Window






The left-hand pane of the Web Interface window is the Products pane and contains a list of all the categories of Web object to which you have access.

The right-hand pane is the object pane. When you select a Web object category, this pane contains a list of all the Web objects to which you have access.

Select the **View web objects for all platforms** check box to display Web object categories for all platforms, not just your current platform. You can also modify the default value for this check box, so that the value you specify is already selected when the Web Interface opens.

At the bottom of the browser window is the task bar. To the left of the task bar, there is a status display showing who is logged in to the Web Interface. To the right of the task bar is a group of buttons. Table 51 describes these buttons:

Table 51. Web Interface Task Bar Buttons

Select this button...	To do this...
	Display the Operations Console
	Display the Web Interface home page
	Display the Help page
	Display the password authorization page
	Display the logoff page

Operations Console

When you first start the Web Interface, a second window appears automatically. This is the Web Interface Operations Console, a Java applet. The Operations Console initializes the Web Interface environment by downloading two sets of files from the Tivoli server. The first set of files checks the local system environment. The second set temporarily creates a Tivoli endpoint environment.

Each time you select an operation to perform from the Web Interface, that operation appears as a pending operation in the upper pane of the Operations Console:

Starting the Web Interface



You can change the order of the operations using the arrow keys. Each operation is defined in terms of:

- Application
- Type of operation
- Web Package name
- Web Package version

You can delete one or all operations, or just operations that have been completed. When you click **Start**, each operation is performed in the order in which it appeared in the list.

Pending operations showing in the Operations Console stay there until you delete them or complete them. If you close the connection with the server the Operations Console will also close. When the connection is remade, the console is re-opened, and whatever was listed there is still there for you to work with. However, if you complete an operation and then close the connection, that pending operation (no longer pending) is removed from the console so that it does not show when the console is next re-opened.

Messages relating to the operations you are running and a progress bar are shown in the lower pane of the Operations Console.

Using Web Objects

This section describes how to:

- Install, verify, and uninstall software packages
- Run inventory scans
- View and apply (synchronize) reference models

Software Packages

From the Web Interface you can install, verify, and uninstall software packages. You connect to the server, obtain a current list of available software packages, and retrieve them at your convenience.

Installing Software Packages

To install a software package, perform the following steps:

1. Log on to the Web Interface as described in “Starting the Web Interface” on page 309.
2. In the Product pane, click the icon next to **Software Packages**. A list is displayed showing a sub-catalog of software packages.
3. Click a sub-catalog. A list of software packages is displayed in the object pane. See the following example:



Table 52 describes the columns in the Software Packages window:

Table 52. Columns in the Software Packages Window

This column...	Contains...
Name	The name of the software package.
Version	The version number of the software package.
Description	A brief description of the software package.
Size	The size of the software package.
Conn Speed	The speed at which you must be connected to the server to install the package.

4. Select the check box next to each software package you want to install.
5. Click **Install**. The operation appears in the Operations Console **Pending operations** table.
6. In the Operations Console, click **Start**. This operation, and all other pending operations start running.

When a software package operation is started, the package is first downloaded to your computer. This may take some time, depending on the size of the package and the connection speed. On operating systems and browsers that support checkpoint-restart downloads, if the connection is severed during the download, on reconnection the system attempts to continue the download automatically from the last checkpoint. Otherwise the operation recommences automatically, from the beginning. Once the download has completed, the install operation goes ahead, even if the connection is subsequently severed.

After you have started processing the pending operations, you can stop the processing at any point by closing down the browser or severing the connection. When next you connect, the Operations Console will show the outstanding operations and you can click **Start** again to continue.

When the installation operation is complete, the results of the operation are sent back to the Tivoli server. If the connection is not available at the moment of completion, the completion message is queued, and is sent as soon as the connection is re-established. The results are recorded in a log file and the Inventory component database is updated accordingly, see “Monitoring the Results of Operations” on page 298.

Note: Once you have selected a software package and installed it, the software package is not removed from the list of available Web objects in the Web Interface window, and remains available for you to select again. However, if you believe that a software package has not been installed correctly, you should contact your system administrator rather than try and re-run the install operation. This is because the operation might not have been designed to allow a re-install of the same software object at the same version level.

Verifying Software Packages

The verification of a software package checks whether the operation executed on the target object is consistent with the installed package, that is, that the files have been successfully installed on the target system.

To verify a software package, perform the following steps:

1. In the Products pane, select **Software Packages**, then select the sub-category you want.
2. In the Software Packages pane, select the check box next to the software package you want to verify.
3. Click **Verify**. The operation appears in the Operations Console **Pending operations** table.
4. In the Operations Console, click **Start**. This operation, and all other pending operations start running.

Uninstalling Software Packages

The uninstallation of a software package might re-install the prior version, depending on the software object and how the package has been set up. For this information see your administrator.

To uninstall a software package, perform the following steps:

1. In the Products pane, select **Software Packages**, then select the sub-category you want.
2. In the Software Packages pane, select the check box next to the software package you want to uninstall.
3. Click **Uninstall**. The operation appears in the Operations Console **Pending operations** table.
4. In the Operations Console, click **Start**. This operation, and all other pending operations start running.

The same comments apply to the downloading and running of the uninstall as apply to the install operation, except that both of the phases are much quicker. There is therefore much less likelihood of severing the connection while your operations are being performed.

Running Inventory Scans

To run an Inventory scan perform the following steps:

1. In the Products pane, select **Inventory scans**, then select the sub-category you want. A list of Inventory Scans is displayed in the object pane. See the following example:



2. In the Inventory Scans pane, select the check box next to the profile you want to use for scanning.
3. Click **Scan**. The operation appears in the Operations Console **Pending operations** table.
4. In the Operations Console, click **Start**. This operation, and all other pending operations start running.

When the scan operation is complete, the results of the scan are sent back to the Tivoli server, where they update the enterprise Inventory database. If the connection is not available at the moment of completion, the completion message is queued, and is sent as soon as the connection is re-established. The results are recorded in a log file and the Inventory component database is updated accordingly. For more information, see "Monitoring the Results of Operations" on page 298.

The same comments apply to the running of Inventory scans as apply to installing software packages, except that both of the phases are much quicker. There is therefore much less likelihood of severing the connection while your operations are being performed.

Reference Models

Reference models are models of how a corporation would like different types of workstations to be configured. They show, for each significant enterprise software object, the desired version and state of the software. A corporation can design many different models. You might need to ask your system administrator which model applies to you.

Viewing Reference Models

You would view a reference model prior to running it in order to understand what the model is likely to change on your system.

Using Web Objects

To view the contents of a reference model, perform the following steps:

1. In the Products pane, select **Reference Models**, then select the sub-category you want. A list of Reference Models is displayed in the object pane. See the following example:



The reference models available for selection are listed.

Table 53 describes the columns in the Reference Models window:

Table 53. Columns in the Reference Models Window

This column ...	Contains ...
Name	The name of the reference model.
Version	The version number of the reference model.
Description	A brief description of the reference model.
Size	The size of the reference model.
Conn Speed	The speed at which you must be connected to the server to apply (synchronize) the reference model.

2. In the Reference Models pane, select the check box next to the reference model you want to view.
3. Click **View**. The elements of the reference model are displayed:



Applying (Synchronizing) Reference Models

Applying (sometimes called synchronizing) a reference model means allowing the model to determine the level of the software on your computer and comparing that with the desired state information in the model so that it can generate a list of the appropriate software packages to install.

To apply a reference model perform the following steps:

1. In the Products pane, select **Reference Models**, then select the sub-category you want. A list of Reference Models is displayed in the object pane. The reference models available for selection are listed.
Table 53 on page 318 describes the columns in the Reference Models window:
2. In the Reference Models pane, select the check box next to the reference model you want to synchronize.
3. Click **Synchronize** to apply the reference model.

Web Interface immediately commences a scan of your system to determine its current state. It then reports that state to the Web Interface server. The server then determines the required packages and sends the information back as a list of operations to be carried out. No progress bar is displayed during this operation.

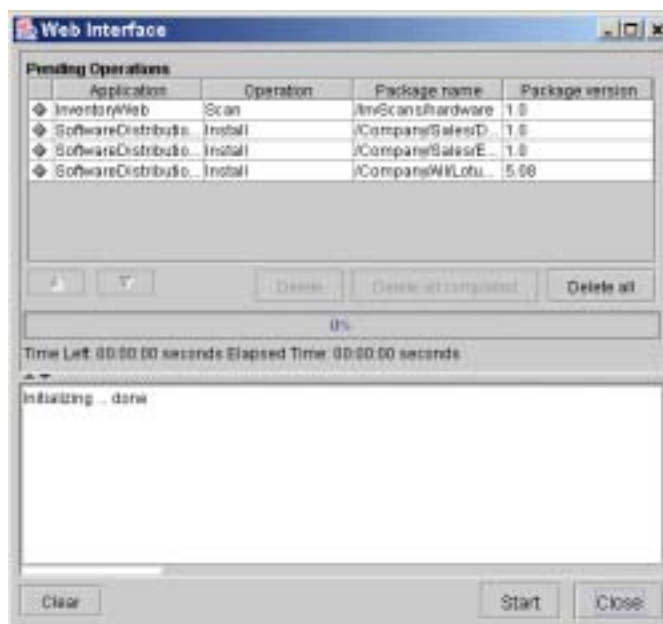
If all the elements of the reference model have been published, the pane changes to show the actions to be performed on the elements of the reference model. See the following example:



If one or more elements of the reference model have not been published, you cannot apply the reference model. Click **Back** and contact your Administrator.

Note: If the software package has not been installed on the target before, the default action becomes **Install**.

4. Click **OK**. The corresponding operations appear in the Operations Console. See the following example:



Note: If the server connection is severed at any time between first clicking on **Synchronize** (in step 3 on page 319) and seeing the pending operations appear in the console, the entire operation must be restarted.

5. In the Operations Console, click **Start**. All pending operations start running.

The pending operations are either software package operations or inventory scans. The comments made above about what happens if the connection is severed during these operations apply separately to each operation. After you have started processing the pending operations, you can stop the processing at any point by closing down the browser or severing the connection. When next you connect, the Operations Console will show the outstanding operations and you can click **Start** again to continue.

Troubleshooting

This section helps you to resolve problems that might occur while using the Web Interface.

Problem: Unsuccessful login for authorized user

You log on to the Web Interface using the correct user name and password, yet receive the following error message:

Your login was unsuccessful

Solution: The internet security options in the Internet Explorer browser are possibly too strict. Lower the security level to medium.

Problem: No Web objects visible

Solution: Contact the system administrator: the Web objects might not be correctly published.

Problem: You cannot see a specific published object

Solution: If you are sure, from other information, that a package has been published, but you cannot see it, contact your system administrator. It will be helpful to tell the administrator if other users can see it.

Problem: You cannot apply a reference model - objects not published

You have tried to apply a reference model but have received an error message saying that not all the objects referenced by the reference model have been published.

Solution: Copy the message and contact the systems administrator. To copy a message you have one of the following options:

- Your system software may let you copy the text and paste it into a mail
- You may be able to save the operations console as a file and attach it to a mail
- You may be able to take a snapshot of the window or screen and paste that into a mail
- You could copy the message word for word into a mail

Problem: You cannot apply a Reference Model - incorrect elements

You have tried to apply a reference model but have received the following error message:

Some elements in the Reference Model don't allow to perform the synchronization. Contact your Administrator.

This means that one or more of the objects in the reference model is not a Software Distribution or Inventory object.

Solution: Copy the message and contact the systems administrator. To copy a message you have one of the following options:

- Your system software may let you copy the text and paste it into a mail
- You may be able to save the operations console as a file and attach it to a mail
- You may be able to take a snapshot of the window or screen and paste that into a mail
- You could copy the message word for word into a mail

Problem: First software package failure

You have synchronized a reference model containing multiple software packages. If the first software package fails, all subsequent software packages also fail.

Solution: Perform the following steps:

1. Open the `webui.properties` file in a text editor. The `webui.properties` file is located in the `%USERPROFILE%` directory.
2. Set the `STOP_ON_ERROR` key to `FALSE`. By default, this key is set to `TRUE`.
3. To make the change effective, stop and restart the Web browser.

Part 5. Managing Resources

Chapter 15. Overview	327
Resource Manager and Pervasive Devices	327
What Is New in This Version	328
Resource Manager and Users	328
 Chapter 16. Installing and Configuring Devices for Resource Manager	331
Installing the Device Agent for Palm OS Devices	331
Using HotSync Manager.	331
Connecting to the Server with a Cradle.	331
Auto-connection to the Web Gateway Server	332
Conduit Communication with the Device Agent	332
Planning for Installation and Configuration	332
Installing the Files.	332
Configuring the Device Agent using the GUI	333
Preparing a Configuration File	334
Setting Up the PDBGenerator Utility	336
Administrative Tasks for Palm OS Devices	337
Changing a Configuration Parameter	337
Changing the Value for the Network Attachment Option	338
Manually Setting Up a Modem Connection	339
Creating a Network Interface (Manual Modem Setup)	339
Configuring the Modem (Manual Modem Setup)	340
Configuring the Device Agent (Manual Modem Setup)	340
SSL and the Device Agent	341
Enabling SSL	341
Disabling SSL	341
Disabling the Auto-connection When Using a Cradle.	341
Using a Security Manager	342
Installing the Device Agent for Windows CE Devices	342
Communicating with the Host PC	342
Planning for the Installation	342
Installing the Device Agent.	343
Configuring the Device Agent.	344
Configuration File.	346
Distributing and Processing the Configuration File.	346
Configuration File Format	346
Sample Configuration File for Windows CE Devices	347
Setting the Device ID.	347
Setting the Authorization Password	347
Scheduled Polling	347
Keeping Device Agent Files on Windows CE.	348
Administrative Tasks for Windows CE Devices	349
Using SSL with the Device Agent.	349
Managing the List of Locally Valid Certificate Authority Certificates.	349
Using a Security Manager	349
Administrative Tasks for Nokia Devices	350
Enabling security for Nokia s60 devices	350
Configuring the DMS server	350
Configuring the IBM HTTP server	350
Downloading the certificate on the device	350
Enabling the WebSphere Application Server security	350
Creating a connection profile on the Nokia s60 device	350
 Chapter 17. Managing Resources	353
Sample Tivoli Environment.	354
Registering the Resource Types	354
Working with Pervasive Devices	355
Registering Endpoints as Resource Gateways	355
Enrolling Pervasive Devices Automatically	355
Discovering Devices	356
Creating Devices	357
Modifying Devices	360
Deleting Devices	361
Nokia Provisioning Scenario	362
Working with Users	368
Viewing Users	368
Grouping Resources	369
Using the Tivoli Policy Facility	369
Writing Scripts	370
Using the Tivoli Query Facility	370
Creating Resource Groups	370
Adding Resources to a Static Resource Group	371
Adding Dynamic Resource Groups	372
Listing and Finding Resources.	373
Customizing Resource Manager Actions	375
Script for wresource action Command	375
 Chapter 18. Device Customization Parameters	377
Creating a Customization File for Palm OS Devices	377
Format Parameters	377
Preset Countries or Regions and Stored Values.	378
Time Formats and Stored Values	379
Date Formats and Stored Values	379
Long Date Formats and Stored Values	379
Number Formats and Stored Values	379
General Parameters	380
Network Parameter	380
TCP/IP Parameters	381
Modem Parameter.	381
Agent Parameters	381
Proxy Parameters	382
Sample File for Windows CE Devices	382
PPP Parameters	383
TCP/IP Parameters	383
Browser Parameters	384
Mailer Parameters.	385
Management Parameters	385

Chapter 19. Using the Command Line	387	Tips for Performance Improvement	409
wresgrp	388	Tips to Improve Web Server Performance on	
wresgw	391	AIX and Solaris Systems.	409
wresource	394	Improving Application Server Performance	410
Chapter 20. Sample Scripts	401	Troubleshooting Connection Problems between	
Script for Use with wresource action Command	401	Device Manager and Devices	411
Script for Default Policy	402	Deleting a Tivoli Endpoint That Is a Resource	
Script to Create Resource Group and Activate		Gateway	411
Default Policy	403	Deleting and Reconnecting an Endpoint with the	
Chapter 21. Troubleshooting	405	Same Label	412
Tivoli Resource Manager Traces	405	Deleting and Reconnecting an Endpoint with a	
Tivoli Web Gateway Logs and Traces	405	New Label	412
Tivoli Web Gateway Configuration Files	407	Changing the User ID of Palm Devices	413
TheTransaction.properties file	407	Resource Groups Do Not Exist	413
Managing the Transaction.properties File	408	Grayed out Query Selection Box on Resource	
The traceConfig.properties File	408	Group Members Dialog	413
		Activity Planner login failure on Linux	413

With Resource Manager you can manage resources, namely users and pervasive devices such as Palm devices, WinCE, and Nokia 9300, 9500, and s60 devices, in your Tivoli environment. You can keep track of pervasive devices in your environment, distribute software, scan devices, and customize devices from a central location.

Locating Related Information

Information related to this service is available from the following sources:

- *Introducing IBM Tivoli Configuration Manager*. This provides an overview of the service.
- *Planning and Installation Guide*. This includes information about how to install and configure Resource Manager to work with resources, pervasive devices and users. For information about configuring the devices themselves, see Chapter 16, “Installing and Configuring Devices for Resource Manager,” on page 331.
- *Messages and Codes*. This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 15. Overview

Resource Manager, a service that runs on the Tivoli server, extends the Tivoli Management Framework to manage various types of resources. Resource Manager adds a fourth tier of *resources* to the three-tier Tivoli architecture of Tivoli server, gateway, and endpoint. Resources can be pervasive devices or users.

Resource Manager and Pervasive Devices

Resource Manager enables you to perform basic operations on *pervasive devices*, such as creating or deleting devices, distributing software, and scanning and customizing devices. Figure 4 shows a simple configuration:

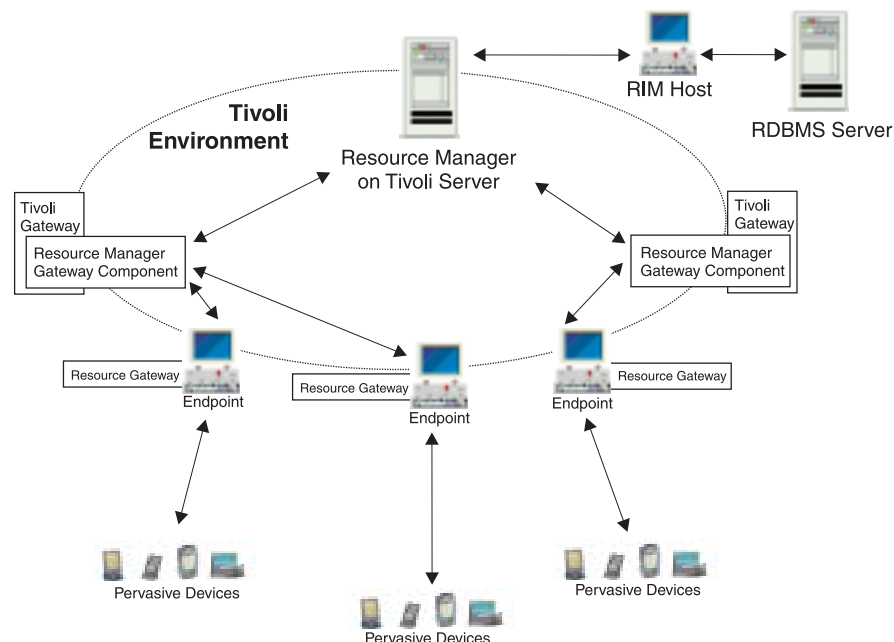


Figure 4. Pervasive Devices Integrated in a Tivoli Environment

Resource Manager, which is installed on the Tivoli server, maintains a master database of the pervasive devices that are managed by the resource gateways. Resource gateways are endpoints that have applications that extend the Tivoli endpoint to manage pervasive devices. In Tivoli Configuration Manager Version 4.3.1, the only supported resource gateway is Web Gateway. Gateways that have the Resource Manager gateway component installed connect the Tivoli server with the resource gateways.

Each resource gateway maintains an independent Web Gateway database. Resource Manager notifies the Web Gateway databases of any changes to the master database and vice versa. For example, when you create or delete a device in the Resource Manager database, Resource Manager notifies the Web Gateway database on the endpoint managing the device to update its database. When a new device connects, it is automatically enrolled and Web Gateway notifies Resource Manager to update its database.

Resource Manager and Pervasive Devices

Depending on the number of pervasive devices, a resource gateway configuration can consist of a single application server with a database management system (DBMS), possibly running on the same machine, or a cluster of Web servers sharing the same DBMS. Web Gateway servers can function independently or as a cluster.

- To function independently, each Web Gateway server must have its own Web Gateway database. In this configuration, no information stored in the database is shared with any other installation of the Web Gateway component.
- To function as a cluster, a single Web Gateway database is shared by multiple installations of Web Gateway servers. The first installation of the Web Gateway server is the primary server in the cluster. Additional installations of Web Gateway servers are secondary servers in the cluster. All secondary servers share the Web Gateway database of the primary server. In this configuration, communication among the Tivoli applications is performed by the primary server while the secondary servers can process incoming requests.

For details about these configurations, refer to *Planning and Installation Guide*.

What Is New in This Version

Tivoli Resource Manager features the following enhancements:

Support of Nokia 9300 and s60 devices

You can define and manage Nokia 9300 and s60 devices.

Auto enrollment Nokia devices

Automatic enrollment is now enabled for all Nokia devices.

One-click persistent installation of IBM agent on WinCE

After a hard reset or when the battery comes down, you can still run a one-click installation of the device agent.

Bootstrap management

Information about bootstrap are now stored in the Inventory database.

Resource Manager and Users

Resource Manager enables you to work with the resource *users* that are defined in an Enterprise Directory server, for example, the Lightweight Directory Access Protocol (LDAP) server.

Users are associated with endpoints in a one-to-one relationship and the mapping is stored in the LDAP server. Resource Manager enables you to view the association between a user and an endpoint.

The following graphic shows how users fit into the Tivoli environment.

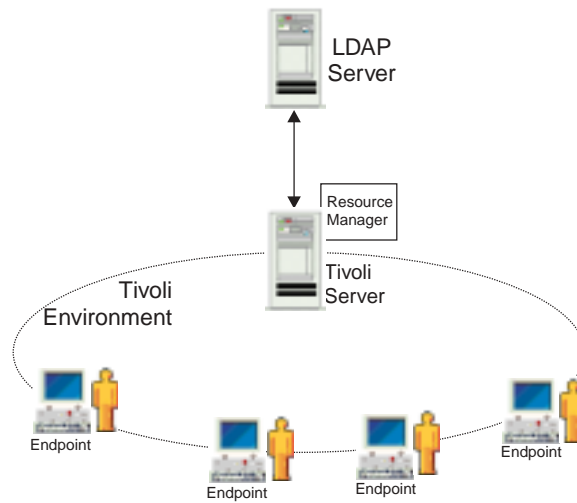


Figure 5. Users Integrated into a Tivoli Environment by Resource Manager

Resource Manager enables you to distribute software packages, using Software Distribution, to the endpoints that are associated with users by distributing the profile to a resource group of users. Similarly, you can distribute an inventory profile to a resource group of users to scan the endpoints associated with the users.

Chapter 16. Installing and Configuring Devices for Resource Manager

This chapter provides instructions for installing and configuring devices to work with Web Gateway and Resource Manager.

Installing the Device Agent for Palm OS Devices

To manage a Palm OS device with Web Gateway, the plug-in and device agent for Palm OS are needed. The plug-in and device agent communicate with each other using HTTP and perform system management tasks. The plug-in is written as a servlet and resides on the Web Gateway server. The device agent for Palm OS resides on the device and functions as the client.

Processing jobs is the main task of the device agent. If a plug-in for Palm OS has a job for a device, the plug-in puts the job in the polling queue. At the next polling cycle, the device agent reads the polling queue and starts processing all waiting jobs.

The device agent supports communication through a Palm cradle, a modem, mobile modem, or any other TCP/IP type of network connection, for example, Infrared, Bluetooth or Wi-Fi connections. Use m-Router™ to access your computer network facilities from your Palm device, for example to browse the Internet. You can connect your Palm device to m-Router using different connection types, such as USB, Infrared, and Bluetooth.

For information on installing the device agent, see “Planning for Installation and Configuration” on page 332.

Using HotSync Manager

To handle the transfer of files between the Palm OS device and the host PC or any other network computer, the device requires HotSync Manager to be installed and running. HotSync Manager synchronizes files using a conduit. Each conduit is application-specific, and each conduit is registered with HotSync Manager.

The version of HotSync Manager must be at least the same version as the Palm OS version on the device.

HotSync Manager is started by a device owner pressing the **HotSync** button on the Palm OS device. The following occurs:

- When the device notifies HotSync Manager on a host PC, HotSync Manager starts all registered conduits.
- After the last conduit returns, HotSync Manager completes the remaining tasks and ends.
- Finally, HotSync Manager notifies the Palm OS device and the device notifies all related applications that their data might be updated.

Connecting to the Server with a Cradle

A cradle provides a direct connection between the Palm OS device and a host PC using a serial cable. With a cradle connection, a device owner can copy files and data between a Palm OS device and a host PC.

Installing Device Agent for Palm OS Devices

After the device agent is installed, use m-Router to connect to the Web Gateway server.

You can also connect to the Web Gateway server using a direct network connection. The direct network connection, which is often a modem or mobile modem, must use a full duplex TCP/IP connection.

Auto-connection to the Web Gateway Server

Each time the data in the Palm OS device is synchronized with the data in the host PC, the device agent connects to the Web Gateway server through the device agent conduit. All jobs that are waiting for the device in the Web Gateway job queue are processed.

Conduit Communication with the Device Agent

The device agent conduit can only read from or write to databases on the device and cannot communicate with the device agent directly. When requests are made to a server, the device agent writes the device status and the information to the database, then HotSync Manager reads it. Similarly, the transaction result is also stored in a database by the device agent conduit, and then the device agent reads it.

All configuration information and data, such as a software distribution job, is saved at the host PC temporarily. When the transaction is complete, the device agent conduit copies the information and data to the Palm OS device. With this model, a server "sees" the conduit as the device.

Planning for Installation and Configuration

In planning for the installation and configuration for a Palm OS device, consider the following:

- Determine the connection type

The Palm OS device can use a cradle connection or a direct network connection, or both.

When you are using a cradle connection with the device agent conduit installed, each time data is synchronized between the Palm OS device and the host PC, the device agent connects to the Web Gateway server through the device agent conduit. All jobs that are waiting for the device in the job queue are processed.

A direct network connection is typically a modem or mobile modem.

- m-Router must be installed and working, so your Palm OS device can exchange files with a host PC or another network computer.
- After installing the device agent, as described in "Installing the Files," you can configure the device agent using the GUI, as described in "Configuring the Device Agent using the GUI" on page 333, or you can create a configuration file to be distributed to the device, as described in "Preparing a Configuration File" on page 334.

Although other installation and configuration scenarios are possible, the typical scenario is either of the following:

- A cradle connection that uses a configuration file prepared by a network administrator.
- A modem connection that was configured by the network administrator, and uses a configuration file prepared by a network administrator.

Installing the Files

To install the files, do the following steps:

1. Ensure that m-Router is installed and working, so your Palm OS device can exchange files with a host PC or another network computer.
2. Copy the device agent (**pvcpalm.prc**) and the Palm OS resource file (**dmsagentresources.pdb**) to the device using m-Router.

The device agent and associated files are located at:

For IBM AIX

/install_directory/agents/palm

Where the default *install_directory* is *usr/TivTwg*

For Sun Solaris Operating Environment, Linux and HP-UX

/install_directory/agents/palm

Where the default *install_directory* is */opt/TivTwg*

For Microsoft Windows

install_directory\agents\palm

Where the default *install_directory* is *c:\TivTwg*

3. Copy the configuration file, if you created one as described in “Preparing a Configuration File” on page 334.

The device agent is now installed.

Configuring the Device Agent using the GUI

To configure the device agent, do the following steps:

1. Click the device agent icon on the PalmOS application list.
If the device agent has not been configured, the following warning message is displayed:
DYM6312W: No configuration data.
2. At **Manual Setup**, click **Hide**.
3. Click the agent icon and select **Options » Login Setting** to display the Logon dialog box.
4. In the **Username** field, type the user name.
If you are using IBM Tivoli Access Manager WebSEAL, use the same user name that authenticates with Policy Director.
5. In the **Password** field, type a password.
If you are using IBM Tivoli Access Manager WebSEAL, use the same password that authenticates with Policy Director.
6. Click **OK** to return to the main window.
7. Click the agent icon and select **Options » Server Setting** to display the Server Information dialog box.
8. In the **Address** field, type the host name or IP address of the Web Gateway server. If you are using IBM Tivoli Access Manager WebSEAL, type the host name of the WebSEAL server.
9. In the **Port** field, type the port number. The default value is 80. If you are using WebSEAL, type the number of the port used by the WebSEAL server.
10. The **Servlet Name** field does not need to change unless you are using IBM Tivoli Access Manager WebSEAL. If you are using IBM Tivoli Access Manager WebSEAL, type the following for the **Servlet Name** field:

/junction_name/dmservlet/PalmServlet

where:

Installing Device Agent for Palm OS Devices

junction_name

is the name of the junction that has been configured in WebSEAL to redirect the communication to WebSphere.

11. Click **OK** to close the Server Information dialog box. The configuration is now complete.

To start the connection, click **Connect**.

Preparing a Configuration File

As an alternative to using the GUI to configure the device agent, you can create and distribute a configuration file.

A configuration file assigns values to configuration parameters for the device agent. The configuration parameters included vary depending on the connection type.

The network administrator should prepare a configuration file with the appropriate configuration parameters for the device owner. The values are specified in a plain text file. A command-line utility converts the text file to a Palm database file. For more information on the command-line utility, see “Setting Up the PDBGenerator Utility” on page 336.

Typically for all connections, you include the following parameters:

- Service name for the device agent
- Web Gateway server address and port number
- Servlet name
- Palm user ID for the Palm OS device
- SSL setting
- In some situations, a password for authentication

In addition, for a modem connection you use all of the above configuration parameters plus the following parameters:

- PPP user name
- DNS addresses
- Modem phone number
- In some situations, a password for PPP access

To prepare a configuration file, do the following steps:

1. Open a file called Config.ini.sample in a text editor.

To get started quickly, use a copy of the Config.ini.sample file which is available on the Web Gateway server at:

For IBM AIX

/install_directory/agents/palm

Where the default *install_directory* is *usr/TivTwg*

For Sun Solaris Operating Environment, Linux and HP-UX

/install_directory/agents/palm

Where the default *install_directory* is */opt/TivTwg*

For Microsoft Windows

install_directory\agents\palm

Where the default *install_directory* is *c:\TivTwg*

- For all connections, edit your copy of the Config.ini.sample file so it contains the following parameters:

```
ServiceName=DevAgent
DMSAddress=DMServer.somedomain.com
DMSPort=80
PalmServletName=/junction_name/dmserver/PalmServlet
PalmUserID=PalmUser@rtp
SSLon=0
AttachmentOption=0
```

You must edit the values above for the **DMSAddress** and **PalmUserID** parameters. Review the values for the remaining parameters.

Use the following configuration parameter descriptions to determine the parameter values and whether additional parameters are required for your network.

Table 54. Palm OS configuration parameters

Configuration Parameter	Description	Sample Values/Notes
ServiceName	Network interface name to use for device agent	String value, such as: DevAgent
DMSAddress	Web Gateway server address or host name. If you are using IBM Tivoli Access Manager WebSEAL, type the host name of the WebSEAL server.	String value
DMSPort	Web Gateway server port number (integer). If you are using WebSEAL, enter the number of the port used by the WebSEAL server	80
PalmServletName	Palm servlet name	String value, such as: /dmserver/PalmServlet. If you are using IBM Tivoli Access Manager WebSEAL, type the following for the Servlet Name field: /junction_name/ /dmserver/PalmServlet
PalmUserID	The user ID for Web Gateway	String value
SSLon	SSL security (encryption level)	0=Off or 1=On
AttachmentOption	Determines the network connection type	Range 0 through 3. A value of 0 specifies that the Palm OS device should automatically detect the type of connection (modem or cradle) being used.

- If you are using a modem connection, add the following parameters to your Config.ini.sample file from the previous step:

```
UserName=PPUser
ModemPhone=123456789
DNSQuery=0
PrimaryDNS=9.68.4.11
SecondaryDNS=9.68.1.11
```

Installing Device Agent for Palm OS Devices

Modify the values for the following parameters in the Config.ini.sample file, except **DNSQuery**.

Configuration Parameter	Description	Sample Values/Notes
UserName	User name for a PPP connection	String value
ModemPhone	Modem telephone number	String value
DNSQuery	PPP queries for the DNS address (integer)	1=On 0=Off
PrimaryDNS	Primary DNS IP address	String value
SecondaryDNS	Secondary DNS IP address	String value

4. Convert your Config.ini.sample file to a Palm database file, called config.pdb, with the PDBGenerator command-line utility. For information on setting up this utility, see “Setting Up the PDBGenerator Utility.”

The conversion file utility is part of the pdbgene.jar file, which is installed with Web Gateway. Type the command as a single line (for readability, the command below appears on three lines):

```
java -cp $WAS_HOME/installedApps/hostname_DMS_Webapp.ear/dmserver.war/WEB-INF \
/lib/pdbgene.jar com.tivoli.dms.tool.pdbgene.PDBGenerator
Config.ini.sample config.pdb
```

where:

\$WAS_HOME

Is the WebSphere installation directory

You might have to set up the Java Virtual Machine (JVM) or identify the classpath before using the PDBGenerator utility.

After the initial device setup, you can change a configuration parameter value with a device configuration job, or by creating an updated config.pdb file and distributing that file to the Palm OS device.

When the device agent runs the first time, it locates **DevAgent** as the Palm OS **Service** parameter (on the **Network** menu) and sets the configuration parameters from the config.pdb database file.

Setting Up the PDBGenerator Utility

To set up the PDBGenerator utility, do the following steps:

1. Create a Config.ini.sample file with a plain text editor. For example, use Notepad on a host PC in a Windows environment.
2. Copy the Config.ini.sample file to the directory with the pdbgene.jar file (/install_directory/agents/tools).
3. Open a command prompt and navigate to the /agents/tools directory.
4. To make sure a JVM is properly installed and available to the /agents/tools directory (in your classpath), type java -version at the command prompt you opened in the previous step.

If the JVM is properly installed and available, the java -version command displays the version of your JVM. You must have version 1.2 or greater.

If the version is not displayed, review the installation of your JVM and your classpath.

5. From the command prompt, which is open to the /agents/tools directory, run the PDBGenerator utility. This command creates a Palm database file with your

updated configuration parameters for the device agent. Type the command as a single line (for readability, the command below appears on three lines):

```
java -cp $WAS_HOME/installedApps/hostname_DMS_Webapp.ear/dmsserver.war/WEB-INF \
/lib/pdbgene.jar com.tivoli.dms.tool.pdbgene.PDBGenerator
Config.ini.sample config.pdb
```

where:

\$WAS_HOME

Is the WebSphere installation directory

hostname

Is the name of the Web Gateway

The command format is:

```
java -cp JAR_file_directory JavaClassName arguments
```

where:

JAR_file_directory

path to the pdbgene.jar file

JavaClassName

com.tivoli.dms.tool.pdbgene.PDBGenerator

arguments

Config.ini.sample config.pdb

Administrative Tasks for Palm OS Devices

Administrative tasks that you might need to do in the future for these devices include the following:

- “Changing a Configuration Parameter”
- “Changing the Value for the Network Attachment Option” on page 338
- “Manually Setting Up a Modem Connection” on page 339
- “SSL and the Device Agent” on page 341
- “Disabling the Auto-connection When Using a Cradle” on page 341
- “Using a Security Manager” on page 342

Changing a Configuration Parameter

During a typical device setup scenario, the network administrator assigns values to configuration parameters for the device agent. After the initial device setup, the administrator can change a configuration parameter value with a device configuration job or by creating an updated config.pdb file and distributing that file to the Palm OS device.

To change a configuration parameter value, do the following steps:

1. Open the Config.ini.sample file in a text editor.
2. Include the configuration parameters that you want to change in your configuration file. Use the configuration parameter descriptions and the Config.ini.sample file as guides.

For example, the following two lines allow PPP queries for a DNS address (DNSQuery=1) and change the IP address for the secondary DNS.

```
DNSQuery=1
SecondaryDNS=9.68.1.15
```

Only the configuration parameters listed in the configuration file are changed. All other configuration parameters retain their previous values.

3. Convert the configuration file to a Palm database file, called config.pdb, with the PDBGenerator command-line utility.

The conversion file utility is part of the pdbgene.jar file, which is installed with Web Gateway. Type the command as a single line (for readability, the command below appears on three lines):

```
java -cp $WAS_HOME/installedApps/hostname_DMS_Webapp.ear/dmserver.war/WEB-INF \
/lib/pdbgene.jar com.tivoli.dms.tool.pdbgene.PDBGenerator
Config.ini.sample config.pdb
```

where:

\$WAS_HOME

Is the WebSphere installation directory

configuration file

Is the name of the configuration file you created.

You might have to set up the JVM or identify the classpath before using the PDBGenerator utility.

4. Either copy the config.pdb file from the host PC to the Palm OS device with HotSync Manager or distribute the config.pdb file to the device as part of a software distribution job.
5. Run the device agent to implement the change to the configuration parameters.
When the device agent next runs, it sets the new values for the configuration parameters from the updated config.pdb database file.

Changing the Value for the Network Attachment Option

A Palm OS device uses either a modem or a cradle (with the HotSync Manager program) to connect to the network. For the device agent, the network connection type is determined by the value of the **AttachmentOption** parameter in the Config.ini.sample file.

The default value (3) displays a prompt and allows the user to choose the connection type. You must use a configuration file to change the value for the **AttachmentOption** parameter.

The values for the **AttachmentOption** parameter are the following:

AttachmentOption=0

Specifies that the Palm OS device should automatically detect the type of connection (modem or cradle) being used.

If **AttachmentOption=0** and there is no modem attached to the Palm device (no physical modem connection) and no cradle connection, then the device agent assumes a mobile modem connection is being used.

AttachmentOption=1

Specifies that the Palm OS device should always use the modem connection. The modem can be physically attached to the Palm OS device or connected to the device through a mobile phone (mobile modem).

If you are using a mobile phone connection, the **Service** parameter is generally set to your mobile phone connection.

AttachmentOption=2

Specifies that the Palm OS device should always use the cradle connection with the HotSync Manager program.

AttachmentOption=3

Specifies that the Palm OS device should display a dialog box and allow the user to choose the connection type (cradle or modem). This is the default value.

The value for the **AttachmentOption** parameter must be changed using the Config.ini.sample file. To change the value for this parameter, do the following steps:

1. Set the **AttachmentOption** parameter to the desired value in the Config.ini.sample file.
2. Use the PDBGenerator command-line utility to convert the Config.ini.sample text file to a Palm database file (config.pdb).

The conversion file utility is part of the pdbgene.jar file, which is installed with Web Gateway. Type the command as a single line (for readability, the command below appears on three lines):

```
java -cp $WAS_HOME/installedApps/hostname_DMS_Webapp.ear/dmserver.war/WEB-INF \
/lib/pdbgene.jar com.tivoli.dms.tool.pdbgene.PDBGenerator
Config.ini.sample config.pdb
```

where:

\$WAS_HOME

Is the WebSphere installation directory

You might have to set up the JVM or identify the classpath before using the PDBGenerator utility.

3. Use HotSync Manager to copy the config.pdb file from the host PC to the Palm OS device.
4. Run the device agent to implement the change to the **AttachmentOption** parameter.

Manually Setting Up a Modem Connection

In some situations, the network administrator has the device owner do the setup for a modem connection manually. For this, the device owner does as follows:

- Creates a network interface using the PalmOS Prefs application
- Configures the modem using the PalmOS Prefs application
- Configures the device agent

To setup a modem manually, the device owner must enter the needed configuration parameters, which are generally supplied by the network administrator as part of the installation instructions.

If the device owner receives the following warning message from the device agent at the initial connection, then manual setup for a modem connection is required.

Initial configuration is not done automatically,
please set required parameters manually.

Creating a Network Interface (Manual Modem Setup)

To create a network interface using the PalmOS Prefs application, do the following steps:

1. Open the Palm OS Prefs application on the device.
2. Select the **Network** category.
3. Click the **Menu** icon.
4. From the **Service** menu, click **New**.

5. In the **Service** field, type the service name, for example, **DevAgent**.
6. Type the PPP user name and password in the **User Name** and **Password** fields.
7. Click **Phone** to open the Phone Setup dialog box.
 - a. In the **Phone #** field, type the telephone number.
 - b. As needed, select and type the phone settings for a dial prefix, call waiting, and use of a calling card.
 - c. Click **OK**.

The network interface is created. The next step is configuring the modem.

Configuring the Modem (Manual Modem Setup)

To configure the modem using the PalmOS Prefs application, do the following steps:

1. Open the Palm OS Prefs application on the device.
2. Select the **Modem** category of the Prefs application.
3. From the **Modem** list, select the modem type, for example a Palm V Modem.
4. As appropriate, select the other modem preferences, such as speed and flow control.
5. If needed, click **String** to change the default modem setup string.

In some situations, the default **String** value (**AT&FX4**) causes a connection error. If you receive an error, change the **String** value to **AT&FX3**.
6. Set the dial type to **TouchTone** or **Rotary**.

The modem configuration is complete. The final step of the manual modem setup is configuring the device agent.

Configuring the Device Agent (Manual Modem Setup)

To configure the device agent, do the following steps:

1. Click the device agent icon on the PalmOS application list.

If the device agent has not been configured, the following warning message is displayed:

No configuration data.
2. At **Manual Setup**, click **Hide**. The Logon dialog box is displayed.
3. In the **Username** field, type the user name. For devices that do not have a serial number, such as the Palm IIIe, the user name is the device ID string, which is 12 bytes in length.

If you are using IBM Tivoli Access Manager WebSEAL, use the same user name that authenticates with Policy Director.
4. In the **Password** field, type a password.

If you are using IBM Tivoli Access Manager WebSEAL, use the same password that authenticates with Policy Director.
5. In the **ServiceName** field, type the name which should be the same as **Username**.
6. Click **OK**. A Server Address Missing warning message is displayed. Click **OK** again.
7. Click the **Menu** icon and select **Options ” Server Setting** to display the Server Information dialog box.
8. In the **Address** field, type the host name or IP address of the Web Gateway server. If you are using IBM Tivoli Access Manager WebSEAL, type the host name of the WebSEAL server.

9. In the **Port** field, type the port number. The default is 80.
10. The **Servlet Name** field does not need to change unless you are using IBM Tivoli Access Manager WebSEAL. If you are using IBM Tivoli Access Manager WebSEAL, type the following for the **Servlet Name** field:
`/junction_name/dmservlet/PalmServlet`
11. Click **OK** to close the Server Information dialog box.
12. Click **Connect**.

SSL and the Device Agent

Palm OS does not supply any SSL functions, so the Palm OS agent provides the elements needed to implement secure connections. When SSL is enabled, consider the following:

- There is 128-bit encryption as well as data integrity checking of communications between the device agent and the plug-in.
- The HTTP server must be set up to support SSL connections.
- The port for an SSL connection is whatever port is configured on the device agent (port 443 is not the default port).

The Palm device gives no indication that SSL is enabled or disabled. In addition, unlike many configuration parameters, the device owner cannot enable or disable SSL using the configuration dialog boxes on the Palm OS device.

To enable or disable SSL on the device, you must customize the **SSLon** option on the device. Refer to the "Creating a Software Package for Devices" chapter in the *User's Guide for Software Distribution* for details.

Enabling SSL

The value for the **SSLon** parameter determines if SSL is used for PalmOS devices. If you are using the PDBGenerator utility, this parameter is set in the `Config.ini.sample` file. To enable SSL, do the following steps:

1. Set the **SSLon** parameter to **1 (SSLon=1)** in the `Config.ini.sample` file (refer to the sample `Config.ini.sample` file).
2. Optionally, change the **DMSPort** parameter to **443** in the `Config.ini.sample` file.

Disabling SSL

The value for the **SSLon** parameter determines if SSL is used for PalmOS devices. This parameter is set in the `Config.ini.sample` file. To disable SSL, do the following steps:

1. Set the **SSLon** parameter to **0 (SSLon=0)** in the `Config.ini.sample` file.
2. Optionally, change the **DMSPort** parameter to **80** in the `Config.ini.sample` file.

Disabling the Auto-connection When Using a Cradle

With the device agent conduit installed, each time there is a synchronization between the Palm OS device and the host PC, the device agent connects to the Web Gateway server through the device agent conduit. All jobs that are waiting for the device in the job queue are processed.

To disable this auto-connection, do the following steps on the Palm OS device:

1. Display the applications list.
2. Click the **HotSync** application.
3. Click **Menu**.

4. From the **Options** menu, click **Conduit Setup**.
5. Scroll through the list of conduits to locate the **IBM agent**.
6. Click the **IBM agent** in the list so the checkmark is cleared.
7. Click **OK**.

Using a Security Manager

To provide for increased security in a network, you can place the Web Gateway behind a security manager which requires basic HTTP authentication. The IBM Tivoli Access Manager WebSEAL is such a security manager.

Where a security manager is used, the device agent connects to the Web Gateway through the security manager. The device agent must provide a user ID and password for the basic HTTP authentication. The user ID and password are passed in the authentication request to the security manager. Once authenticated, the connection is passed to the Web Gateway.

To configure the device agent with a security manager, type the host name for the security manager in the **DMSAddress** field and the junction qualified servlet name in the **PalmServletName** field when you configure the device agent. You must also configure the enrollment servlet for the plug-in in the Web Gateway to use a security manager based Web address.

Installing the Device Agent for Windows CE Devices

Windows CE devices are those devices that use the Microsoft Windows CE for Handheld PC Professional Edition operating system. These include handheld PC, pocket-type, or sub-notebook devices for sales and service personnel, mobile business professionals, and other field personnel who need access to their network or the Internet

Such devices require the plug-in for Windows CE and device agent for Windows CE. The plug-in and device agent perform system management tasks and communicate with each other by using HTTP.

Communicating with the Host PC

To establish communications between your Windows CE device and the host PC, you must install Windows CE Services with ActiveSync on the host PC. These programs operate together to synchronize, back-up, and restore information on your device, add and remove files from your device, and copy data between your device and the host PC.

Once Windows CE Services is installed on the host PC, you are prompted to create a partnership with your Windows CE device using a serial cable connection.

Planning for the Installation

To plan for the installation and configuration for a Windows CE device, consider the following:

- Determine the CPU type
The CPU type and file structure differs among the various Windows CE devices. For each CPU type, there is a different device agent installation package.
- Determine the connection type

Depending on the device type, you can configure the device agent to use one or more connection types. However, for all connection types, the device must be able to connect to the plug-in on the Web Gateway server, which is specified by the **Server Address** field.

For devices using Pocket PC 2002, consider the following:

- The ActiveSync connection with a cradle uses a RAS connection to allow communication through the host PC to the plug-in.
- A LAN connection or modem connection can also be used for communication directly with the plug-in.
- If you have a modem on the host PC, you can connect from the device to the host PC with a RAS connection, and then to the plug-in.

For older Windows CE devices such as Pocket PC, consider the following:

- You can use a LAN connection or a modem connection for communication directly with the plug-in.
- Where a LAN connection is used, you must install the LAN driver and the LAN adapter on the device.
- Where a modem connection is used, you must create a “dial-up network connection” for each network that you want to access remotely.
- Where a modem is used on the host PC, you can connect from the device to the host PC with a RAS connection, and then to the plug-in.

Installing the Device Agent

Using the host PC, do the following steps:

1. Copy the appropriate device agent installation package CAB file from the Web Gateway server to the host PC.

For IBM AIX

/install_directory/agents/wince/PocketPC/CPU_type

where:

- The default *install_directory* is *usr/TivTwg*
- *CPU_type* can be *arm* or *x86*

For Sun Solaris Operating Environment, Linux and HP-UX

/install_directory/agents/wince/PocketPC/CPU_type

where:

- The default *install_directory* is */opt/TivTwg*
- *CPU_type* can be *arm* or *x86*

For Microsoft Windows

install_directory\agents\wince\PocketPC\CPU_type

where:

- The default *install_directory* is *c:\TivTwg*
- *CPU_type* can be *arm* or *x86*

2. Copy the device agent installation package CAB file from the host PC to the device by doing the following steps:
 - a. For your host PC, locate the installation package CAB file for the device agent.
 - b. Open the ActiveSync Desktop.
 - c. Select the **Explore** icon from the ActiveSync Desktop to open the Mobile Device dialog box.

Installing Device Agent for Windows CE Devices

- d. If you do not have a **Temp** directory on your device, in the Mobile Device dialog box, do the following steps:
 - 1) Click **File**.
 - 2) Click **New Folder**.
 - 3) Name the folder **Temp**.
- e. Copy the installation package CAB file from your host PC to the **Temp** directory on your device.

It might take a few minutes to copy this file.
3. After copying the files on the device, perform the following steps:
 - a. Use Windows Explorer to navigate to the **Temp** directory. It might take a few minutes for the file to be displayed in the directory on your device.
 - b. Double-click the installation package CAB file for the device agent.

When the installation is complete, the device agent, named IBM agent, is displayed on the **Start »Programs** list.

Configuring the Device Agent

If needed, you can modify the device agent configuration after the initial configuration is complete.

To configure the device agent, do the following steps:

1. Open the device agent by clicking on **Start »Programs »IBM agent**

The Device Agent dialog box opens with the following buttons:

Connect

Connect to the Web Gateway server and look for waiting jobs. If you are using IBM Tivoli Access Manager WebSEAL, the first time that you connect to Web Gateway, you are asked for the user ID and password that have been registered in Policy Director.

Hide Minimize the Device Agent dialog box and put an icon for the device agent in the Windows CE tray.

Exit Close the IBM agent program.

Configure

Open the configuration dialog box for the device agent.

2. Click **Configure**. A window with the following pages opens:

- **General Page**
- **Connection Page**
- **Proxy Page**

3. In the **General Page** do the following steps:

- a. If you use IBM Tivoli Access Manager WebSEAL, for the **User ID** field, type the information supplied by your network administrator.
- b. For the **Server URL** field, type the Web address that points to the plug-in for Windows CE on the Web Gateway server. In the Web address example below, for *server_name* substitute the Web Gateway server.

Type this Web address, as modified by the server name, for the **Server URL** field.

`http://server_name/dmservlet/WinceServlet`

If you use IBM Tivoli Access Manager WebSEAL, type:

`http://WebSEAL_hostname:port/junction_name/dmservlet/WinceServlet`

where:

WebSEAL_hostname:port

Is the hostname and port of the WebSEAL server

junction_name

Is the name of the junction that has been configured in WebSEAL to redirect the communication to WebSphere.

- c. If necessary, clear the **Poll automatically** check box. The check box is selected by default. Generally, you want the **Poll automatically** field cleared.

Poll automatically check box selected

The device agent automatically starts and connects to the Web Gateway server. Waiting jobs are processed. Polling for jobs occurs at the interval set by the `mgmt.pollingtimer` keyword. The connection and polling continues until you click **Exit**. For more information on this keyword, see Chapter 18, "Device Customization Parameters," on page 377.

For automatic polling to occur, the device must be powered on. Automatic polling does not power on the device.

Poll automatically check box cleared

There is no automatic connection to the Web Gateway server.

To connect to the Web Gateway server, click **Connect**. The device connects to the Web Gateway server, waiting jobs are processed.

4. In the **Connection Page** do the following steps:

- a. For a modem connection, in the **Access point** field, select the access point from the drop-down list. You can create additional access points by configuring additional modem connections with the Windows CE configuration prompts.
- b. Determine the setting for the **Dial at startup** check box based upon using a LAN connection, a RAS connection, or a modem connection. Cleared is the default.

For a LAN or RAS connection, make sure the **Dial at agent startup** check box is cleared.

For a modem connection, set the **Dial at agent startup** check box to control the startup behavior as follows:

- When the **Dial at agent startup** check box is selected, the device agent automatically opens the Dial-up Connection dialog box so a connection can be established when the device agent starts. If a connection is already established, the device agent does not open the Dial-up Connection dialog box.
- When the **Dial at agent startup** check box is cleared, the user has to initiate the opening of the Dial-up Connection dialog box in Windows CE to establish a PPP connection. Establish the connection before the device agent is started.

5. In the **Proxy Page** do the following steps:

- a. As appropriate, set the **Proxy enable** check box to use a proxy server. The check box is cleared by default.
- When selected, the device agent connects to the Web Gateway server via the proxy server.

Installing Device Agent for Windows CE Devices

- When cleared, the device agent connects to the Web Gateway server directly.
- b. If you are using a proxy server, in the **Proxy address** and **Proxy port** fields type the information supplied by your network administrator.
- c. Click Save. The Device Agent dialog box is displayed.

Configuration File

The Windows CE device agent can be configured with a configuration file. A configuration file assigns values to configuration parameters. The configuration parameters included in the file vary based upon the connection type, network parameters, browser configuration, security needs, and so on.

Distributing and Processing the Configuration File: The IBM_DeviceAgent.ini file must be placed in either the root ("\\") or the My Documents folder on the device. The placement of the file depends on the locale.

If the IBM_DeviceAgent.ini file exists in either the root ("\\") or the My Documents folder on the device, the device agent automatically processes the configuration file when it starts to set the configuration values and the user is not prompted to supply configuration values using the configuration windows on the device.

The IBM_DeviceAgent.ini file is processed before any Device Manager jobs are run during the session. This allows configuration file processing to occur whenever the administrator needs to load a new IBM_DeviceAgent.ini file on the device.

Configuration File Format: The configuration file, which is named IBM_DeviceAgent.ini, uses a keyword=value pair format. The keywords can be any of the configuration parameters and fully qualified registry entries for Windows CE devices. Do not include spaces around the equal sign (=).

Specify only one keyword=value pair per line and end the line with a CRLF or LF. The configuration file is case sensitive so ensure a keyword or registry entry uses the correct capitalization of characters. The encoding of the file must be UTF-8.

Use the sample configuration file, located in *TWG_home/agents/wince/PocketPC/IBM_DeviceAgent.ini*, as a guide when preparing your file.

```
#-----Begin Copyright - do not add comments here-----
#
# 5724-B07
# (C) Copyright IBM Corp. 2003. All Rights Reserved.
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with
# IBM Corp.
#
#
#-----End Copyright-----
# This is a sample IBM_DeviceAgent.ini file. The format of this file is as follows:
# - Lines beginning with a # sign in column zero are commented (ignored).
# - Remove the # sign and modify the value (as appropriate) for any parameters
#   that you want to configure.
# - Each key=value pair must not have extraneous spaces anywhere in the line.
#   I.e., spaces are treated as normal characters and not stripped.
# - See the "Device Configuration" | "Configuration Parameters" section of the
#   documentation for details regarding appropriate values for all fields.

# Agent (Mgmt) Parameters
mgmt.serveraddr=http://1ab134048/dmservlet/WinceServlet
mgmt.delete_ini_after_processing=no
```

```
#mgmt.pollingtimer=1
#mgmt.pollingschedule=01:00,05:00,09:00,13:00,17:00,21:00
#mgmt.scheduleoffsetmax=10
#mgmt.agentrunmode=0N
#mgmt.proxy.enable=0FF
#mgmt.proxy.addr=mgmt.proxy.com
#mgmt.proxy.port=80
mgmt.auth.user=andreotto
#mgmt.auth.pass=somepassword
#mgmt.savepassword=0N
#mgmt.device.id=UniqueDeviceID34321
#mgmt.maxlogsize=150000

# Browser Parameters
#bsr.startpage=http://somepage.domain.com
#bsr.proxyaddr=proxy.com
#bsr.proxyport=80
#pct.enable=0N
#ssl2.enable=0N
#ssl3.enable=0N

# PPP Parameters
#ppp.dial=555-555-5555
#ppp.user=pppuser

# TCP/IP Parameters
#net.dns1=1.2.3.4
#net.dns2=1.2.3.5

# Mailer Parameters
#pop3.server=pop3srv.domain.com
#smtp.server=smtpsrv.domain.com
#pop3.uid=pop3user
#mail.address=someid@pop3server.domain.com
```

Sample Configuration File for Windows CE Devices: A sample IBM_DeviceAgent.ini file is:

```
mgmt.serveraddr=http://dms.rtp.ibm.com/dmserver/WinceServlet
mgmt.auth.user=pat_smith
mgmt.auth.pass=cool_password
```

Setting the Device ID: The device ID is a unique identifier for the device. You can set the device ID with the `mgmt.device.id` keyword in the configuration file.

If a device ID is set with the `mgmt.device.id` parameter in a configuration file, it is imperative that each device has a unique value across the management environment. The device ID is used when accessing the Device Manager database.

If no value is set, the Tivoli Web Gateway generates a device ID when the device connects to the Tivoli Web Gateway for the first time.

If you are changing the value of a device ID, the new value does not take effect until you exit the device agent and restart the device agent.

Setting the Authorization Password: The authorization password for the device owner can be set with the `mgmt.auth.pass` keyword in the configuration file. The authorization password is not stored in the Device Manager database nor passed over the network.

Scheduled Polling: For the Windows CE device agent, you can set parameters in the configuration file, IBM_DeviceAgent.ini, to allow for scheduled polling. This

Installing Device Agent for Windows CE Devices

means you set the exact time the device agent should connect to the Web Gateway server and check for jobs that are waiting for the device.

The polling schedule is independent of the polling interval and one or both can be used with a device. There are 2 parameters that can be set in the configuration file: `mgmt.pollingschedule` and `mgmt.scheduleoffsetmax`.

The polling schedule, `mgmt.pollingschedule`, is specified with a comma-separated list of 24-hour times during each day.

The offset, `mgmt.scheduleoffsetmax`, sets the offset in minutes for the polling schedule. It is the maximum number of minutes to "offset" each schedule polling time. Here is an example:

```
mgmt.pollingschedule= 06:00,13:30,15:30
mgmt.scheduleoffsetmax=20
```

Keeping Device Agent Files on Windows CE

To ensure a persistent one-click installation of the device agent, after a hard reset or when the battery comes down, you can use either expansion cards or persistent paths. Both of them prevent the .ini and CAB files from being deleted after they are processed.

To install and configure the device agent on WinCE with a persistent one-click installation, perform the following steps:

1. Set the `mgmt.delete_ini_after_processing` keyword to no in the `IBM_DeviceAgent.ini` configuration file.
2. Copy the `IBM_DeviceAgent.ini` configuration file and the device agent installation package CAB file in the root of a persistent store path of the device.
3. Create a launcher link file, `InstallAgent.lnk`, that uses the **wceload** option to not remove the cab file.

```
66#\windows\wceload.exe /delete 0 \Built-in Storage\ceagent.CPU_typeCAB
```

where:

66 Represents the number of characters that follow the # symbol.

windows\wceload.exe /delete 0

Represents the command that prevents the CAB file from being deleted.

\Built-in Storage\ceagent.CPU_typeCAB

Represents the full path of the CAB file. In particular:

\Built-in Storage

Is the persistent path. For example the built-in storage path on HP iPAQ PocketPC h4100 is \iPAQ File Store.

CPU_type

Is the type of device. It can have the following values: arm or x86

4. Copy the `InstallAgent.lnk` file in the root of a persistent store path.

In this way, any time you need to install and configure the device agent on WinCE, you must just click the `InstallAgent.lnk` file on the device.

Administrative Tasks for Windows CE Devices

Administrative tasks that you might need to do in the future for these devices include the following:

- Using SSL with the device agent
- Using a security manager

Using SSL with the Device Agent

Web Gateway relies on the SSL features supplied with Microsoft Windows CE. When SSL is enabled, consider the following:

- There is 128-bit encryption of communications between the device agent and the plug-in.
- As an option, there is server authentication against a list of locally trusted Certificate Authority certificates. If the server being connected to is not validated by a locally trusted certificate, then the device agent user has the option of continuing with the connection anyway.
- The HTTP server must be set up to support SSL connections.
- The default port for an SSL connection is 443.

You enable SSL in the device agent for Windows CE by setting the **Server URL** field to begin with HTTPS:// (secure SSL). When SSL is enabled, a lock icon is displayed in the lower left corner of the device agent status dialog box.

To disable SSL in the device agent, set the **Server URL** field to begin with HTTP:// (non-secure). When SSL is disabled, the lock icon is not displayed in the device agent status dialog box.

If the device agent attempts an SSL connection to an HTTP server with a certificate that is not matched by a locally trusted Certificate Authority Certificate, the device agent asks the user if it can use the connection anyway.

Managing the List of Locally Valid Certificate Authority Certificates

To manage the list of locally valid Certificate Authority Certificates on Windows CE, Microsoft has documented the registry values that must be maintained. Microsoft supplies a root certificate utility program to change the registry values.

Refer to the *Microsoft SQL Server 2000 Windows CE Edition* documentation for the SSL features and steps to update the database of trusted Certificate Authorities on Windows CE.

Using a Security Manager

To provide for increased security in a network, you can place the Web Gateway behind a security manager which requires basic HTTP authentication. The IBM Tivoli Access Manager WebSEAL is such a security manager.

Where a security manager is used, the device agent connects to the Web Gateway through the security manager. The device agent must provide a user ID and password for the basic HTTP authentication. The user ID and password are passed in the authentication request to the security manager. Once authenticated, the connection is passed to the Web Gateway.

Administrative Tasks for Windows CE Devices

To configure the device agent with a security manager, type the Web address for the security manager in the **Server Address** field when you configure the device agent. You must also configure the enrollment servlet for the plug-in in the Web Gateway to use a security manager based Web address.

Administrative Tasks for Nokia Devices

Nokia devices are provided with the device agent already installed by the device vendor.

Enabling security for Nokia s60 devices

This section describes the steps needed to configure the Tivoli Configuration Manager environment in order to successfully use the Nokia s60 devices and the actions supported on these devices.

Configuring the DMS server

Configure the Device Management Server (DMS) server on the Tivoli endpoint by editing the `web.xml` file. In this XML file both parameters `DefaultServerID.NokiaOMADM` and `DefaultServerPwd.NokiaOMADM` are set by default to IBM. You can modify these parameters and then restart the `DMS_AppServer` application server, to make the changes effective.

Configuring the IBM HTTP server

Modify the `httpd.conf` file on the IBM HTTP server accordingly, to enable the server to manage HTTPS connections.

Configure the IBM HTTP server certificate on the Tivoli endpoint, by ensuring that the values set for the certificate keys `label` and `common name` are the same values.

The values for the certificate keys `label` and `common name` are either the IP Address or the fully qualified host name of the Tivoli endpoint on which the IBM HTTP server resides.

Downloading the certificate on the device

Export the certificate used in the HTTPS setup as a `.CER` or `.DER` file and copy the certificate from the Tivoli endpoint to the Nokia s60 device, using Bluetooth or the Internet browser of the device. When the certificate file is displayed in the inbox of the Nokia device, trust the application by selecting both check boxes, and save the certificate.

Enabling the WebSphere Application Server security

On the WebSphere Application Server enable the Global Security check box and disable the Java Security check box. You must also specify the type of user registry you want to use.

Creating a connection profile on the Nokia s60 device

To create a connection profile, perform the following steps on the Nokia device:

1. Go to Menu -> Connectivity -> Device mgr -> Options -> New server profile.
2. In the server profile enter the following information:

Server name

Enter a name for the connection profile.

Server ID

Enter the same value specified for the `DefaultServerID.NokiaOMADM` parameter in the `web.xml` file.

Server password

Enter the same value specified for the DefaultServerPwd.NokiaOMADM parameter in the web.xml file.

Session mode

Enter Internet.

Access point

Enter the name of the access point which provides the connection GPRS, WiFi, or other connection types.

Host address

Enter the following URL:

`https://TWG_workstation/dmservlet/OMADMServletAuthRequired`

where:

TWG_workstation

Is the same address specified as label and common name in the IBM HTTP server certificate on the Tivoli endpoint.

Port Enter 443, the default port number.

User name

(Option valid only for devices created on the Tivoli server). Enter the user ID specified during the manual device creation on the Tivoli server.

Password

(Option valid only for devices created on the Tivoli server). Enter the user password specified during the manual device creation on the Tivoli server.

Allow configuration

Enter Yes.

Auto-accept all requests

Enter Yes.

Network authentication

Enter Yes.

Network user name

Enter a user name existing in the user registry specified for the WebSphere Application Server security.

Network password

Enter a password for the network user name.

Chapter 17. Managing Resources

Resource Manager enables you to examine the computers in your enterprise to determine where resources, either pervasive devices or users, are associated. In this chapter, the term *resources* is used when an action or operation refers to both pervasive devices and users. Otherwise, the term *devices* or *users* is used to refer to the specific resource type.

To manage resources, you must do the following main tasks:

1. Register the resource types.
2. Register the endpoints as resource gateways. A resource gateway is an endpoint that has applications that extend the endpoint to manage pervasive devices.
3. Manage the resources in the Resource Manager database (for pervasive devices only). For example enroll, create, discover, list, modify, or delete the resources.
4. Organize the resources in groups.

This chapter describes how to perform the following tasks from the Tivoli desktop. For instructions on the command line, see Chapter 19, “Using the Command Line,” on page 387.

Task	See Page
Registering the Resource Types	354
Registering Endpoints as Resource Gateways	355
Enrolling Pervasive Devices Automatically	355
Discovering Devices	356
Creating Devices	357
Grouping Resources	369
Listing and Finding Resources	373
Modifying Devices	360
Deleting Devices	361
Customizing Resource Manager Actions	375

After you create or enroll and group your pervasive devices, you can work with pervasive devices. You can perform the following operations:

- Run programs
- Remove software packages installed on the device using Software Distribution
- Customize device settings
- Scan pervasive devices for software and hardware information
- Provide initial configuration parameters to Nokia devices
- Install and configure applications on Nokia devices

To perform these tasks, you must create a software package that contains the actions that you want to perform on the devices and distribute it to the target devices.

Managing Resources

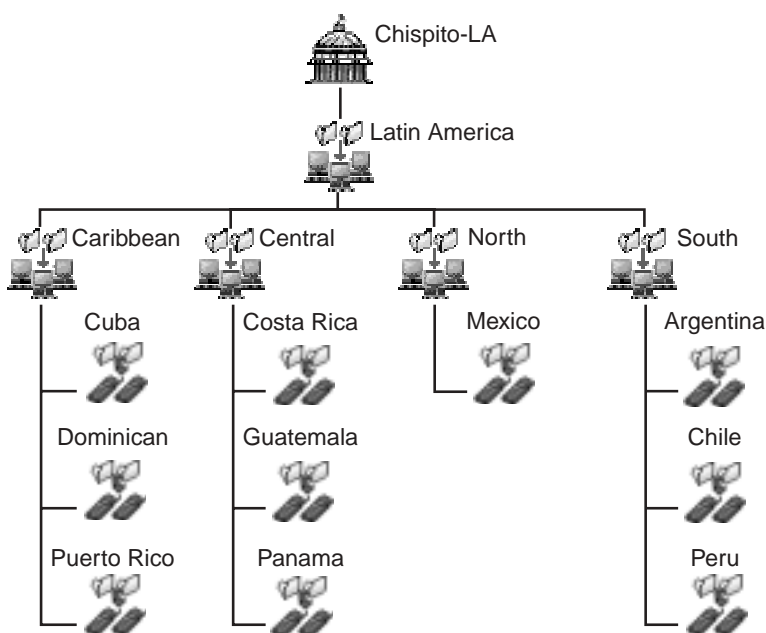
You can create the software package either by using the Software Package Editor (see “Creating a Software Package for Devices” in the *User’s Guide for Software Distribution*) or by creating a software package definition file (see “Editing the Software Package Definition (SPD) File” in the *Reference Manual for Software Distribution*).

In addition, you can scan pervasive devices for software and hardware information. For information about how you do this, see the *User’s Guide for Inventory*.

Sample Tivoli Environment

The concepts and examples described in this and subsequent chapters are illustrated using a fictitious international company named Chispito, based in Latin America.

At Chispito, the region **Chispito-LA** contains a profile manager **Latin America**,



which contains the profile managers, **Caribbean**, **Central**, **North**, and **South**. Each profile manager contains a resource group for each country office in its area. The profile manager **Caribbean** has the resource groups **Cuba**, **Dominican**, and **Puerto Rico**; the profile manager **Central** has resource groups **Costa Rica**, **Guatemala**, and **Panama**; the profile manager **North** has resource group **Mexico**; and the profile manager **South** has **Argentina**, **Chile**, and **Peru**.

Registering the Resource Types

To register the resource types with which you will work, run a script to start each type. These scripts are installed in the directory \$BINDIR\TRM, when you install Resource Manager.

RegisterUser.sh

To start the User resource type. You must have Enterprise Directory Query Facility installed before you run this script.

RegisterPervasive.sh

To start the Pervasive resource type

If a Resource Manager is already installed with User or Pervasive Devices registered, you must unregister and then reregister the resource type. The command to unregister, for example, the resource type `<Pervasive_Device>` is:

```
wresource remove_type Pervasive_Device
```

Use the **wresource ls** command to list all registered devices.

You then must run the relevant script (either *RegisterUser.sh* or *RegisterPervasive.sh*) to register the same resource type again.

Working with Pervasive Devices

The following sections describe the tasks that you can perform on pervasive devices.

Registering Endpoints as Resource Gateways

Before you can start working with devices, you must perform the following steps:

1. Install the resource gateway on the workstation you plan to use as a resource gateway. The Tivoli endpoint is installed automatically with the resource gateway, if not already present. You can also install the resource gateway on an existing endpoint. The supported resource gateway is Tivoli Web Gateway.
2. Register the resource gateway with the Tivoli server. To perform this operation, use the **wresgw add** command. For example, to associate a resource gateway with the endpoint `rvargas`, type the following command:

```
wresgw add rvargas -C TWG
```

3. Run a discovery operation to align the Web Gateway database with the Resource Manager database. For more information on the discovery operation, see “Discovering Devices” on page 356.
4. Enroll or create devices as described in the following sections.

To manage resource gateways from any managed node in interconnected regions, you must run the **wresgw add** command from each region.

See “wresgw” on page 391 for more details.

Enrolling Pervasive Devices Automatically

When you connect a device to the Web Gateway, it is registered in the Web Gateway database. With automatic enrollment, these resources are automatically registered in the Resource Manager database. Devices must be registered in the Resource Manager database (enrolled or created) to enable them to be managed in the Tivoli environment.

If you want to manually create devices, refer to “Creating Devices” on page 357.

Task	Context	Required Role
Enrolling devices automatically	Tivoli desktop or CLI	admin

For instructions on how to perform this task from the command line, see **wresgw autoenroll** on page 391.

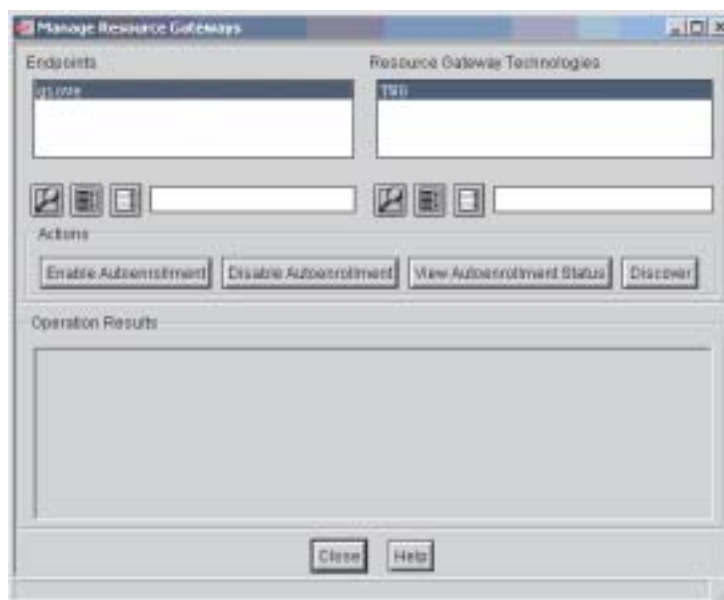
Working with Pervasive Devices

To enroll a device automatically from the Tivoli desktop, perform the following steps:

1. From the Tivoli desktop, double-click the **Resource Manager** icon. The Resource Type Table dialog is displayed:



2. Select the resource type **Pervasive_Device** and click **Manage**. The Manage Resource Gateways dialog is displayed:



3. Select an entry from the **Endpoints** list and an entry from the **Resource Gateway Technologies** list. Click **Enable Autoenrollment**.
4. Click **Close**. The device is enrolled automatically.

Discovering Devices

During discovery operations, Resource Manager checks for devices that have already been defined in the Web Gateway database on the endpoint but have not been created in the Resource Manager database.

To ensure that the Web Gateway database is aligned with the Resource Manager database, run a discovery operation after registering the endpoint as resource gateway and repeat discovery operations on a regular basis.

Task	Context	Required Role
Discovering Devices	Tivoli desktop or CLI	admin

For instructions on how to perform this task from the command line, see **wresgw discover** on page 259.

To discover devices, perform the following steps:

1. From the Tivoli desktop, double-click the **Resource Manager** icon.



The Resource Type Table dialog is displayed.

2. Select the type **Pervasive_Device** and click **Manage**. The Manage Resource Gateways dialog is displayed.
3. Select an entry from the Endpoints list and an entry from the Resource Gateway Technologies list. Click **Discover**.

Creating Devices

In addition to discovering devices in your environment, you can create them manually.

Task	Context	Required Role
Creating Devices	Tivoli desktop or CLI	admin

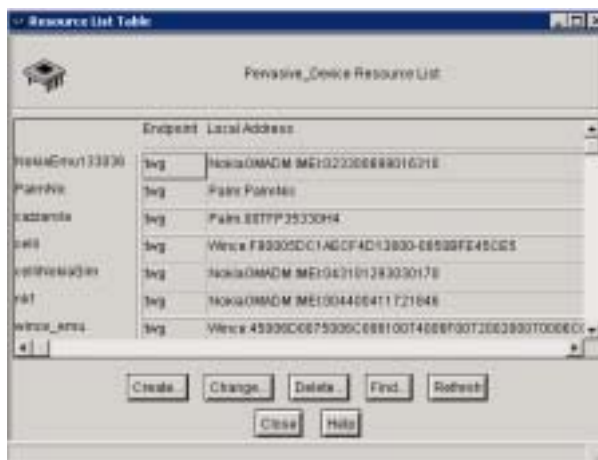
For instructions on how to perform this task from the command line, see **wresource add** on page 394.

Creating a device adds it to the list of resources. In the Tivoli Desktop, you can create devices one at a time by performing the following steps:

1. From the Tivoli desktop, double-click the **Resource Manager** icon. The Resource Type Table dialog is displayed.

Working with Pervasive Devices

2. Select the type **Pervasive_Device** and click **View**. The Resource List Table dialog is displayed.



3. Click **Create**. The Create a Resource dialog is displayed.



4. Type the following information in the appropriate fields:

Resource Label

An informal alphanumeric name to identify the resource in the Tivoli management region. If you installed the relational database management system extension, the resource label has a maximum of 128 characters. If you use special characters, put the name in double-quotation marks.

Endpoint Name

The name of the endpoint that is a resource gateway. Click **Endpoints** for a list of available endpoints.

Local Address

The local address can have a maximum of 180 characters. If you use Inventory queries, the local address must have a maximum of 64 characters. It must contain no blanks, asterisks (*) or semicolons (;). Use the format:

device_class:device_id

where:

device_class

NokiaOMADM:IMEI (for Nokia devices), Palm, or WinCE.

device_id

Indicates the IMEI number of Nokia devices or the serial number of Palm devices.

For Windows CE devices: The local address can either be generated automatically by the resource gateway or user-defined. When you create Windows CE devices from the desktop or the command line, use the following format:

Wince:user_name:device_name

where:

user_name

Is the user ID of the device, and is required

device_name

Is optional and must be separated from the *user_name* by a colon (:).

If you define the *device_name*, the final local address will be *Wince:devicename*. Otherwise, the resource gateway generates a local address in the format:

Wince:auto-generated address

This is the address that appears when you list the device after it is created.

Subtype

Specifies subtypes for the resource type *Pervasive_Device*. Valid subtypes are as follows:

- Nokia9300
- Nokia9500
- Nokia_s60
- Palm
- WinCE

Other Information

This option is available only for Nokia devices. Type the following information in the appropriate fields:

Device user name

Specifies the client name defined in the OMA DM protocol for authenticating the device to the Tivoli Web Gateway. The maximum length for the user name is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

Device user password

Specifies the password that the device uses to authenticate itself to the Tivoli Web Gateway. The maximum length for the device password is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

Server ID

Identifies the Tivoli Web Gateway. The maximum length for the

server ID is 255 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

Server password

Specifies the password for the server defined in the OMA DM protocol. The password must be unique for each device. The maximum length for the password is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

Dialing number

Specifies the telephone number of the mobile device in international format. Type a plus sign (+) followed by numbers with intervening hyphens (-) and periods (.) as necessary (for example: +41-1-123.1234 or +1-919-555-4321). This field is required.

NETWPIN Data

Specifies security information for sending provisioning data. This information guarantees a secure connection to the Tivoli Web Gateway. The maximum length for the NETWPIN data is 40 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). You must fill in at least the NETWPIN or USERPIN fields.

USERPIN Data

Specifies additional security information for sending provisioning data. This information guarantees a secure connection to the Tivoli Web Gateway. The maximum length for the USERPIN data is 40 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). You must fill in at least the NETWPIN or USERPIN fields.

Provisioning Document

Specifies the .xml file used for the initial provisioning of the device. This field is required.

Notification protocol

Displays the notification method assigned to the device. The notification protocol supported for this version is WAP Pushv12. This field cannot be edited.

5. Click **Create & Close**.

Modifying Devices

This section explains how to modify devices. You can change the resource label and a subset of the information for Nokia devices:

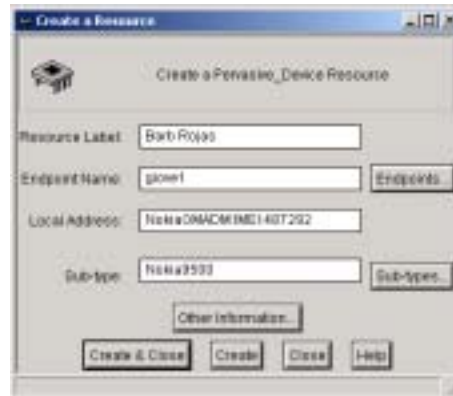
Task	Context	Required Role
Modifying devices	Tivoli desktop or CLI	admin

For instructions on how to perform this task from the command line, see **wresource edit** on page 394.

To modify the resource label of a device, perform the following steps:

1. List the device (see “Listing and Finding Resources” on page 373).

2. In the Resource List Table dialog, either double-click the device you want to modify, or select it and click **Change**. To modify more than one device, select all the devices required and click **Change**. The Change a Resource dialog is displayed.



3. You can change the information in the **Resource Label** text box only. For Nokia devices, you can also modify the following information stored in the **Other Information** dialog box:

- Device user name
- Device user password
- Server ID
- Server password
- Dialing number
- NETWPIN Data
- USERPIN Data
- Provisioning Document

For more information on these options, see “Creating Devices” on page 357.

4. After you have made your changes, click **Change & Close**.

If you have selected several resources, the dialog for the next resource is displayed. Repeat the previous steps until you have finished modifying all the resources you selected. The changes are displayed in the Resource List Table dialog.

Deleting Devices

This section explains how you delete devices that have been discovered or created.

Task	Context	Required Role
Deleting Devices	Tivoli desktop or CLI	admin

For instructions on how to perform this task from the command line, see **wresource remove** on page 394.

To delete a device, do the following:

1. List the devices defined on the endpoint (see “Listing and Finding Resources” on page 373).
2. In the Resource List Table dialog, select the device you want to delete and click **Delete**. The Delete a Resource dialog is displayed.



3. Click **Delete & Close**. The device is deleted from the **Resource List Table**.

To delete more than one device:

1. List the devices defined on the endpoint (see “Listing and Finding Resources” on page 373).
2. In the Resource List Table dialog, select the devices to be deleted and click **Delete**. The Delete a Resource dialog is displayed.
3. Click **Delete All**. The selected devices are deleted.

Nokia Provisioning Scenario

This scenario requires the following software installed and configured in addition to Resource Manager:

Tivoli Web Gateway

Is the resource gateway.

Device Manager server

Is an internal Configuration Manager component, installed automatically with Tivoli Web Gateway. Device Manager interacts seamlessly with Tivoli Web Gateway to manage devices and users.

IBM WebSphere Application Server (WAS)

Is the application server for Device Manager.

WebSphere Everyplace Connection Manager (WECM)

Is a distributed, scalable communications platform that supports secure data access by both Wireless Application Protocol (WAP) and non-WAP clients over a wide range of wireless network technologies, and local area (LAN) and wide area (WAN) wireline networks.

Device provisioning or initialization provides the pervasive device with the following features:

- A secure relationship with the Tivoli Web Gateway
- Key parameters for the initial connection to the Tivoli Web Gateway
- Repair of the initial configuration if it becomes damaged or corrupted

You can initialize one or more devices. The provisioning results and details are stored in the Tivoli Web Gateway database.

The following is a list of the operations involved in provisioning a Nokia device:

1. To use the WebSphere Everyplace Connection Manager with Device Manager, you must change the URL for the Push Proxy Gateway to the location where the IBM WebSphere Everyplace Connection Manager is installed. The URL is defined in the `push.properties` file, which can be found at the following location:

`$WAS_HOME/installedApps/node_name/DMS_AppServer/dmsserver.war/WEB-INF/classes`

where

\$WAS_HOME

Is the WebSphere installation directory.

node_name

Is the name of the node where WebSphere is installed

DMS_AppServer

Is the directory where the application server for Device Manager is installed.

In the `push.properties` file, change the **push.proxy-url** parameter to the URL of the connection manager. For example, if the connection manager is at 192.168.00.01 and the message services uses a non-secure port number 13131, the `push.proxy-url` parameter would have the following value:

`push.proxy-url=http://WECM.IBM.COM:13131`

You can also edit the following parameters:

push.logging

Specifies whether the logging function is enabled. Supported values are **true** and **false**.

push.tracing

Specifies whether the tracing function is enabled. Supported values are **true** and **false**.

push.pap-dtd-uri

Specifies the location of the Push Access Protocol (PAP) Document Type Definition (DTD). A Document Type Definition (DTD) is a formal description in XML declaration syntax of a particular type of document. It sets out what names are to be used for the different types of element, where they may occur, and how they all fit together. This document validates the content of the provisioning document used to initialize the device. Enable one of the available locations for the `pap_1.0.dtd` by removing the pound (#) sign, depending on whether the Device Manager server can access the Internet or whether the file has been copied to a local workstation.

After changing the value for a parameter, restart the Device Manager server for the change to take effect.

The following is an example of the `push.properties` file with the proper values configured for connection to the appointed WebSphere Everyplace Connection Manager server.

```
#-----Begin Copyright - do not add comments here-----
#
# 5724-B07, 5724-E69
# (C) Copyright IBM Corp. 2003, 2004. All Rights Reserved.
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with
# IBM Corp.
#
#-----End Copyright-----
#####
# Properties file for IBM Java Push API
#####
# The IBM Java Push API implementation supports the generation of
```

Working with Pervasive Devices

```
# logging output. Logging output can be sent to the console or a file
# or both. Presently, only two log levels are distinguished, ON and
# OFF.
#-----
# Switch to turn logging ON or OFF ["true", "false"]
push.logging=true

# Switch to turn console logging ON or OFF ["true", "false"]
push.logging.console=true

# Name of the log file [path]; logging to file is disabled if
# this property is missing
push.logging.filename=execution.log


# The IBM Java Push API implementation supports the generation of
# tracing output. Tracing output can be sent to the console or a file
# or both. Presently, only two trace levels are distinguished, ON and
# OFF.
#-----
# Switch to turn tracing ON or OFF ["true", "false"]
push.tracing=true

# Switch to turn console tracing ON or OFF ["true", "false"]
push.tracing.console=true

# Name of the trace file [path]; tracing to file is disabled if
# this property is missing
push.tracing.filename=trace.log


# The IBM Java Push API supports substitution of the URI for the PAP
# DTD. This may required in situations where the Push Initiator is
# behind a firewall.
#-----
# Examples of URIs of PAP DTD
# Loading from local http server
#   push.pap-dtd-uri=http://localhost/pap_1.0.dtd
# Loading from a file on a Windows platform
#   push.pap-dtd-uri=file:///c:\\PAP_DIR\\pap_1.0.dtd
# Loading from a file on a Unix platform
#   push.pap-dtd-uri=file:///etc/PAP_DIR/pap_1.0.dtd
#
push.pap-dtd-uri=http://www.wapforum.org/DTD/pap_1.0.dtd


# The IBM Java Push API supports national languages.
#-----
# Preferred language for message texts generated by the Push Proxy Gateway.
# Language tags conforming to RFC 1766
push.proxy-language=en


# Character encoding used for push messages of text media types (see
# RFC 2046).
# The IBM Java Push API currently only supports UTF-8 and US-ASCII
# encodings.
push.encoding=UTF-8


# Specifies the default domain for push addresses.
[subdomain.subdomain.subdomain]
push.ppg-specifier=pi.ibm.com


# The IBM Java Push API implementation entertains network connections
```



```

# to and from the Push Proxy Gateway. Interactions with the Push Proxy
# Gateway are via the HTTP protocol. For installations where a single
# Push Proxy Gateway is used, or in situations where a default Push
# Proxy Gateway exists, a URL for this gateway can be specified. For
# installations that support secure communication via SSL this can be
# enabled by specifying a "https" URL.
# NOTE: All of these parameters can also be set programmatically on the
# Pusher object and then override the properties settings.
#-----
# URL for the Push Proxy Gateway [URL]
# For example: push.proxy-url=http://wecmHostName.ibm.com:13131
push.proxy-url=http://WECM.IBM.COM:13131

# Secure communication via SSL requires a valid client certificate.
# This property specifies the mechanism and the required parameters to
# retrieve this certificate [mechanism:parameter]
push.ssl-context=CLASS:trusted.class;trusted

# Secure communication via SSL requires a valid client certificate.
# This property specifies the mechanism and the required parameters to
# retrieve this certificate [mechanism:parameter]
push.ssl-class=com.ibm.wireless.push.SSLightTransport

# The IBM Java Push API implementation supports forwarding of result
# notifications to multiple interested parties. It thus operates as a
# proxy for the respective HTTP POSTs. Result notifications are
# received on a particular port, which can be specified via this
# properties file.
#-----
# Port for receiving result notifications [port]
push.notification-port=1234
push.ssl-notification-port=4321

# Switch to indicate whether progrDocument Type Definitioness-notes in
# push responses are
# requested. ["true", "false"]
push.progress-notes-requested=false

# The IBM Java Push API implementation supports the specification of
# default quality-of-service parameters via this properties files.
# Requesting quality-of-service constraints is an optional feature of
# the WAP PAP protocol (see Wireless Application Protocol - Push Access
# Protocol Specification", WAP Forum, Ltd, 08-Nov-1999).
#-----
# Switch to turn quality-of-service specification ON or OFF
# ["true", "false"]
push.qos=false

# The priority ["low", "medium", "high"]
push.qos.priority=medium

# The delivery-method ["confirmed", "preferconfirmed", "unconfirmed",
# "notspecified"]
push.qos.delivery-method=notspecified

# The network ["Any", "GSM", "ANSI-136", "IS-95 CDMA", "AMPS",
# "PDC", "IDEN", "Paging Network", "PHS", "TETRA", "Mobitex"]
push.qos.network=Any

# Switch to indicate whether network is required or desired. ["true",
# "false"]
push.qos.network-required=false

# The bearer ["Any", "USSD", "SMS", "GUTS/R-Data", "CSD",
# "Packet Data", "GPRS", "CDPD", "FLEX", "SDS",

```

Working with Pervasive Devices

```
# "ReFLEX", "MPAK", "GHOST/R_DATA"]
push.qos.bearer=Any

# Switch to indicate whether network is required or desired. ["true",
# "false"]
push.qos.bearer-required=false
```

2. Create the device in Tivoli Resource Manager. You can create the new device using the Tivoli Resource Manager GUI or CLI. To perform this operation from the GUI, open the Tivoli desktop and navigate to the Create a Resource dialog box. The Nokia device requires some required information. To provide it, click the **Other Information** button in the Create a Resource dialog box. The Nokia Other Information window is displayed.



Figure 6. Nokia Other Information dialog

For more details on this procedure, see “Creating Devices” on page 357.

To perform this operation from the CLI, use the **wresource add** command. For example, to create a device called toronto with all the information contained in Figure 6, type the following command:

```
wresource add -i -o DEVICE_USERNAME="toronto" -o DEVICE_PASSWORD="devpswd"
-o SERVER_ID= "server_id" -o SERVER_PASSWORD="serverpswd"
-o DIALING_NUMBER="+1-919-555-4321" -o PROVISIONING_DOCUMENT=
"provisioning.xml"
-o USERPIN="pin" -o NETWPIN="pin" Pervasive_Device labelNokia myendpt
NokiaOMADM:IMEI:testNokia 4
```

For more information on this command, see “wresource” on page 394. The device is now registered in the Tivoli Resource Manager and Tivoli Web Gateway databases.

3. To enable the device to communicate with its appointed Tivoli Web Gateway, you must send a software package containing a provisioning file to the device. A provisioning document is a file in .xml format containing the key parameters for the initial connection to the Tivoli Web Gateway, such as the Tivoli Web Gateway address, the port number to be used for connections, and the information for establishing a connection with the carrier (Network Access Point). It also contains the same information inserted in the Other Information window in the Tivoli Resource Manager GUI or using the **-o** option with the **wresource add** command. For the connection to succeed, this information must match on the server and the device. The provisioning file must be created and edited by the user.

The following file is an example of a provisioning document:

```
<?xml version="1.0"?>
<!DOCTYPE wap-provisioningdoc PUBLIC "-//WAPFORUM//DTD PROV 1.0//EN"
"http://www.wapforum.org/DTD/prov.dtd">
<wap-provisioningdoc version="1.1">

  <characteristic type="NAPDEF">
    <parm name="NAPID" value="NAPid1"/>
    <parm name="BEARER" value="GSM-GPRS"/>
    <parm name="NAME" value="Carrier for DM"/>
    <parm name="NAP-ADDRESS" value="isp.cingular"/>
    <parm name="NAP-ADDRTYPE" value="IPV4"/>
    <parm name="DNS-ADDR" value="123.123.123.123"/>
    <parm name="DNS-ADDR" value="234.234.234.234"/>
    <characteristic type="NAPAUTHINFO">
      <parm name="AUTHTYPE" value="PAP"/>
      <parm name="AUTHNAME" value="XXXXXXXX@W5.MYCINGULAR.COM"/>
      <parm name="AUTHSECRET" value="XXXXXXXXXXXX"/>
    </characteristic>
  </characteristic>

  <characteristic type="APPLICATION">
    <parm name="APPID" value="w7"/>
    <parm name="PROVIDER-ID" value="ibm"/>
    <parm name="NAME" value="IBM DM Server"/>
    <parm name="ADDR" value="http://DMSSERVER.IBM.COM:80/dmserver
/SyncMLDMServlet"/>
    <parm name="TO-NAPID" value="NAPid1"/>
    <parm name="INIT" value=""/>
    <characteristic type="APPAUTH">
      <parm name="AAUTHTYPE" value="HTTP-DIGEST"/>
      <parm name="AAUTHNAME" value="name1"/>
      <parm name="AAUTHSECRET" value="secret1"/>
    </characteristic>
    <characteristic type="APPAUTH">
      <parm name="AAUTHLEVEL" value="APPSRV"/>
      <parm name="AAUTHTYPE" value="DIGEST,BASIC"/>
      <parm name="AAUTHNAME" value="servername"/>
      <parm name="AAUTHSECRET" value="serverpwd"/>
      <parm name="AAUTHDATA" value="servernonce"/>
    </characteristic>
    <characteristic type="APPAUTH">
      <parm name="AAUTHLEVEL" value="CLIENT"/>
      <parm name="AAUTHNAME" value="clientname"/>
      <parm name="AAUTHSECRET" value="clientpwd"/>
      <parm name="AAUTHDATA" value="clientnonce"/>
    </characteristic>
  </characteristic>
</wap-provisioningdoc>
```

The first section in the above file provides the Network Access Point (NAPDEF) information for a CSM/GPRS connection via a fictitious carrier called Carrier, including the connection type, the carrier, the address and the authorization properties.

The second section provides the OMA DM account information, including the provider, a link to an access point, as well as authorization properties, both on the server and client sides.

4. The administrator can now create a software package containing the provisioning document using the Software Package Editor, as described in *IBM Tivoli Configuration Manager: User's Guide for Software Distribution*.
5. The software package containing the provisioning file is sent to the resource group to which the device belongs in a Short Message Service (SMS). Check the WebSphere Application server SystemOut.log and the WebSphere Everyplace

Access Connection Manager logs to make sure that the SMS has been sent correctly. For more information on the WebSphere Application server logs, see “Tivoli Web Gateway Logs and Traces” on page 405.

6. The provisioning information contained in the SMS is saved locally and the device is configured automatically. Depending on the security level defined on the device, a confirmation might be requested before the action is performed on the device. The device can now connect to its appointed Tivoli Web Gateway.
7. The device connects to the Tivoli Web Gateway. This connection closes the circle, because the device is properly configured and the Tivoli Web Gateway is aware of its current status.
8. A new `DEVICES_NOT_BOOTSTRAPPED` query is added to the pervasive queries, to notify the administrator about the Nokia devices which have not been bootstrapped yet. This query starts working as soon as at least an Inventory scan on a device is performed on the Web server.

Note: If you need to modify the device configuration parameters after provisioning the device, you have to perform a device configuration operation. If you use the device configuration operation to modify security settings such as the server or client password, the change must be performed first on the device and then on the Device Manager server, otherwise the device will not be able to log in to the server. If the device profile is corrupted on the device, you need to delete the device and create it again in Tivoli Resource Manager to be able to perform the provisioning again.

You can now manage the device using Software Distribution to send packages containing the following information:

- Software applications to be installed
- Device configuration information to configure the applications you installed, or to modify existing settings

For more information on creating and sending this type of software package, refer to *IBM Tivoli Configuration Manager: User's Guide for Software Distribution*.

Working with Users

The following sections describe the tasks that you can perform on users.

Viewing Users

A user is a type of resource that is defined in the User database by applications, such as Lightweight Directory Access Protocol (LDAP). Because these users are managed by the applications that define them, you cannot create, modify, or delete users as you do other resources. However, you can list the users in the database and group them.

To list users, see “Listing and Finding Resources” on page 373. The User List Table dialog shows the users and the endpoints with which they are associated in the policy region.

For instructions about setting up the User database, see Part 6, “Managing an Enterprise Directory,” on page 415.

Grouping Resources

To work with resources after you enroll or create them, you must make them part of a resource group. You can group resources of the same type only, but you can group devices of different subtypes. For example, you can group Palm devices with Nokia devices, but you cannot group users and pervasive devices together. You can define either dynamic or static groups.

A dynamic resource group defines its current members at runtime. These members might change each time you distribute a profile to the group. You can use a dynamic resource group to make sure that the members always match the required criteria. For example, if you want to distribute a software package to a group of devices with specific hardware or software characteristics, you create a dynamic resource group based on a database query for the specified characteristics. The resource group is populated each time a distribution is submitted and the members are always up to date.

A static resource group defines the list of members when you create the group. That list varies only if you add or remove members from the group.

Both types of resource groups define their members using a policy or a query. You can use a default policy script or a query to filter the available members. For dynamic groups only, if you do not run a query or a policy, all resources are displayed.

The following table shows the results that you obtain using each type of filter for a dynamic or static group:

Table 55. Filtering Methods

Using the following:	For a static group	For a dynamic group
Default policy script	Filters available members	Shows current members
Validation policy	Filters current members	Shows current members
Query	Filters available members	Shows current members

Note: In case two interconnected regions contain two resource groups with the same name, any changes performed on one resource group are also applied to the other.

Using the Tivoli Policy Facility

You use the Tivoli Policy Facility to create resource groups. Policies are rules that help you manage resources. You can use the following policy methods to manage resource groups:

Default policy, `rg_def_subscribers`

This policy generates a default list of resources for the available members of a resource group. It is defined in the policy region and applies to all resource groups within the region. You can enable this policy from the Static Resource Group Members dialog, from the Dynamic Resource Group Current Members dialog, or using the **wresgrp** command line.

The validation policy, `rg_val_subscribers`

This policy ensures that the available members of a given resource group maintain acceptable values. Enable this policy if you want to validate the resources within a resource group for security reasons. It is defined in the

Grouping Resources

policy region and applies to all resource groups within the region. You can enable this policy when you manage resources in the policy region.

If you require different filtering behavior in the same policy region, you must provide a policy script with a case statement to handle each resource group instance.

Writing Scripts

Resource Manager provides required information (resource group, resource type, or resources) through the standard input and arguments within a script:

- The resource group name is defined using \$1 within a script.

Note: Do not include special characters in the resource group name.

- The resource type of the given group is defined using \$2 within a script.
- The list of resources (one on each line) to be filtered. These are defined in the standard input. The filtered resources are returned through the standard output in the corresponding resource line. It returns 0 to reject a resource and any other value to accept it.

Resource information is given within the standard input in the following structure:

```
Label|Manager|Local_Address|Flag
```

If the script terminates correctly, it finishes with exit 0; if not, it finishes with exit 1. The script can accept all resources and return a TRUE tag in the standard output. For example, the default policy script for rg_def_subscribers and rg_val_subscribers return TRUE as the first line of the standard output to accept all proposed resources.

See “Script for Default Policy” on page 402 for a sample script and refer to the *Tivoli Management Framework: User's Guide* for information on how to modify scripts defined for a given policy method.

Using the Tivoli Query Facility

To filter both the available and current members of a resource group, you can define queries in the Tivoli Query Library or use the LDAP query QueryDirectory for users. Refer to the *Tivoli Management Framework: User's Guide* for information on how to create these queries.

The table used to define your query must contain:

For the Tivoli Query Library

The ID column (containing device IDs) or the Label column (containing resources labels). The case of each column name depends on the database that you are using. If both columns are present, Resource Manager uses the ID column.

For the LDAP query

The tmeTrmId column (containing user IDs).

Creating Resource Groups

Grouping resources requires two steps:

1. creating a resource group
2. subscribing resources to it

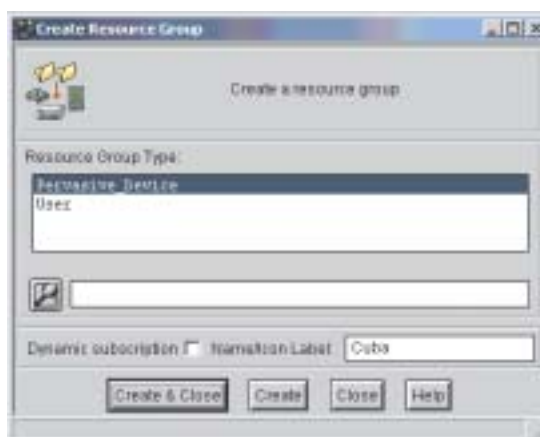
Before you start, ensure that you set the managed resources in the policy region to include **ResourceGroup**. If you want to validate the resources within a resource group for security reasons, enable the validation policy in the policy region.

Task	Context	Required Role
Creating resource groups	Tivoli desktop or CLI	senior
Subscribing resources to a resource group	Tivoli desktop or CLI	admin

For instructions on how to perform these tasks from the command line, see **wresgrp new** on page 265 and **wresgrp subscribe** on page 265, respectively.

To create a resource group, perform the following steps:

1. From the Tivoli desktop, double-click the policy region icon. The Policy Region dialog is displayed.
2. Select **Create -> ResourceGroup**. The Create Resource Group dialog is displayed.



3. In the **Resource Group Type** list, select:
 - Pervasive_Device** To group devices
 - User** To group users
4. Select the **Dynamic subscription** check box to create a dynamic group of resources.
5. In the **Name/Icon Label** text box, specify a name for your resource group. Valid characters: alphanumeric, underscore (_), dash (-), period (.), and blanks.
6. Click **Create & Close**.

Adding Resources to a Static Resource Group

To add resources to a static resource group, perform the following steps:

1. Double-click the static resource group icon. The Static Resource Group Members dialog is displayed.

Grouping Resources



2. To enable a query, select **Enable** and specify the query in the Name text box. Click **Expand** to list all the available queries.
Or:
To enable the default policy, select **Enable default policy rg_def_subscribers** check box.
3. Click **Update Available Members**. The **Available Members** list is updated to show only the filtered resources.
4. Select the members to be included in your resource group from the **Available Members** list and click the left arrow button. The selected members are moved to the **Current Members** list.
To remove a member from the list, double-click it.
5. To see validated current members, click **Apply & Close** and reopen the group. (If you have enabled the validation policy, the current members are also validated.)

Adding Dynamic Resource Groups

To add resources to a dynamic resource group, perform the following steps:

1. Double-click the dynamic resource group's icon. The Dynamic Resource Group Current Members dialog is displayed.



2. If necessary, click **Preview Current Members** to list the resources in the group.
3. To enable a query select **Enable** and specify the query in the Name text box. Click the Expand button to list all the available queries.
Or:
To enable the default policy, select **Enable default policy rg_def_subscribers** check box.
4. Click **Preview Current Members** to view your current resource group members. This list is updated to show only the filtered resources. If you have enabled the validation policy, the current members are also validated.

Listing and Finding Resources

To work with resources, you need to choose them from the list of resources. You can perform a search to find one or more resources in a list.

Task	Context	Required Role
Listing Resources	Tivoli desktop or CLI	admin

For instructions on how to perform this task from the command line, see **wresource ls** on page 394.

To list resources, perform the following steps:

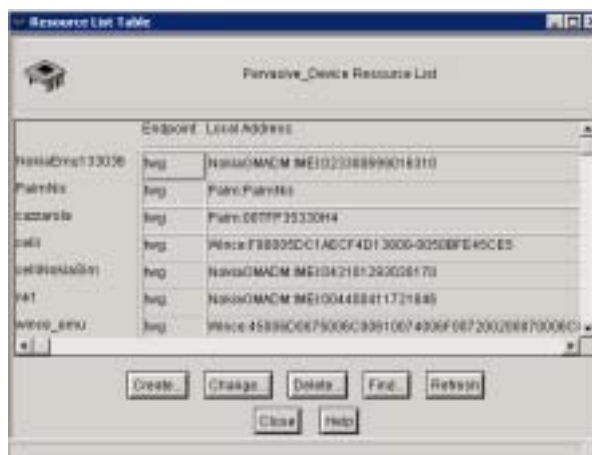
1. From the Tivoli desktop, double-click the **Resource Manager** icon. The Resource Type Table dialog is displayed.
2. Select the resource type:

Pervasive_Device
To list devices

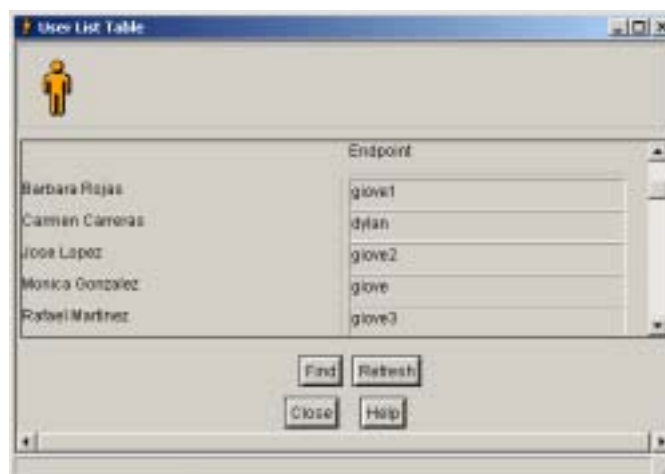
User To list users

Then, click **View**. The dialog that is displayed depends on the resource type listed.

If you specified Pervasive_Device, the Resource List Table dialog lists the resources defined in the policy region.



If you specified User, the User List Table dialog lists the users defined in the policy region. Note that in order for users to be listed here they must be included in the directory context called **directory**. See “Creating a Directory Query” on page 418 for instructions.



To find a particular resource, perform the following steps:

1. In the Resource List Table dialog, click **Find**. The Find Records dialog is displayed.
 - If you specified Pervasive Device, the following dialog is displayed:



- If you specified User, the following dialog is displayed:



2. Select an attribute in the **Attributes** list that can be used to find the resource you require.
3. Select a search condition and enter search text in the text box.
4. Click **Find All** to find all the resources in the list that match your search criteria.

Customizing Resource Manager Actions

You can customize the creating, deleting, or editing of resources by running a script automatically when you perform any of these three actions. See the description of the **wresource action** command in Chapter 19, “Using the Command Line,” on page 387 for the details.

Script for wresource action Command

The script run by the **wresource action** command receives information from either the standard input or from a file generated by Resource Manager.

The command can run in either RESOURCES or FILE mode: if the command is for one resource, it automatically runs in RESOURCES mode; if it is for two or more resources, it automatically runs in FILE mode. If you are doing a bulk operation in which you add one or more, the command runs in FILE mode. For normal operations, the command runs in RESOURCES mode.

In FILE mode, a file provides resource information. In RESOURCES mode, resource information is provided through the standard input by Resource Manager.

The arguments supplied by the file or standard input are:

- The *operation mode* (RESOURCES or FILE). This is the first parameter of the script (defined as \$1 within a script).
- The *type of resource* (Pervasive_Device). This is the second parameter of the script (defined as \$2 within a script).
- The *file path* that contains the resource information. This is the third parameter of the script (defined as \$3 within a script). It is defined only if the operation mode is FILE.
- The *managed resource information* through the standard input if mode is RESOURCES; a file if mode is FILE.
 - If you are editing resources, the first line contains the old settings of the resource and the second line contains the new settings.
 - If you are creating or removing resources, the first line contains all the resource information.

Customizing Resource Manager Actions

Information about resources is returned in the following structure:

Label|Endpoint|Local_Address|Flag

To end the script normally, finish with exit 0; otherwise, finish with exit 1.

Note: Do not use special characters within a script.

For an example of a script, see “Script for Use with wresource action Command” on page 401.

Chapter 18. Device Customization Parameters

You can customize devices in either of the following ways:

- Using the graphical interface of the Software Package Editor and generating a software package for devices. (See the "Creating a Software Package for Devices" chapter in the *User's Guide for Software Distribution* for details.)
- Creating a software package definition (SPD) file. (See the "Editing the Software Package Definition (SPD) File" chapter in the *Reference Manual for Software Distribution* for details.)

Either way, you include the resulting file in a profile and distribute the profile to resource groups containing the target devices. This chapter lists the parameters that can be customized on devices and provides the keywords and valid values that you can specify in the SPD file.

Creating a Customization File for Palm OS Devices

The following sections describe the customization keywords for Palm devices.

Format Parameters

The following table lists the keywords that customize time and date formats.

Creating a Customization File for Palm OS Devices

<i>Format Keywords</i>		
Configuration Keyword	Description	Valid Values
Preset	<p>Preset country or region (integer)</p> <p>Note: Changing the configuration value for Preset has no effect on the values for the other Format configuration keywords.</p> <p>However, changing the Preset value on the Palm device Preferences window <i>does change</i> the values for the other Format configuration keywords (time, date, week start day, and number) to the default values for that country.</p>	23 (for the United States). Refer to “Preset Countries or Regions and Stored Values” for a list.
TimeFormat	Time format (integer)	1 (for 2:00 pm format). Refer to “Time Formats and Stored Values” on page 379 for a list.
DateFormat	Date format (integer)	0 (for 12/31/99 format). Refer to “Date Formats and Stored Values” on page 379 for a list.
LongDateFormat	Long date format (integer)	7 (for December 31, 1995 format). Refer to “Long Date Formats and Stored Values” on page 379 for a list.
WeekStartDay	<p>Week start day (integer)</p> <p>Range 0 through 6</p>	<p>0=Sunday</p> <p>1=Monday</p> <p>.....</p> <p>6=Saturday</p>
NumberFormat	Number format (integer)	0 (for Comma Period format). Refer to “Number Formats and Stored Values” on page 379 for a list.

Preset Countries or Regions and Stored Values

Australia	= 0
Austria	= 1
Belgium	= 2
Brazil	= 3
Canada	= 4
China (Hong Kong S.A.R.)	= 9
Denmark	= 5
Finland	= 6
France	= 7
Germany	= 8
Iceland	= 10
Ireland	= 11
Italy	= 12
Japan	= 13
Luxembourg	= 14

Mexico	= 15
Netherlands	= 16
New Zealand	= 17
Norway	= 18
Spain	= 19
Sweden	= 20
Switzerland	= 21
United Kingdom	= 22
United States	= 23

Note: Changing the configuration value for **Preset** has no effect on the values for the other **Format** configuration keywords. However, changing the **Preset** value on the Palm device Preferences window *does change* the values for the other **Format** configuration keywords (time, date, week start day, and number) to the default values for that country or region.

Time Formats and Stored Values

1:00	HH:MM	= 0
1:00 pm	HH:MM am/pm	= 1
13:00	24HH:MM	= 2
1.00	HH.MM	= 3
1.00 pm	HH.MM am/pm	= 4
13.00	24HH.MM	= 5
1 pm	H am/pm	= 6
13	24H	= 7
13,00	24HH,MM	= 8

Date Formats and Stored Values

12/31/95	M/D/Y	= 0
31/12/95	D/M/Y	= 1
31.12.95	D.M.Y	= 2
31-12-95	D-M-Y	= 3
95/12/31	Y/M/D	= 4
95.12.31	Y.M.D	= 5
95-12-31	Y-M-D	= 6
Dec 31, 1995	MDY Long Year w/ comma	= 7
31 Dec 1995	DMY Long Year no comma	= 8
31. Dec 1995	DMY Long Year w/ dot	= 9
Dec 1995	MY Long Year no Day	= 10
31 Dec, 1995	DMY Long Year w/ comma	= 11
1995.12.31	YMD Long Year w/ Dot	= 12
1995 Dec 31	YMD Long Year w/ Space	= 13
Dec '95	MY Med Year w/ '	= 14
Dec 95	MY No Post	= 15

Long Date Formats and Stored Values

Dec 31, 1995	MDY Long Year w/ comma	= 7
31 Dec 1995	DMY Long Year no comma	= 8
31. Dec 1995	DMY Long Year w/ dot	= 9
Dec 1995	MY Long Year no Day	= 10
31 Dec, 1995	DMY Long Year w/ comma	= 11
1995.12.31	YMD Long Year w/ Dot	= 12
1995 Dec 31	YMD Long Year w/ Space	= 13

Number Formats and Stored Values

CommaPeriod	= 0
PeriodComma	= 1
SpaceComma	= 2
ApostrophePeriod	= 3
ApostropheComma	= 4

General Parameters

The following table lists the keywords that customize general and sound settings.

<i>General Keywords</i>		
Configuration Keyword	Description	Valid Values
SetDateTime	Set the system date and time. Use the keyword CURRENT or a string value. The string value format for the timestamp is: yyyy/MM/dd HH:mm:ss (Use 24 hour notation for HH)	CURRENT (uses the server system date and time - default) String values for date and time 2002/07/04 14:05:25 (July 4, 2002 at 2:05:25 pm)
AutoOffTimer	The device has an automatic shutdown feature that turns the power off after a period of inactivity. This feature conserves the battery power.	30 seconds (30) One minute (60) Two minutes (120) Three minutes (180) Never (0)
SystemSound	System sound setting (integer)	No sound (0) Low (8) Medium (32) High (64)
AlarmSound	Alarm sound setting (integer)	No sound (0) Low (8) Medium (32) High (64)
GameSound	Game sound setting (integer)	No sound (0) Low (8) Medium (32) High (64)

Network Parameter

The following table lists the keyword that customizes network settings.

<i>Network Keywords</i>		
Configuration Keyword	Description	Valid Values
UserName	Subscriber name for PPP (See Note below)	String value

Note: The Network **UserName** keyword enables you to use the TCP/IP software that is included with the Palm OS device, such as a PPP connection or a remote access server (RAS). This configuration keyword is separate from the Device Manager specific Agent **PalmUserID** keyword.

To use TCP/IP, you must configure both the Modem and Network preferences.

TCP/IP Parameters

The following table lists the keywords that customize TCP/IP settings.

<i>TCP/IP Keywords</i>		
Configuration Keyword	Description	Valid Values
DNSQuery	PPP queries for DNS address (integer)	1=On (checkmark displayed) 0=Off
PrimaryDNS	Primary DNS IP address	String value
SecondaryDNS	Secondary DNS IP address	String value

Modem Parameter

The following table lists the keywords that customize the modem number.

<i>Modem Keyword</i>		
Configuration Keyword	Description	Valid Values
ModemPhone	Modem telephone number	String value

Note: The Network **UserName** keyword enables you to use the TCP/IP software that is included with the Palm OS device, such as a PPP connection or a remote access server (RAS). To use TCP/IP, you must configure both the Modem and Network preferences.

Agent Parameters

The following table lists the keywords that customize settings for the Resource Manager agent that runs on the device.

<i>Agent Keywords</i>		
Configuration Keyword	Description	Valid Values
SSLOn	SSL security	0=Off 1=On
PalmUserID	The subscriber's user ID (See Note below)	String value
DMSAddress	Resource gateway address or host name. If you are using IBM Tivoli Access Manager WebSEAL, type the host name of the WebSEAL server.	String value.
DMSPort	Resource gateway port number (integer). If you are using WebSEAL, type the number of the port used by the WebSEAL server.	The default value is 80.

Creating a Customization File for Palm OS Devices

PalmServletName	Palm servlet name (string value)	/dmserver/PalmServlet (See Note 4 below).
ServiceName	Network interface name to use for Palm OS agent program	String value. For example: DevAgent
BufferSize	Device agent receive buffer size in KB (integer)	Range 4 through 24

Notes:

1. The UserName parameter under Network Parameters enables you to use the TCP/IP software that is included with the Palm OS device, such as a PPP connection or a remote access server (RAS).
The UserName parameter is separate from the Device Manager specific Agent parameter called PalmUserID.
2. The PalmUserID keyword is a unique identifier for the device and is stored in the Device Manager database. Normally, the ROM serial number is used as the value for PalmUserID. However, if the device's ROM does not have a serial number, then the user's name, as entered on the device, serves as the unique identifier.
3. If the ROM of the device does not have a serial number and the value for the Palm user ID is changed, then to Resource Manager, it appears to be a "new" device which must be enrolled again. New records in the Resource Manager database are created for this device and these new records are not linked to the previous data for the device.
4. If you are using IBM Tivoli Access Manager WebSEAL, type the following for the **PalmServlet Name** field:

/junction_name/dmserver/PalmServlet

where:

junction_name

Is the name of the junction that has been configured in WebSEAL to redirect the communication to WebSphere.

Proxy Parameters

The following table lists the keywords that customize the proxy server settings.

<i>Proxy Keywords</i>		
Configuration Keyword	Description	Valid Values
Pyroxenes	Enable proxy server (integer)	0=Off 1=On (checkmark displayed)
Proxy Address	Proxy server address	www.proxy.com
Participator	Proxy server port (integer)	80

Sample File for Windows CE Devices

The following sections describe the customization keywords for Windows CE devices.

PPP Parameters

The following table lists the keywords that customize point-to-point protocol (PPP) settings.

<i>PPP Keywords</i>		
Configuration Keyword	Description	Valid Values
ppp.dial	PPP telephone number to dial into (or access) the service provider	String value
ppp.user	PPP user ID	String value

TCP/IP Parameters

The following table lists the keywords that customize TCP/IP addresses.

<i>TCP/IP Keywords</i>		
Configuration Keyword	Description	Valid Values
net.dns1	Primary DNS	String value. For example: 9.4.3.100
net.dns2	Secondary DNS	String value. For example: 9.4.3.101

Browser Parameters

The following table lists the keywords that customize browser settings.

<i>Browser Keywords</i>		
Configuration Keyword	Description	Valid Values
bsr.startpage	Start page URL	String value. http://www.tivoli.com
bsr.proxyaddr	Proxy server IP address - See Note below.	String value proxy.tivoli.com
bsr.proxyport	Proxy server port number (integer) - See Note below.	80
pct.enable	PCT setting for browser On pocket PC devices, the Internet Explorer browser does not support the PCT option.	String value ON (checkmark displayed) OFF
ssl2.enable	SSL2 setting for browser There is no corresponding Pocket PC parameter. You cannot view the value with the device.	String value ON (checkmark displayed) OFF
ssl3.enable	SSL3 setting for browser There is no corresponding Pocket PC parameter. You cannot view the value with the device.	String value ON (checkmark displayed) OFF

Notes:

1. The settings for **mgmt.proxy.addr** and **mgmt.proxy.port** do not apply to the Pocket Internet Explorer browser or any other Web browser. The proxy settings of the Pocket Internet Explorer browser are independent from these settings.
2. Some browsers no longer support the PCT option. In this situation, the value for the **pct.enable** keyword is ignored.
3. On Pocket PC devices, the Internet Explorer browser does not support all features, so all Browser keywords are not used. In this situation, the Browser keywords should be commented out in the configuration template file, such as the **pct.enable** keyword.

Mailer Parameters

The following table lists the keywords that customize settings for e-mail. The mailer parameters are only available for Windows CE devices and Pocket PC devices.

<i>Mailer Keywords</i>		
Configuration Keyword	Description	Valid Values
pop3.server	POP3 server address or server name The e-mail server you use to send and receive messages.	String value. For example: pop3svr.tivoli.com
smtp.server	SMTP server address or server name If your e-mail service uses a separate server for SMTP, enter the address in this field.	String value. For example: smtpsvr.tivoli.com
pop3.uid	POP3 user ID or e-mail ID	String value
mail.address	Mail account for return e-mail address. By default, the return e-mail address is set to: pop3.uid@pop3.server If this is not your e-mail return address, enter the correct address in this field.	String value

Note: The mailer parameters are not available for Pocket PC 2002 devices.

Management Parameters

The following table lists the keywords that customize management settings.

<i>Mgmt Keywords</i>		
Configuration Keyword	Description	Valid Values
mgmt.serveraddr	Resource gateway URL	String value. For example: http://9.3.4.1/mgsrvlt
mgmt.pollingtimer	Polling timer in hours (integer) There is no corresponding Pocket PC parameter. You cannot view the value with the device.	Range: 1 to 65535 For example:10

Sample File for Windows CE Devices

mgmt.agentruntime	Determines if the device agent program automatically connects to the server and polls for waiting jobs. ON : Agent program automatically starts and polls for waiting jobs. Hides the Device Agent window at startup. OFF: Displays the Device Agent window at startup. To connect to the server, click Connect . See the Poll automatically check box on the General tab for the agent setup information. There is no corresponding Pocket PC parameter. You cannot view the value with the device.	String value ON (checkmark displayed) OFF
mgmt.proxy.enable	Enable proxy server	ON (checkmark displayed) OFF
mgmt.proxy.addr	Proxy server address - See Note below.	www.proxy.com
mgmt.proxy.port	Proxy server port (integer) - See Note below.	80
mgmt.auth.user	The user ID	String value. For example: Chris

Note: The settings for **mgmt.proxy.addr** and **mgmt.proxy.port** do not apply to the Pocket Internet Explorer browser or any other Web browser. The proxy settings of the Pocket Internet Explorer browser are independent from these settings.

Chapter 19. Using the Command Line

This chapter describes how to work with Resource Manager using the command line interface (CLI).

For information about syntax conventions and getting help for the command line, see “Using the Command Line” on page xix.

Table 56. Commands for working with Resource Manager

Command	Purpose	Details on page
wresgrp	Enables you to work with individual and groups of resources	388
wresgw	Enables you to work with the resource gateway	391
wresource	Enables you to work with individual resources	394

wresgrp

Works with resource groups.

Syntax

```
wresgrp ls [group_name... ]
```

```
wresgrp new resource_type group_name policy_region [-d] [-q -t query_type -n query_name | -p]
```

```
wresgrp set group_name {-q | -t query_type -n query_name | -p | -d}
```

```
wresgrp subscribe group_name {resource_label... }
```

```
wresgrp unsubscribe group_name {resource_label... }
```

Description

The **wresgrp** command is run at the Tivoli management region server or at a managed node. It enables you to create resource groups, define their members, list the groups and their members.

Options

The following lists the subcommands of the **wresgrp** command:

- ls** Lists the resource groups. If you specify the group name, it lists the group members
- new** Creates a static or dynamic group of resources with the group name and in the policy region that you specify. You can use the flags **-q -t -n** to enable a query to create the group
- set** Enables a query or default policy to be run on the specified group. Also, modifies the properties of the resource group
- subscribe**
Adds one or more resources to a static group
- unsubscribe**
Removes one or more resources from a static group

The following lists the options and variables belonging to the subcommands:

- d** For **wresgrp set**, the options stands for: Disables the policy and query
For **wresgrp new**, the options stands for: Dynamic, the resource group is created dynamically, using a query
- group_name*
Specifies the name of the group

Note: Valid characters are alphanumeric, underscore (_), dash (-), period (.), and blanks
- n query_name**
Specifies the name of the query used to create the group. Use this option with the **-t query_type** option.
- p** Enables the rg_def_subscribers default policy. For more information on this policy, see “Using the Tivoli Policy Facility” on page 369

-q Enables the query

resource_label

Specifies the label of the resource to be added using the **wresgrp subscribe** command or removed using the **wresgrp unsubscribe** command

resource_type

Specifies the resource type. The valid types are: Pervasive_Device or User

policy_region

Specifies the policy region where the new resource is created

-t query_type

Specifies the type of query that creates the group. For the Tivoli Management Framework, enter TMF_Query. For LDAP, enter QueryDirectory. Use this option with the **-n query_name** option.

Authorization

The **admin** role is required for all **wresgrp** commands, except **resgrp new**, which requires the **senior** role.

Examples

Create

To create a static group called **Puerto Rico** in the **Chispito-LA** policy region, enter the following:

```
wresgrp new Pervasive_Device "Puerto Rico" Chispito-LA
```

To create a dynamic resource group called **Peru** in the **Chispito-LA** policy region, use the following command:

```
wresgrp new Pervasive_Device Peru Chispito-LA -d
```

List

To list existing resource groups:

```
wresgrp ls
```

To list the resources that belong to the group called **Puerto Rico**, enter the following:

```
wresgrp ls "Puerto Rico"
```

Set

To enable the default rg_def_subscribers policy on a static or dynamic resource group named **Venezuela**, enter the following:

```
wresgrp set Venezuela -p
```

To set and enable a Tivoli Management Framework query named **Palmdevices** on a static or dynamic resource group named **Guatemala**, enter the following:

```
wresgrp set Guatemala -q -t TMF_Query -n Palmdevices
```

Subscribe

To add two Windows CE devices called **Felix Gonzalez** and **Rosita Vargas** to the static resource group called **Puerto Rico**, enter the following:

```
wresgrp subscribe "Puerto Rico" "Felix Gonzalez" "Rosita Vargas"
```

wresgrp

Unsubscribe

To remove a device called **Felix Gonzalez** from the resource group called **Puerto Rico**:

```
wresgrp unsubscribe "Puerto Rico" "Felix Gonzalez"
```

wresgw

Enrolls and discovers resources and enables their management.

Syntax

```
wresgw add endpoint -C resource_gateway_type...
```

```
wresgw autoenroll { enable | disable } [-C resource_gateway_type] endpoint...
```

```
wresgw discover [-v] [-C resource_gateway_type] [-f] [-a]endpoint...
```

```
wresgw ls [endpoint]
```

```
wresgw remove endpoint [-C resource_gateway_type...]
```

```
wresgw update endpoint [new_endpoint]
```

```
wresgw view_config [-C resource_gateway_type] endpoint...
```

Description

The **wresgw** command is run at the Tivoli management region server or at a managed node. It enables you to enroll, discover, list, and remove, and configure resource gateway options.

Options

The following lists the subcommands of the **wresgw** command:

add Associates one or more resource gateways with the Tivoli server

autoenroll

Enables auto-enrollment of resources on the resource gateway of the type or on the endpoints specified

discover

Discovers resources associated with target resource gateways of the type or on the endpoints specified

ls

Lists all known endpoints on which a resource gateway is installed. It also lists the status of each endpoint if it no longer exists or if the object ID has changed

If you include the *endpoint* argument, it lists all known resource gateway types on the endpoint and the object ID of the endpoint

remove

Removes the registration for one or more resource gateways with the Tivoli server

update

Updates the object ID and/or the label of an endpoint that has been registered with the resource gateway

view_config

Retrieves resource gateway settings from the type or the endpoint you specify. This command also identifies the resource gateways that can receive autoenrollment reports.

The following lists the options and variables belonging to the subcommands:

-C resource_gateway_type

Specifies the resource gateway type. Currently, the only supported type is TWG

disable

Disables resource from being enrolled automatically

enable

Enables resources to be enrolled automatically

endpoint

Specifies the endpoint on which the resource gateway is installed.

For the **wresgw ls** command, lists all known resource gateway types on the endpoint that you specify.

For the **wresgw update** command, indicates the endpoint for which the object ID or endpoint label is being updated. This option is mandatory.

new_endpoint

Indicates the new label for the endpoint being updated

-v

Displays further information on discovered devices. The following information is returned:

- endpoint object ID
- device local address

-f

Discovers all devices on the specified endpoint. If you do not specify this option, the discovery operation returns only devices added since the last discovery operation.

-a

Discovers devices asynchronously. The results of the operation are saved to the discover.log file located in the /work directory. This operation is provided with a distribution ID and you can view its status with the **wmdist** command. For more information on this command, refer to *Tivoli Management Framework Reference Manual*.

Authorization

The **admin** role is required for all **wresgw** commands.

Examples

add

To associate a resource gateway, of the type Web Gateway, with the endpoint rvargas, enter the following:

```
wresgw add rvargas -C TWG
```

autoenroll

To enable auto-enrollment of resources on the endpoint jlopez, enter the following:

```
wresgw autoenroll enable jlopez
```

discover

To discover resources associated with the endpoint jlopez and register them in the Tivoli Resource Manager (TRM) database, enter the following:

```
wresgw discover -C TWG jlopez
```

list

To list all known resource gateway types on the endpoint rmartin and to see the object ID of rmartin, enter the following:

```
wresgw ls rmartin
```

remove

To remove a resource gateway, of the type Web Gateway and associated with the endpoint fgonzal, from the Tivoli server, enter the following:

```
wresgw remove fgonzal -C TWG
```

update

To update the label and object ID of an endpoint that is registered with the resource gateway from rmartin to brojas, enter the following:

```
wresgw update rmartin brojas
```

view_config

To retrieve the settings of a resource gateway, associated with the endpoint fgonzal, enter the following:

```
wresgw view_config fgonzal
```

wresource

Lists resources, their types, and their details. Manages the Resource Manager database.

Syntax

```
wresource action resource_type {-e | -d | -r | -v} {add_resource | remove_resource | edit_resource}
```

```
wresource add [-i] [-f] [-u] {-F filename resource_type | [-o key=value]...  
{{resource_type resource_label resource_manager local_address} device_subtype } ...}
```

```
wresource edit resource_type resource_label [-u] [-l new_resource_label] [-m  
new_resource_manager] [-a new_local_address] [-o key=value]...
```

```
wresource ls [-o] [resource_type [resource_label ...]]
```

```
wresource remove {-a resource_type | -F filename resource_type} | {{resource_type  
resource_label}... }
```

Description

The **wresource** command is run at the Tivoli management region server or at a managed node. It enables you to define, edit, and remove resources.

Options

The following lists the subcommands of the **wresource** command:

- action** Defines actions performed using a script when an add, remove, or edit is carried out on the resource
- add** Defines one or more resources in the Tivoli environment and defines the related settings. To create multiple resources, use a script, a list, or a file containing a list of resources
- edit** Edits a resource. Using this option, you can change the resource label and the following information for Nokia devices:
 - DEVICE_USERNAME
 - DEVICE_PASSWORD
 - SERVER_ID
 - SERVER_PASSWORD
 - DIALING_NUMBER
 - USERPIN
 - NETWPIN
 - PROVISIONING_DOCUMENT

To change information for the Nokia device, use the **-o key=value** option, as described below.
- ls** Lists resources, their types, and their details
- remove** Removes one or more resources from the Tivoli environment

The following lists the options and variables belonging to the subcommands:

-a All, removes all resources

add_resource

Specifies the add action defined by **wresource action**

-d Disables the action defined by **wresource action**

device_subtype

Required when you add pervasive devices using the **-i** option. Specifies subtypes for the resource type Pervasive_Devices. Valid subtypes are:

For Palm OS devices

Specify: **Palm** or **1**

For Windows CE devices

Specify: **WinCE** or **2**

For Nokia 9500 Communicator series devices

Specify: **Nokia9500** or **4**

For Nokia 9300 Communicator series devices

Specify: **Nokia9300** or **5**

For Nokia s60 Communicator series devices

Specify: **Nokia_s60** or **6**

-e Enables the action defined by **wresource action**

edit_resource

Specifies the edit action defined by **wresource action**

-f Force, provides a new label for the resource if the label already exists in the resource manager

-F File, adds or removes the resources listed in the file that you specify

filename

Specifies the file in which the resources to be defined or removed are listed. These must be listed in the following format:

resource_label resource_manager local_address device_subtype

You can also define additional information for Nokia devices using the format: **key=value;key=value;.**

Note: The key values cannot be set unless you have set the NOTIFICATION_PROTOCOL and DIALING_NUMBER keys using the following command

```
wresource edit Pervasive_Device Administrator -o NOTIFICATION_
PROTOCOL=WAPPushv12 -o DIALING_NUMBER=+<123456>
```

The two keys must be set at the same time (in the same command).

The following keys are supported:

DEVICE_USERNAME

Specifies the client name defined in the OMA DM protocol for authenticating the device to the Tivoli Web Gateway. The maximum length for the user name is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

DEVICE_PASSWORD

Specifies the password that the device uses to authenticate itself to the Tivoli Web Gateway. The maximum length for the device password is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

SERVER_ID

Identifies the Tivoli Web Gateway. The maximum length for the server ID is 255 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

SERVER_PASSWORD

Specifies the password for the server defined in the OMA DM protocol. The password must be unique for each device. The maximum length for the password is 60 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). This field is required.

DIALING_NUMBER

Specifies the telephone number of the mobile device in international format. Type a plus sign (+) followed by numbers with intervening hyphens (-) and periods (.) as necessary (for example: +41-1-123.1234 or +1-919-555-4321). This field is required.

NETWPIN

Specifies security information for sending provisioning data. This information guarantees a secure connection to the Tivoli Web Gateway. The maximum length for the NETWPIN data is 40 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). You must fill in at least the NETWPIN or USERPIN fields.

Note: Ensure that you do not remove the information specified in at least one of the two NETWPIN and USERPIN fields.

USERPIN

Specifies additional security information for sending provisioning data. This information guarantees a secure connection to the Tivoli Web Gateway. The maximum length for the USERPIN data is 40 characters. Blanks and special characters are permitted, except for asterisks (*) and semicolons (;). You must fill in at least the NETWPIN or USERPIN fields.

Note: Ensure that you do not remove the information specified in at least one of the two NETWPIN and USERPIN fields.

PROVISIONING_DOCUMENT

Specifies the name of a .xml file used for the initial provisioning of the device. This field is required.

NOTIFICATION_PROTOCOL

Displays the notification method assigned to the device.
The notification protocol supported for this version is WAP Pushv12. This field cannot be edited.

Notes:

1. List the *device_subtype* only if the *resource_type* is Pervasive_Device.
 2. If you use a file (-F argument) to remove devices, the only required argument is *resource_label*. All other arguments listed above are optional.
- i Required when you add single or multiple pervasive devices from the command line only. Do not use this argument when you add pervasive devices in a bulk operation by listing them in a file
- l Label, specifies a new name for a resource

local_address

The local address can have a maximum of 180 characters. If you use Inventory queries, the local address must have a maximum of 64 characters. It must contain no blanks, asterisks (*) or semicolons (;). Use the format:

device_class:device_id

where:

device_class

NokiaOMADM:IMEI (for Nokia devices), Palm, or Wince

device_id

Indicates the IMEI number of Nokia Communicator series devices or the serial number of Palm devices

For Windows CE devices: The local address can either be generated automatically by the resource gateway or user-defined. When you create Windows CE devices from the desktop or the command line, use the following format:

Wince:user_name:device_name

where:

user_name

Is the user ID of the device, and is required

device_name

Is optional and must be separated from the *user_name* by a colon (:))

If you define the *device_name*, the final local address will be *Wince:devicename*. Otherwise, the resource gateway generates a local address in the format:

Wince:auto-generated address

This is the address that appears when you list the device after it is created

new_resource_label

Specifies the new resource label

new_resource_manager

Specifies the new endpoint to which the resource connects

-o Specifies or edits more information available for the Nokia Communicator device. The valid options when creating a new resource are as follows:

- DEVICE_USERNAME
- DEVICE_PASSWORD
- SERVER_ID
- SERVER_PASSWORD
- DIALING_NUMBER
- PROVISIONING_DOCUMENT
- USERPIN
- NETWPIN

The valid options when editing an existing resource are as follows:

- DEVICE_USERNAME
- DEVICE_PASSWORD
- SERVER_ID
- SERVER_PASSWORD
- DIALING_NUMBER
- PROVISIONING_DOCUMENT
- USERPIN
- NETWPIN

For more information on these options, see on page 395.

-r For **wresource action**, the option stands for: registers the action defined by **wresource action**

remove_resource

Specifies the remove action defined by **wresource action**

resource_label

Specifies the resource label

resource_manager

Specifies the endpoint to which the resource is connected

resource_type

Specifies the resource type. The valid types are: Pervasive_Device or User

-u Unique, forces the manager-local-address pairing to be unique

-v Verbose, displays the action settings for you to view

Authorization

The **admin** role is required for all **wresource** commands.

Examples

Action

To register the script MY_SCRIPT.SH and cause it to run every time you add a resource, enter the following:

```
wresource action Pervasive_Device -r add_resource < MY_SCRIPT.SH
wresource action Pervasive_Device -e add_resource
```

To automatically run a script when you modify resources, enter the following:

```
wresource action Pervasive_Device -r edit_resource < MY_SCRIPT.SH
```

```
wresource action Pervasive_Device -e edit_resource
```

To automatically run a script when you delete resources:

```
wresource action Pervasive_Device -r remove_resource < MY_SCRIPT.SH
wresource action Pervasive_Device -e remove_resource
```

Create

To create a Palm device and a Windows CE device from the CLI by listing them, enter the following:

```
wresource add -i Pervasive_Device "Jose Lopez" giove2 50C015F94799 1\
Pervasive_Device "Rafael Martinez" giove3 rmartin@IBM 2
```

To create a Nokia 9500 device from the CLI, enter the following:

```
wresource add -i -o DEVICE_USERNAME="username" -o DEVICE_PASSWORD="devpswd"
-o SERVER_ID= "server_id" -o SERVER_PASSWORD="serverpswd"
-o DIALING_NUMBER="+390287743" -o PROVISIONING_DOCUMENT="provisioningfile"
-o USERPIN="pin" -o NETWPIN="pin" Pervasive_Device labelNokia myendpt
NokiaOMADM:IMEI:testNokia 4
```

To pass a file called ChileDevices, which lists the resources to be created, enter the following command:

```
wresource add -F ChileDevices Pervasive_Device
```

Edit

To modify the label of resource **Carmen Carreras** to **Manuel Chavez**, enter the following:

```
wresource edit Pervasive_Device "Carmen Carreras" -l "Manuel Chavez"
```

List

To list pervasive devices, enter the following:

```
wresource ls Pervasive_Device
```

To create Windows CE devices in a bulk operation using the file GuatemalaDevices, enter the following:

```
wresource add -F GuatemalaDevices Pervasive_Device
```

Remove

To remove the resource **Carmen Carreras**, enter the following:

```
wresource remove Pervasive_Device "Carmen Carreras"
```

Chapter 20. Sample Scripts

This appendix provides sample scripts. You can use them as a basis to create your own.

Script for Use with wresource action Command

The following schema offers a basis for a script that can be used by the **wresource action** command. It shows how to manage input from the Resource Manager concerning actions performed on targeted resources. This script shows an example of a Nokia 9500 device. For more information about the command, see Chapter 19, "Using the Command Line," on page 387.

```
#!/bin/sh
# Resource Manager Action Feature Sample Script
DEMO_OUT_FILE="/DEMO_OUT_FILE"
echo "Resource Manager Action Feature Sample Script output" > $DEMO_OUT_FILE

# Receives the operation mode through $1
OPERATION_MODE=$1
echo "Operation mode: "$OPERATION_MODE >> $DEMO_OUT_FILE
# Receives the resource type through $2
RESGRP_TYPE=$2
echo "Resource Type: "$RESGRP_TYPE >> $DEMO_OUT_FILE
if [ "$OPERATION_MODE" = "RESOURCES" ]
then
    # Receives Nokia 9500 information through the stdin
    echo "Input from stdin" >> $DEMO_OUT_FILE
    echo "Nokia 9500 Information Received" >> $DEMO_OUT_FILE
    while read line
    do
        echo $line >> $DEMO_OUT_FILE
        #You can parse the Nokia 9500 information using the awk
        #echo $line | awk -F'|' '{print $1; print $2; print $3; print $4;}'
    done
else
    # Receives Nokia 9500 information through a file
    echo "Input from file" >> $DEMO_OUT_FILE
    # Mode is FILE, $3 contains the path of the input file
    INPUT_FILE=$3
    echo "Input file: "$INPUT_FILE >> $DEMO_OUT_FILE
    echo "Nokia 9500 Information Received" >> $DEMO_OUT_FILE
    cat < $INPUT_FILE >> $DEMO_OUT_FILE
fi

# Script terminates correctly
exit 0
# exit 1 if there is an error
```

Script for Default Policy

The following script defines subscribers for existing groups of Palm devices, but you can customize it to create any subtype of pervasive devices. It can be used as the default policy or the validation policy for both static and dynamic resource groups. It is used in conjunction with a script like that shown in “Script to Create Resource Group and Activate Default Policy” on page 403.

Note that in the `if [$RESOURCE_TYPE -eq 1];then` statement, the resource type is defined as 1. This refers to the subtype of `Pervasive_Device`. Subtype values for the script are as follows:

- 1 Palm devices
- 2 Windows CE devices
- 4 Nokia 9500 devices
- 5 Nokia 9300 devices
- 6 Nokia s60 devices

```
#!/bin/sh
# Resource Group Policy Feature Sample Script
# This script writes the received information in the file '/DG_OUT'
OUT_FILE="/DG_OUT"
echo > $OUT_FILE

# The script receives the resource group name through $1
RESGRP_NAME=$1
echo "Resource Group Name: "$RESGRP_NAME >> $OUT_FILE

# The script receives the resource type through $2
RESGRP_TYPE=$2
echo "Resource Type: "$RESGRP_TYPE >> $OUT_FILE

if [ "$RESGRP_TYPE" = "Pervasive_Device" ];then

    if [ "$RESGRP_NAME" = "PalmDevices" ];then

        # The script receives resource information through the stdin
        echo "Receiving Palm Device Information" >> $OUT_FILE

        while read line
        do

            echo >> $OUT_FILE
            echo $line >> $OUT_FILE
            RESOURCE_TYPE=`echo $line | awk -F'|' '{print $4;}'`

            if [ $RESOURCE_TYPE -eq 1 ];then
                echo DMS Palm Device managed Resource >> $OUT_FILE
                if [ "$RESGRP_NAME" = "PalmDevices" ];then
                    # accept the resource
                    echo 1
                    echo Accepted >> $OUT_FILE
                    continue
                fi

                # refuse the resource
                echo 0
                echo Refused >> $OUT_FILE
                continue
            fi

            # refuse the resource
            echo Refused >> $OUT_FILE
```

```

        echo 0
    done
else
    # do not filter other Resource Groups
    echo >> $OUT_FILE
    echo Resource Group \'$RESGRP_NAME\' not handled by this policy>> $OUT_FILE
    echo TRUE
fi
else
    echo $RESGRP_TYPE not supported by the policy >> $OUT_FILE
    # do not filter anything else but Pervasive_Device resources
    echo TRUE
fi

echo >> $OUT_FILE
echo Exit>> $OUT_FILE
# Return 0 if everything is ok, !0 otherwise

exit 0

```

Script to Create Resource Group and Activate Default Policy

The following script creates a resource group of Windows CE devices and activates the default policy shown in “Script for Default Policy” on page 402.

```

#!/bin/sh
IRO=`wlookup InterRegion`
IRONAME=`idlattr -t -g $IRO name string`
IRONAME=`eval echo $IRONAME`
wresgrp new Pervasive_Device WinCEDevices $IRONAME -d -p
wputpolm -d ResourceGroup BasicResourceGroup dg_def_subscribers < ./sample_policy.sh

```

Chapter 21. Troubleshooting

This chapter provides information to help you do the following when using Resource Manager:

- Avoid problems
- Work around problems
- Gather information to analyze and solve problems

Tivoli Resource Manager Traces

Resource Manager generates a number of trace files containing information on internal operations for troubleshooting purposes. These trace files are used by IBM Software Support to determine the cause of the error.

Trace files can log operations with three levels of detail:

- 1 Logs error messages. This is the default value
- 2 Logs warning and error messages
- 3 Logs informational, warning and error messages

To modify the detail level, use the **odadmin environ set** command by adding the following variable to the Tivoli environment:

```
TRM_DEBUG_LEVEL=x
```

where:

x Is 1 (MIN), 2 (MID), or 3 (MAX)

To make this change effective, you have to stop and restart Resource Manager.

The following trace files are generated by Resource Manager and are located in the \$DBDIR on the Tivoli server:

- TRMDGMAAppMgr.log
- TRMDGMAAppMgrUI.log
- TRMDGMDowncalls.log
- TRMDGMRegistry.log
- TRMGroup.log
- TRMGroupUI.log
- TRMRDBMS.log
- TRMResourceManager.log
- TRMResourceManagerUI.log
- TRMUserUI.log

Tivoli Web Gateway Logs and Traces

Tivoli Web Gateway provides internal services for Tivoli Resource Manager. Information concerning problems and errors for Tivoli Resource Manager is written in the Tivoli Web Gateway logs. The following log files are available for the Tivoli Web Gateway on the application server:

DMSMsgn.log

This log file contains important information, warning, and error messages. This file is self-propagating and limited in size. The contents of the file are split into up to three separate files when the maximum size is exceeded. The resulting files are named DMSMsgn.log where *n* indicates the number of the message log file that switches between numbers 1, 2, and 3. The xxxMsg1.log file always contains the newest messages and the xxxMsg3.log file contains the oldest messages.

This file is located in `$WAS_HOME/logs/DMS_AppServer`

where:

`$WAS_HOME`

Is the WebSphere installation directory

n Is 1, 2, or 3

TraceDMSn.log

The TraceDMSn.log file contains trace information to be used for debugging purposes. To enable this file, set the appropriate key in the traceConfig.properties file, as described in “The traceConfig.properties File” on page 408. The trace file is located in `$WAS_HOME/logs/DMS_AppServer`

where:

`$WAS_HOME`

Is the WebSphere installation directory

SystemOut.log

The SystemOut.log file gathers standard out information from the DMS_AppServer application server. Use this log file to determine if DMS_AppServer was started without exceptions and to view trace messages when tracing is active. This file is located in `$WAS_HOME/logs/DMS_AppServer`

where:

`$WAS_HOME`

Is the WebSphere installation directory

SystemErr.log

The SystemErr.log file gathers standard error information from the DMS_AppServer application server. Use this log file to view exceptions that were written to the standard error stream. This file is located in `$WAS_HOME/logs/DMS_AppServer`

where:

`$WAS_HOME`

Is the WebSphere installation directory

The following trace file is available for the Tivoli Web Gateway on the endpoint:

TWGapi.log

The TWGapi.log file contains trace information to be used for debugging purposes. To enable this file, set the **tracerEnabled** key to true in the twg.cfg file. The default value is false. The trace file and the configuration file are located in `$LCF_DATDIR`

where:

\$LCF_DATDIR

Is the endpoint dat directory

Tivoli Web Gateway Configuration Files

This section describes Tivoli Web Gateway configuration files that can be changed and how those changes affect Tivoli Web Gateway. After you make any changes to these configuration files, you must restart WebSphere Application Server to make the changes effective.

The Tivoli Web Gateway configuration files that can be changed are:

- **Transaction.properties**. For more details, see “TheTransaction.properties file.”
- **traceConfig.properties**. For more details, see “The traceConfig.properties File” on page 408.

TheTransaction.properties file

The Transaction.properties file allows you to configure the database parameters for Tivoli Web Gateway. It contains the following parameters:

JDBC.dbUser

Identifies the DB2 user who owns the database tables in the Device Manager database. Device Manager is an internal Configuration Manager component, installed automatically with Configuration Manager. Device Manager interacts seamlessly with Tivoli Web Gateway to manage devices and users. The value for this parameter cannot be changed. The value must be **dmsadmin**.

JDBC.dbPassword

Identifies the password associated with **dmsadmin**.

JDBC.dbDriver

Identifies the Java Database Connectivity (JDBC) driver to be used by Device Manager. The default value is

`COM.ibm.db2.jdbc.app.DB2Driver`

JDBC.dbConnect

Identifies the Java Database Connectivity connect Web address. The default value is `jdbc\:db2\:dms`.

MinDBConnections

Identifies the minimum number of connections to the Device Manager database. This parameter can be changed to improve performance. The default value is 1.

MaxDBConnections

Identifies the maximum number of connections to the Device Manager database. This parameter can be changed to improve performance. The default value is 200. The recommended maximum value is **256** or **512**, depending on your environment.

This file is located in `$WAS_HOME/installedApps/hostname/DMS_WebApp.ear/dmsserver.war/WEB-INF/classes/`

where:

`$WAS_HOME`

Is the WebSphere installation directory

Managing the Transaction.properties File

To improve Tivoli Web Gateway server performance, set the database connection parameters in the Transaction.properties file to a higher value.

It is more efficient to open a larger number of database connections at startup rather than later, during run time. Later, opening database connections slows server capabilities to process requests, which affects device clients and users.

If you plan 70 to 100 concurrent device connections, use that range as a guideline to set the **MinDBConnections** parameter. The value for the **MaxDBConnections** parameter should be larger, but not more than the system can realistically handle. The recommended maximum value is **256** or **512**. The default value for the **MinDBConnections** parameter is 1.

When using a workstation with 1 gigabyte RAM, it is advisable to set the maximum number of database connections to 40 using the **MaxDBConnections** parameter in the Transaction.properties file.

The traceConfig.properties File

The traceConfig.properties file allows you to configure the trace settings for Tivoli Web Gateway. It contains the following parameters:

TraceLevel

Specifies the level of detail for trace files. Supported values are:

- 0 (none)
- 1 (fatal)
- 2 (fatal and error)
- 3 (fatal, error, and warning)

DISPLAY_ENTRY_EXIT

Specifies whether exit and entry events for Tivoli Web Gateway methods must be logged. Supported values are **true** and **false**. The default value is **true**.

MaxFileSize

Specifies the maximum size for each trace file. The default value is 512 KB.

MaxTraceFiles

Sets the maximum number of trace files to be created. When this number is reached, the oldest file is deleted.

Setting the keywords listed below to **true**, you can enable the tracing function for the corresponding components:

- component.console
- component.dmsserver
- component.event
- component.notification
- component.plugins
- component.resultscollector
- component.twgapi
- component.database

- component.enrollserver
- component.datconverter
- component.mcollect
- component.notificationhandler
- component.api
- component.userservices

This file is located in `$WAS_HOME/installedApps/hostname/DMS_WebApp.ear/dmsserver.war/WEB-INF/classes`

where:

`$WAS_HOME`

Is the WebSphere installation directory

The tracing function is intended for debugging purposes. If enabled for extended periods of time, tracing can decrease performance and slow the processing of the product considerably.

Tips for Performance Improvement

Device Manager is an internal Configuration Manager component, installed automatically with Tivoli Web Gateway. Device Manager interacts seamlessly with Tivoli Web Gateway to manage devices and users. To manage devices, a Device Manager server, which is comprised of a set of servlets running in an application server, works as the engine for processing operations on devices. A Device Manager database acts as the repository for device-related data and is synchronized with the Tivoli Web Gateway database on a regular basis.

Device Manager uses a Web server to redirect servlet calls and deliver Web pages, images, and documents over the network for viewing from remote Web browsers. The application server and Device Manager both require a supported Web server such as IBM HTTP Server.

The IBM HTTP Server is installed by IBM WebSphere Application Server, which is a requirement of Device Manager. The Web server must be installed on the workstation before Device Manager.

The port to be used for the Device Manager server must be configured on the Web server. Port 80, or port 443 if you are enabling optional SSL support, is required. The port numbers used by the Web server and IBM WebSphere Application Server must match one another.

IBM WebSphere Application Server uses the same port number used by the Device Manager server. This is the port number configured for Device Manager on your Web server. Either port 80, or port 443 for SSL, is recommended. These port numbers must match one another. Refer to the product documentation for all the Web server parameters that need to be supplied during Device Manager installation.

Tips to Improve Web Server Performance on AIX and Solaris Systems

To improve IBM HTTP Server performance on AIX systems and Solaris systems, you can adjust some of the parameters located in the following files:

/usr/IBMHttpServer/conf/httpd.conf

On AIX systems

/opt/IBMHttpServer/conf/httpd.conf

On Solaris systems

MinSpareServers

Determines the minimum number of extra inactive processes that should be kept available to handle new, incoming requests.

MaxSpareServers

Determines the maximum number of extra inactive processes that should be kept available to handle new, incoming requests.

MaxClients

Indicates the number of requests the Web server can handle at any given time before connections are denied. This parameter is limited by the amount of memory available on the server.

StartServers

Is the number of HTTP daemon processes that are started when the Web server is initialized.

The **MinSpareServers**, **MaxSpareServers**, and **MaxClients** parameters should be set to the same value and this value should be large enough to accommodate the number of device clients you expect the Web server to support. The performance is more efficient if you start a larger number of processes at startup rather than later, during run time. Later, HTTP daemon startups slow Web server capabilities to process requests, which affects device clients and therefore device users.

Because the **MaxClients** parameter is limited by the amount of memory available on the server, do not set the **MinSpareServers**, **MaxSpareServers**, and **MaxClients** parameters to a value higher than the total megabytes of memory available after all other processes are started. An HTTP daemon process uses about 788 KB memory.

Improving Application Server Performance

You can improve the application server performance by setting the Java heap size for DMS_AppServer to **256** MB or greater from the WebSphere Application Server Administrative Console. By default, these settings are low. If the server has 512 MB or more of memory, increasing the heap size would improve performance when garbage collection and heap reallocation occurs during run time.

To reset the heap size from the WebSphere Application Server Console, perform the following steps:

1. On the left pane, open **Servers > Application Servers** in the tree.
2. In the list of active servers, select the DMS_AppServer to be changed.
3. Open **Process Definition > Java Virtual Machine** in the tree.
4. In the **Initial Heap Size** field, type:
256
5. Click **Apply**.

If the application server is allowed too many concurrent connections, it affects performance. Nevertheless, some concurrence is needed to keep device clients from constantly being rejected. A maximum connections setting of 100 appears to provide optimum performance.

To increase the number of concurrent database connections from the WebSphere Application Server Administrative Console, perform the following steps:

1. On the left pane, open **Resources > JDBC Providers**
Select the database product you are using from the JDBC Providers list. If your database product is not displayed, see the WebSphere Application Server online help for information about adding a database to the list.
2. Select **Data Sources > Connection Pool**.
If no data sources are displayed, see the WebSphere Application Server online help for information about creating a data source.
3. In the **Maximum Pool Size** field, change the default value of 30 connections as appropriate. The recommended value is 100.
4. Click **Apply**.

Troubleshooting Connection Problems between Device Manager and Devices

If you experience connection problems between the Device Manager server and the devices it manages, this might depend on a DNS resolution issue. To work around this problem, change the value of the `DMS_PROXY_HOSTNAME` keyword from the fully qualified hostname to the IP address of the Device Manager proxy.

The `DMS_PROXY_HOSTNAME` keyword is located in the `variables.xml` file, which is stored in the following path:

```
$WAS_HOME/AppServer/config/cells/node_name/nodes/  
node_name/servers/DMS_AppServer
```

where:

`$WAS_HOME`

Is the WebSphere installation directory

`node_name`

Is the name of the node where WebSphere is installed

To make the change effective, stop and restart the Device Manager application server.

Deleting a Tivoli Endpoint That Is a Resource Gateway

Problem: You want to delete an endpoint that is a resource gateway from the Tivoli server. If you run the `wresgw ls` command after deleting the endpoint, you receive the message that the endpoint no longer exists.

Solution: The following example shows the steps that you should follow to remove traces of the endpoint in the Resource Manager database.

1. Run the command:

```
wresgw ls jlopez
```

where `jlopez` indicates the label of the endpoint that was reported as no longer existing. You receive the following informational message:

```
FBBRG0019I Available resource gateway types for endpoint 'jlopez':  
[TWG]  
Endpoint object ID:  
'1530040932.22.506+#TMF_Endpoint::Endpoint#'
```

2. Access the Resource Manager database and delete all the manager = '1530040932.22.506+#TMF_Endpoint::Endpoint#' records from the TRM_RESOURCES table in the Resource Manager database:

```
delete from trm_resources where manager = \
'1530040932.22.506+#TMF_Endpoint::Endpoint#'
```
3. Unregister the endpoint with the resource gateway using the command:

```
wresgw remove jlopez
```

Deleting and Reconnecting an Endpoint with the Same Label

Problem: You want to delete a Tivoli endpoint that is a resource gateway and reconnect it with the same endpoint label. If you run the **wresgw ls** command after deleting and reconnecting the endpoint, you receive the message that the endpoint object ID no longer exists.

Solution: The following example shows the steps that you should follow to redefine the endpoint in the Resource Manager database.

1. Run the command:

```
wresgw ls jlopez
```

where jlopez indicates the label of the endpoint for which the object ID was reported as no longer existing. You receive the following informational message:

```
FBBRG0019I Available resource gateway types for endpoint 'jlopez':
[TWG]
Endpoint object ID:
'1530040932.22.506+#TMF_Endpoint::Endpoint#'
```

2. To update the object ID of the endpoint, run the command:

```
wresgw update jlopez
```

You receive the following informational message:

```
FBBRG0031I The object ID of endpoint has been successfully updated.
Old object ID: '1530040932.22.506+#TMF_Endpoint::Endpoint#'
New object ID: '1530040932.23.506+#TMF_Endpoint::Endpoint#'
```

Deleting and Reconnecting an Endpoint with a New Label

Problem: You want to delete a Tivoli endpoint that is a resource gateway and reconnect it with the a new endpoint label. If you run the **wresgw ls** command after deleting and reconnecting the endpoint, you receive the message that the endpoint no longer exists.

Solution: The following example shows the steps that you should follow to redefine the endpoint in the Resource Manager database.

1. Run the command:

```
wresgw ls jlopez
```

where jlopez indicates the endpoint that was reported as no longer existing. You receive the following informational message:

```
FBBRG0019I Available resource gateway types for endpoint 'jlopez':
[TWG]
Endpoint object ID:
'1530040932.22.506+#TMF_Endpoint::Endpoint#'
```


2. To update the object ID and label of the endpoint to rmartin, run the command:

```
wresgw update jlopez rmartin
```

You receive the following informational message:

```
FBBRG0031I The object ID of endpoint has been successfully updated.
Old object ID: '1530040932.22.506+#TMF_Endpoint::Endpoint#'
New object ID: '1530040932.23.506+#TMF_Endpoint::Endpoint#'
```

Changing the User ID of Palm Devices

The ROM serial number of Palm devices is used as the user ID for the device. However, if the ROM does not have a serial number, then the user name of the device is used as the user ID. If you change the user ID on devices without a serial number, the device appears to be a new device and requires enrollment.

Resource Groups Do Not Exist

You find that resource groups that you have seen in the past no longer exist. These groups are created automatically by components with which Resource Manager interacts and are automatically deleted when they are no longer needed for an operation. It is recommended that you not work with resource groups with names that begin with `_INTERNAL_RESGRP_`

Grayed out Query Selection Box on Resource Group Members Dialog

Problem: When you open either a dynamic or static resource group of the type User, the Query Selection group box on the Resource Group Members dialog is greyed out.

Solution: Ensure that you have run the `update_trm_query.sh` script from the `$BINDIR/TAS/DirQuery/SCRIPTS` directory on the Tivoli server.

Activity Planner login failure on Linux®

Problem: When you start the Activity Planner on Linux, the following error might occur during the login:

```
AMN0121E Activity Planner initialization failed. Check whether
the Activity Planner user has been created correctly and/or
the user and password maintained by Activity Planner are synchronized
with the corresponding values of the operating system.
```

and the `oservlog` might contain reports similar to the following:

```
2005/09/19 10:50:58 -01: PAM: pam_acct_mgmt failed=User account has expired (13)
2005/09/19 10:50:58 -01: @verify_password: Invalid username or password
2005/09/19 10:50:58 -01: @rconnect: Login failed for tivapm from host 100007f
```

Solution: Go to `/etc/pam.d` and edit the `oserv` file

```
##PAM-1.0
# Created by etc-tivoli.cfg for DS/Win and JCF login on systems with Pluggable
# Authentication Modules (PAM). Install will not overwrite this file if it
# exists. See the PAM doc for your platform for details on modifying this file.
auth    required          /lib/security/pam_unix.so
```

Add the following line:

```
account required          /lib/security/pam_unix.so
```

Part 6. Managing an Enterprise Directory

Chapter 22. Using the GUI	417	Directory Query Context Commands	433
Creating a Directory Query Library	417	wcrtdirctx	434
Creating a Directory Query.	418	wgetdirctx	436
Editing a Directory Query	420	wsetdirctx	437
Using Directory Queries.	421	wsetdirctxpw	438
Running a Directory Query.	421	wmanagedir.	439
 Chapter 23. Using the Command Line	425	 Chapter 24. Troubleshooting	441
Directory Query Commands	425	Enabling Traces.	441
wcrtdirquerylib.	426	Registering Enterprise Directory Query Facility	
wcrtdirquery	427	Resources	441
wsetdirquery	429	Values set for the LD_LIBRARY_PATH	
wgetdirquery	431	environment variable are lost	441
wrunidirquery	432		

The Enterprise Directory Query Facility allows users to use information stored in Enterprise directories to create lists for software distribution or inventory scanning. It allows a Configuration Change Manager (Change Manager) administrator to select a specific directory object or container of directory objects as subscribers to a Change Manager reference model.

Locating Related Information

Information related to this service is available from the following sources:

- *IBM Tivoli Configuration Manager: Introducing IBM Tivoli Configuration Manager.* This provides an overview of the service.
- *IBM Tivoli Configuration Manager: Planning and Installation Guide.* This includes information about how you install the service.
- *IBM Tivoli Configuration Manager: Messages and Codes.* This provides details of all messages generated by the IBM Tivoli Configuration Manager components and services.

Chapter 22. Using the GUI

The Enterprise Directory Query Facility is designed to allow a Configuration Change Manager (Change Manager) administrator to use information stored in enterprise directories to select a specific directory object or container of directory objects as subscribers to a reference model. Subscribers to a reference model can then be targets for software distribution or inventory scan operations.

The Enterprise Directory Query Facility enables you to access the information contained in an enterprise directory server, for example in the Windows 2000 active directory server.

The directory query feature consists of directory query libraries and directory queries. Directory query libraries reside in policy regions and are created to contain directory queries. Directory queries enables you to find information about the users or the workstations defined in the enterprise directory server.

This chapter contains instructions for using the Enterprise Directory Query Facility graphical user interface to perform the following tasks:

- Creating a Directory Query Library.
- Creating a Directory Query.
- Editing a Directory Query.
- Running a Directory Query.

Creating a Directory Query Library

Directory queries reside in directory query libraries. Before you can create a directory query, you must first create a directory query library in which to store it. Use directory query libraries to organize similar directory queries into logical groups. For example, in each policy region, you could create a directory query library to hold the directory queries that select subscribers for Tivoli Software Distribution software packages. Each directory query and directory query library in the Tivoli management region must have a unique name.

The following table provides the context and authorization roles required to perform this task:

Activity	Context	Required Role
Creating a directory query library	Policy region	senior or super

Use the following procedure to create a directory query library. For information about creating a directory query library using the command line, see “wcrtdirquerylib” on page 426.

1. Select the policy region in which you want to create the directory query library, and add the Directory Query Library resource to the policy region list of managed resources.
2. From the policy region, select **DirectoryQueryLibrary** from the **Create** menu. The **Create Directory Query Library** dialog box is displayed:

Creating a Directory Query Library



3. In the **Name/Icon Label** text box, type a unique name for the directory query library in the **Name/Icon Label** text box.
4. Click **Create & Close**. Tivoli Management Framework creates the directory query library and returns to the **Policy Region** window. The policy region will contain an icon for the new directory query library.

Creating a Directory Query

When you create a directory query, you specify the repository in which to search for information and the set of information you want to retrieve. For example, from enterprise directories you can retrieve such enterprise-specific information as user names, phone numbers, and e-mail addresses.

The **Create Directory Query** dialog box enables you to select a subset of the attributes in the view and specify a search string that returns the information you need from that view. When you create a directory query, you must specify the following items:

- A **Directory Query Name** that is unique in the Tivoli management region.
- A **Directory Context** is a set of information required to connect to a directory server. Directory context information is created at installation time. The default value is **directory**.

Notes:

1. You can also create other directory contexts using the CLI. For information about this, see “wcrtdirctx” on page 434. When running operations on directory queries you must use the **directory** directory context.
 2. At installation, the default directory context “directory” is created. If you delete this default directory context and create a new directory context named “directory” the IDL calls used by Tivoli Resource Manager will not work correctly. Use the **wsetdirquery** command if you want to change some parameters for the default directory context. If you want to set a different value to run a generic query, create and use a new directory context using the **wcrtdirctx** command.
- A set of chosen attributes to be part of the retrieved information.
 - Specify a description for the directory query and a search clause. A search clause specifies which information the directory query will return. The search clause information is created using the RCF-2254 standards.

The following table provides the context and authorization roles required to perform this task:

Activity	Context	Required Role
Creating a query	Query library	admin, senior or super

Use the following procedure to create a directory query. For information about creating a directory query using the command line, see “wcrtdirquery” on page 427.

To create a query from the desktop, perform the following steps:

1. From the policy region, double-click the icon of the directory query library in which the directory query will reside. The directory query library window is displayed.
2. From the **Create** menu, select **Directory Query**. The **Create Directory Query** dialog box is displayed:



3. In the **Directory Query Name** text box, type a unique name for the directory query. The name can be any set of alphanumeric characters, uppercase letters, or lowercase letters.
4. In the **Description** text box, type a brief description of the directory query. This text box is optional.
5. From the **Directory Context** drop-down list, which lists the names of the directory context objects in the local Tivoli management region, select a directory context against which the query will run.
6. In the **Naming Context** text box, type a naming context. Specify the path in LDAP format from where to begin the search.
7. In the **Scope** scrolling list specifies the point in the directory tree hierarchy at which you begin the search. Possible values are:

object The search is performed only on the specified object.

one level

The search is performed on one level of the tree.

subtree

The search is performed on the specified tree and on all the descendent directories.

Creating a Directory Query

8. In the **Chosen Attributes** scrolling list, select the attributes about which you want to retrieve information. The attributes you select will be added to the attribute field.
Click **Close** to return to the **Create Directory Query** dialog box.
9. To add an attribute to the **Chosen Attributes** list, type an attribute in the **Attribute** text box and click **Add**.
10. To edit an attribute in the **Chosen Attributes** list, choose an attribute and click **Edit**. The attribute appears in the **Attribute** text box. Modify it and click **Replace**. You must specify the attributes related to the object you are searching for.
11. Enter a search string in the **Search String** scrolling text area to specify what information for which the directory query will search.
12. Select **Ignore Case**, if necessary.
13. In the **Query Timeout** text box, type the required directory response time in milliseconds. The default is -1, which corresponds to unlimited response time.
14. In the **Max Number of Entries** text box, type the maximum number of entries to be returned. The default is -1, which corresponds to all entries.
15. Click **Create & Close** to return to the Directory Query Library window.

Editing a Directory Query

After you have created a directory query, you can change information in any field on the dialog box.

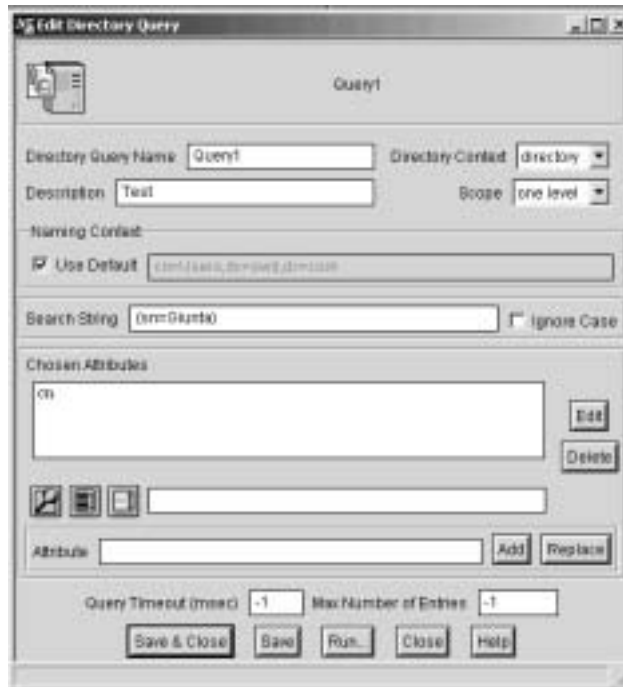
The following table provides the context and authorization roles required to perform this task:

Activity	Context	Required Role
Editing a directory query	Policy region	admin, senior, super, or query_directory_edit

Use the following procedure to edit a directory query. For information about editing a directory query using the command line, see “wsetdirquery” on page 429.

1. Either right-click or double-click the directory query icon.
Right-click the icon of the directory query you want to change in a directory query library, and then select **Edit Directory Query** from the pop-up menu.

Alternatively, double-click the directory query icon. The **Edit Directory Query** dialog box is displayed. This dialog box is the same as the **Create Directory Query** dialog box.



2. Change the **Directory Query Name**, **Directory Context**, **Naming Context**, **Chosen Attributes**, **Attributes**, or **Search String** fields as necessary. Refer to “Creating a Directory Query” on page 418 for an explanation of the fields on this dialog box.
3. Click **Save & Close** to save the changes.

Using Directory Queries

After you create or edit a directory query, you can run the directory query to access information in a directory server.

The main function of the directory query facility is to access information in a directory server. You can run a directory query to access general information display the query results or save the results in a file. For example, you might want to query the user e-mail address information stored in the enterprise directory.

To run any directory query and display or save the results, perform *one* of the following actions:

- Run **wrundirquery**.
- Click **Run Query** on the **Edit Query** dialog box.
- Click **Run Query** on the **Create Query** dialog box.
- Right-click the icon of the query and select **Run Query**.

Refer to “Running a Directory Query” for more information.

Running a Directory Query

You can run a directory query and view the results or save them in a file.

Using Directory Queries

The following table provides the context and authorization roles required to perform this task:

Activity	Context	Required Role
Running a directory query	Policy region	admin, senior, super, or query_directory_execute

Use the following procedure to run a directory query. For information about run a directory query using the command line, see “wrundirquery” on page 432.

1. Either right-click the icon or select **Run Query**.

Right-click the icon of a directory query in a directory query library, and then select **Run Query** from the pop-up menu.

Alternatively, select **Run Query** from either the **Create Query** or **Edit Query** dialog box.

The **Run Query** dialog box is displayed, showing the results of the directory query in tabular form:



2. If you want to save the results of the directory query, you can do so from the **Run Query** dialog box as follows:
 - a. In the **Run Query** dialog box, click the **Export** button. The **Export Query Results** dialog box is displayed:



- b. In the **Host** text box, type the name of the managed node on which you would like to save the results file. If you do not specify a managed node, the file will be saved on the local machine.
- c. In the **File** text box, type the location and name of the directory query results file.

If you do not know the location and name of the directory query results file, click the ellipses (...) button to browse the file system.

- d. Click one of the **Delimiter** radio buttons to specify how to separate entries in the query results file. If you want to use a delimiter other than a comma or a tab, click **Custom** and type a delimiter in the text box immediately to the right of it.
- e. Select **Print Headers** if you want the output file to include the name of the directory query, the number of rows, and the names of the columns.
- f. Click **Save & Close** to create the file.

Chapter 23. Using the Command Line

This chapter describes the Enterprise Directory Query Facility command line interface (CLI). For information about the Enterprise Directory Query Facility graphical user interface (GUI), see Chapter 22, “Using the GUI,” on page 417.

Enterprise directory query facility commands can be categorized as either Directory Query commands or Directory Query context commands. Directory query commands allow you to create and manage directory queries and directory query libraries. Directory context commands allow you to manage and specify the directory context objects to associate with specific directory queries and specific directory query libraries.

Table 57 shows the commands supported by the Enterprise Directory Query Facility command line.

Table 57. Enterprise Directory Query Facility Commands

Command	Task	Page
wcrtdirquerylib	Create a directory query library	426
wcrtdirquery	Create a directory query	427
wsetdirquery	Modify a directory query	429
wgetdirquery	Display a directory query	431
wrundirquery	Run a directory query	432
wcrtdirctx	Create a directory query context	434
wgetdirctx	Display information about a directory query context	436
wsetdirctx	Modify information on a directory query context	437
wsetdirctxpw	Set directory query context password	438
wmanagedir	Manage a directory services database.	439

Directory Query Commands

This section contains information about the directory query commands. For information about directory query context commands see, “Directory Query Context Commands” on page 433.

wcrtdirquerylib

Creates a directory query library.

Note: You can also create a directory query library from the GUI. For information about this, see “Creating a Directory Query Library” on page 417.

Syntax

wcrtdirquerylib *policy_region dirquery_lib_name*

Description

The **wcrtdirquerylib** command allows you to create a directory query library.

Options

policy_region

The policy region name.

dirquery_lib_name

The directory query library name.

Authorization

senior or super role.

Examples

To create a directory query library named Firea3 in policy region ED:

```
wcrtdirquerylib ED Firea3
```

See Also

wcrtdirquery, wsetdirquery, wgetdirquery, wrundirquery.

wcrtdirquery

Creates a directory query.

Note: You can also create a directory query from the GUI. For information about this, see “Creating a Directory Query” on page 418.

Syntax

wcrtdirquery [-**d** *description*][-**r** *dirquery_context_name*][-**m** *milliseconds*][-**e** *max_entries*][-**a** *attribute*]... [-**C** *y|n*] [-**O** *0|1|2*][-**c** *naming_context*] -**q** *search_string*
dirquerylib_name dirquery_name

Description

The **wcrtdirquery** command allows you to create a directory query.

Options

d *description*

Describes the directory query library.

r *dirquery_context_name*

Specifies a directory context object label.

m *milliseconds*

Specifies the milliseconds to wait for directory response. The default is unlimited response time.

e *max_entries*

Specifies the maximum number of entries to be returned. The default is all entries.

a *attribute*

Specifies an attribute to be queried.

C *y/n* Specifies that case is to be ignored.

O *0/1/2*

Specifies the point in the directory tree hierarchy at which you begin the search

0 The search if performed only on the specified object.

1 The search if performed on one level of the tree.

2 The search if performed on the specified tree and on all the descendent directories.

c *naming_context*

Specifies the naming context.

q *search_string*

Specifies a search string using the syntax of RFC- 2254.

dirquerylib_name

Specifies the name of the directory query library.

dirquery_name

Specifies the name of the directory query.

Authorization

admin, senior or super role.

Examples

To create a directory query named Queryfirea3 in directory query library named Firea3:

```
wcrtdirquery -r firea3ctx -m 100 -e 10 -a cn -a name -a surname  
-q\"(cn=armando\" Firea3 Queryfirea3
```

Note: The \ is required only on Unix systems.

See Also

wcrtdirquerylib, wsetdirquery, wgetdirquery, wrundirquery.

wsetdirquery

Edits directory query properties.

Note: You can also edit directory query properties from the GUI. For information about this, see “Editing a Directory Query” on page 420.

Syntax

```
wsetdirquery [-n new_name][-d description][-r dirquery_context_name][-m
milliseconds][-e max_entries][-a attribute]... [-u attribute]...[-C y|n] [-O 0|1|2] [-c
naming_context][-q search_string] dirquery_name
```

Description

The **wsetdirquery** command allows you to set directory query properties.

Options

n *new_name*

Specifies the new name of the directory query library.

d *description*

Describes the directory query library.

r *dirquery_context_name*

Specifies a directory context object label.

m *milliseconds*

Specifies the milliseconds to wait for directory response. The default is unlimited response time.

e *max_entries*

Specifies the maximum number of entries to be returned. The default is all entries.

a *attribute*

Specifies an attribute to be queried.

u *attribute*

Specifies the attribute to delete from the list.

C *y/n* Specify that case is to be ignored.

O *0/1/2*

Specifies the point in the directory tree hierarchy at which you begin the search

0 The search if performed only on the specified object.

1 The search if performed on one level of the tree.

2 The search if performed on the specified tree and on all the descendent directories.

c *naming_context*

Specifies the naming context.

q *search_string*

Specifies a search string using the syntax of RFC-2254.

dirquery_name

Specifies the name of the existing directory query.

Authorization

query_directory_edit, admin, senior or super role.

Examples

To set properties on a directory query named Queryfirea3:

```
wsetdirquery -r firea3ctx -m 100 -e 10 -a cn -a name  
-a surname -q\ (cn=armando\ ) -c Firea3 Queryfirea3
```

Note: The \ is required only on Unix systems.

See Also

wcrtdirquerylib, wcrtdirquery, wgetdirquery, wrundirquery.

wgetdirquery

Lists directory query properties.

Syntax

wgetdirquery [-f] *dirquery_name*

Description

The **wgetdirquery** command allows you to list directory query properties.

Options

f Specifies that all properties are displayed. If not used, then just the search string displays.

dirquery_name

Specifies the name of the directory query.

Authorization

user, query_directory_view, admin, senior or super role.

Examples

To display all properties for a directory query named Queryfirea3:

```
wgetdirquery -f Queryfirea3
```

See Also

wcrtdirquerylib, wcrtdirquery, wsetdirquery, wrundirquery.

wrundirquery

Queries the enterprise directory.

Syntax

wrundirquery [-n] [[-h *host_name*] -f *file_name*] *dirquery_name*

Description

The **wrundirquery** command queries the enterprise directory and returns a list of directory entries.

Options

n Specifies whether to show the directory header.

-h *host_name*

Sets the managed node name to which to write the result.

-f *file_name*

Sets the file name to which to write the result.

dirquery_name

Specifies the name of the directory query.

Authorization

query_directory_execute, admin, senior or super role.

Examples

To run a directory query named Queryfirea3 and write the results to a specified host and file:

```
wrundirquery -h host1 -f file1 Queryfirea3
```

See Also

wcrtdirquerylib, wcrtdirquery, wsetdirquery, wgetdirquery.

Directory Query Context Commands

This section contains information about the directory query context commands. For information about directory query commands see “Directory Query Commands” on page 425.

wcrtdirectx

Creates a directory query context.

Syntax

```
wcrtdirectx [-i] -s server -u user -c naming_context [-f provider] [-p port] [-P ssl_port]
[-S y/n] [-k keystore_path] directory_context_name
```

Note: If you specify the **-k** option, the corresponding password is retrieved. If you specify the **-i** option, then the password or passwords are retrieved from standard input, separated by newline code with the directory password listed first. If you do not want passwords to display, omit the **-i** option.

Description

The **wcrtdirectx** command allows you to create a directory query context.

Options

i input Specifies that the password must be read from standard input.

s server

Specifies the server.

u user Specifies the directory server administrator or a user with equivalent authority.

c naming_context

Specifies the naming context to use for the search.

f provider

Specifies the java class name that identifies the directory access protocol. The default is "com.sun.jndi.ldap.LdapCtxFactory".

p port Specifies a server port for a non-secure connection. If not specified, the default value 389 is used.

P ssl_port

Specifies a server port for a secure (SSL) connection. If not specified, the default value 636 is used.

S y/n Enables the Secure Sockets Layer (SSL) between the directory server and the Tivoli region. Y specifies enable, n specifies do not enable. If not specified, SSL is disabled.

k keystore_path

Specifies the path of the keystore containing the certificates used during an SSL connection. If you specify this option you must specify also the *keystore_path* password.

dirquery_context_name

Specifies the name of the new directory query context. When running operations on directory queries you must use the **directory** directory context.

Authorization

senior or super role.

Examples

1. To create a directory query context named firea3ctx for a connection to a Windows2000 server directory with IP address 69.99.20.3:

```
wcrtdirectx -s 69.99.20.3 -u cn=Administrator,cn=Users,dc=enterprise,  
dc=com -c cn=Users,dc=enterprise,dc=com -p 389 firea3ctx
```

Note: Port 389 is the TCP port for LDAP service. cn= and dc= is standard syntax for LDAP queries.

2. To create a directory query context named firea3ctx for a connection to a Novell server directory:

```
wcrtdirectx -s armando.enterprise.com -u cn=admin,o=context1  
-c o=context1 -p 389 novellctx
```

Note: Port 389 is the TCP port for LDAP service. cn= and dc= is standard syntax for LDAP queries. o=context1 is the Novell directory context for users of the Novell Directory domain.

See Also

wgetdirectx, wsetdirectx, wsetdirectxpw, wmanagedir.

wgetdirctx

Gets information on a directory query context object.

Syntax

wgetdirctx *directory_context_name*

Description

The **wgetdirctx** command allows you to list information about a directory query context object.

Options

directory_context_name

Specifies the name of the context from which to retrieve information.

Authorization

senior or super role.

Examples

To list a directory query context object named firea3ctx:

```
wgetdirctx firea3ctx
```

See Also

wcrtdirctx, wsetdirctx, wsetdirctxpw, wmanagedir.

wsetdirctx

Sets information on a directory query context.

Syntax

wsetdirctx [-**n** *name*] [-**s** *server*] [-**u** *user*-**c** *naming_context*] [-**f** *provider*] [-**p** *port*] [-**P** *ssl_port*] [-**S** *y/n*] [-**k** *keystore_path*] *dirquery_context_name*

Description

The **wsetdirctx** command allows you to set information on a directory query context.

Options

n *name*

Specifies the name of the context.

s *server*

Specifies the server.

u *user* Specifies the directory server administrator or a user with equivalent authority.

c *naming_context*

Specifies the naming context to use for the search.

f *provider*

Specifies the java class name that identifies the directory access protocol. The default is "com.sun.jndi.ldap.LdapCtxFactory".

p *port* Specifies a server port for a non-secure connection. If not specified, the default value 389 is used.

P *ssl_port*

Specifies a server port for a secure (SSL) connection. If not specified, the default value 636 is used.

S *y/n* Enables the Secure Sockets Layer (SSL) between the directory server and the Tivoli region. Y specifies enable, n specifies do not enable. If not specified, SSL is disabled.

k *keystore_path*

Specifies the path of the keystore containing the certificates used during an SSL connection.

Authorization

senior or super role.

Examples

To set information on a directory query context named firea3ctx:

```
wsetdirctx -n firea3ctx
```

See Also

wcrtdirctx, wgetdirctx, wsetdirctxpw, wmanagedir.

wsetdirctxpw

Sets the directory query context password.

Syntax

wsetdirctxpw [-**k**] dirquery_context_name [*old-password new-password*]

Description

The **wsetdirctxpw** command allows you to set the directory query context password.

Options

k To change the password of the keystore. If this option is not used, the password is changed only for the specified context.

dirquery_context_name

Specifies the name of the directory query context.

old-password new-password

To verify the old password and specify the new password.

Authorization

senior or super role.

Examples

To set the password for a directory query context named firea3ctx:

```
wsetdirctxpw firea3ctx pswdold pswdnew
```

See Also

wcrtctx, wgetctx, wsetctx, wmanagedir.

wmanagedir

Manages a Directory Services database.

Syntax

wmanagedir {-a *dn* -o *tmeObjectId* [-l *tmeObjectLabel*] | -m *dn* {-o *tmeObjectId* | -l *tmeObjectLabel*} | -r *dn* | -f *filename* [-x] [-s]} *dirquery_context_name*

Description

The **wmanagedir** command manages a Directory Services database.

Options

a dn To create an association between an endpoint and the user that you specify in the *dn* domain name.

o *tmeObjectId*

Specifies the *tmeObjectId* associated with the user *dn*.

l *tmeObjectLabel*

Specifies the label that you want associated with the *tmeObjectId*.

m dn To modify the association between an endpoint and the user that you specify in the *dn* domain name.

o *tmeObjectId*

Specifies the *tmeObjectId* associated with the user *dn*.

l *tmeObjectLabel*

Specifies the label that you want associated with the *tmeObjectId*.

r dn To remove the association between the endpoint and the user that you specify in the *dn* domain name.

f *filename*

Specify a filename that contains the commands to modify, add or remove the association between the user and the endpoint.

The file should contain one command per line with the following syntax:

```
{-m|-a|-r} [-l dn tmeObjectLabel] [-o tmeObjectId]
```

See the example below. A line beginning with # is considered a comment.

```
# test file to populate the DS database
-r cn=test,cn=Users,dc=swd,dc=com
-a cn=test1,cn=Users,dc=swd,dc=com -l ep1 -o oid1
#comment
#comment
-m cn=test,cn=Users,dc=swd,dc=com -o oid2 -l ep2
-m cn=test,cn=Users,dc=swd,dc=com -o oid2
-m cn=test,cn=Users,dc=swd,dc=com -o oid4 -l ep4
```

The default behavior is to execute all the commands in a sequence. For each command that fails an error message is displayed.

x To perform a validation on the syntax of the file that contains the commands. The commands are not performed.

s To perform the commands specified in the file and stop the execution when the first occurs.

dirquery_context_name

Specifies the directory query context name.

Authorization

admin, senior or super role.

Examples

1. To modify the parameters for the user stored in the active directory database:

```
wmanagedir -m cn=Armando,cn=Users,dc=enterprise,dc=com 10 build19 20  
-u cn=Administrator,cn=Users,dc=enterprise,dc=com -pwd tivoli  
-s69.99.20.3
```

Note: If you are associating parameters for the first time, you must use the **-a** option rather than the **-m** option.

2. To modify the parameters for the user stored in the active directory database using SSL support:

```
wmanagedir -m cn=Armando,cn=Users,dc=enterprise,dc=com 1453466303  
lab15161 -nt1453466303.5.506 -u cn=Administrator,cn=Users,  
dc=enterprise,dc=com -pwd tivoli -s69.99.20.3 -P 636 -keystore  
/armando/impcert.cer -k tivoli -S y
```

Note: For the Novell Directory Server, the **wmanagedir** command is the same except for the syntax of the user and the administrator.

See Also

None

Chapter 24. Troubleshooting

This chapter gives an overview of the troubleshooting process and provides descriptions of sources of information that will help you in solving problems with Enterprise Directory operations. It includes the following topics:

- Enabling Traces
- Registering Enterprise Directory Query Facility Resources

Enabling Traces

To enable the trace file for the Directory Query component you must add a new variable in Tivoli environment.

DQ_TRACE = n (where n is the size in MB of DirQuery trace file)

The trace file will be written into \$DBDIR directory. DirQueryCli0.trc will contain the CLI trace while DirQueryEngine0.trc will contain the ENGINE trace.

Registering Enterprise Directory Query Facility Resources

For resource management of endpoints as users, you need to install both the Resource Manager and Enterprise Directory Query Facility components. Because the installation of the Enterprise Directory Query Facility component runs the **wresource** command, you should install the Resource Manager component before installing the Enterprise Directory Query Facility component. If you installed the Enterprise Directory Query Facility component before installing the Resource Manager component, you must run the **wresource** command as follows:

```
wresource add TMF_Query QueryDirectory OBJECT_NIL User
```

For additional information about the command, see “wresource” on page 394.

Values set for the LD_LIBRARY_PATH environment variable are lost

If you are using a Sybase database in your Tivoli environment and the manual configuration of the environment is performed before installing the Directory Query component, then the same configuration should be repeated after the Directory Query installation because the settings of the LD_LIBRARY_PATH environment variable are overwritten by the Directory Query installation.

Appendix A. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in this product enable users to do the following:

- Use assistive technologies, such as screen-reader software and a digital speech synthesizer, to hear what is displayed on the screen. Consult the product documentation of the assistive technology for details on using those technologies with this product.
- Operate specific or equivalent features using only the keyboard.
- Magnify what is displayed on the screen.

In addition, the product documentation has been modified to include features to aid accessibility:

- All documentation is available in both HTML and convertible PDF formats to give the maximum opportunity for users to apply screen-reader software.
- All images in the documentation are provided with alternative text so that users with vision impairments can understand the contents of the images.

Navigating the Interface Using the Keyboard

Standard shortcut and accelerator keys are used by the product and are documented by the operating system. Refer to the documentation provided by your operating system for more information.

Using Java-based GUIs you press the Tab key to select GUI buttons. Perform the function related to the selected button by pressing:

- **Enter** for the default selection
- The spacebar for all other selections

Tables listing the additional keyboard shortcuts that you can use to navigate inside the windows of each service follow.

Activity Planner

The following tables list the additional keyboard shortcuts that you can use to navigate inside the windows of the Activity Plan Editor and the Activity Plan Monitor.

Table 58. Keyboard shortcuts for the Activity Plan Editor service

Menu Selection	Menu Item	Shortcut	Function Performed
File	New	Ctrl+N	Creates a new activity plan
File	Open	Ctrl+O	Opens an existing activity plan
View	Activity Properties	Ctrl+A	Opens the Activity Properties dialog
View	Targets	Ctrl+T	Opens the Add Target dialog
View	Plan Properties	Ctrl+P	Opens the Plan Properties notebook

Navigating the Interface Using the Keyboard

Table 59. Keyboard shortcuts for the Activity Plan Monitor service

Menu Selection	Menu Item	Shortcut	Function Performed
Plans	Submit	Ctrl+S	Submits an activity plan
Selected	Variables	Ctrl+I	Opens the Variables panel
Selected	Show Details	Ctrl+D	Displays details about the selected plan or activity in the main window
View	Reload data from APM database	Ctrl+R	Reloads data from Activity Planner database
		F2	Opens the Update Trace Level dialog

Configuration Change Manager

The following table lists the additional keyboard shortcuts that you can use to navigate inside the windows of Configuration Change Manager:

Table 60. Keyboard shortcuts for Configuration Change Manager

Shortcut	Function Performed
F2	Enables logging function.

Magnifying What Is Displayed on the Screen

You can enlarge information on the product windows using facilities provided by the operating systems on which the product is run. For example, in a Microsoft Windows environment, you can lower the resolution of the screen to enlarge the font sizes of the text on the screen. Refer to the documentation provided by your operating system for more information.

Enabling the IBM Voice Kit to Function with the Activity Planner and Configuration Change Manager GUIs

To enable the IBM Voice Kit to function with the Activity Planner and Configuration Change Manager GUIs, you must add the path to where the Java Runtime Environment (JRE) is installed to the PATH system environment variable. The JRE is installed with the Tivoli_JRE_*interp*.spb package

where

interp is the name of the platform.

For example, if you are working in a Windows environment, add the following path to the PATH system environment variable:

C:\Program Files\Tivoli\JavaTools\JRE\1.3.0\jre

Appendix B. Support information

This section describes the following options for obtaining support for IBM products:

- “Searching knowledge bases”
- “Obtaining fixes”
- “Contacting IBM Software Support” on page 446

Searching knowledge bases

If you have a problem with your IBM software, you want it resolved quickly. Begin by searching the available knowledge bases to determine whether the resolution to your problem is already documented.

Search the information center on your local system or network

IBM provides extensive documentation that can be installed on your local computer or on an intranet server. You can use the search function of this information center to query conceptual information, instructions for completing tasks, reference information, and support documents.

Search the Internet

If you cannot find an answer to your question in the information center, search the Internet for the latest, most complete information that might help you resolve your problem. To search multiple Internet resources for your product, expand the product folder in the navigation frame to the left and select **Web search**. From this topic, you can search a variety of resources including:

- IBM technotes
- IBM downloads
- IBM Redbooks®
- IBM developerWorks®
- Forums and newsgroups
- Google

Obtaining fixes

A product fix might be available to resolve your problem. You can determine what fixes are available for your IBM software product by checking the product support Web site:

1. Go to the IBM Software Support Web site (<http://www.ibm.com/software/support>).
2. Under **Products A - Z**, select your product name. This opens a product-specific support site.
3. Under **Self help**, follow the link to **All Updates**, where you will find a list of fixes, fix packs, and other service updates for your product. For tips on refining your search, click **Search tips**.
4. Click the name of a fix to read the description and optionally download the fix.

To receive weekly e-mail notifications about fixes and other news about IBM products, follow these steps:

1. From the support page for any IBM product, click **My support** in the upper-right corner of the page.
2. If you have already registered, skip to the next step. If you have not registered, click **register** in the upper-right corner of the support page to establish your user ID and password.
3. Sign in to **My support**.
4. On the My support page, click **Edit profiles** in the left navigation pane, and scroll to **Select Mail Preferences**. Select a product family and check the appropriate boxes for the type of information you want.
5. Click **Submit**.
6. For e-mail notification for other products, repeat Steps 4 and 5.

For more information about types of fixes, see the *Software Support Handbook* (<http://techsupport.services.ibm.com/guides/handbook.html>).

Contacting IBM Software Support

IBM Software Support provides assistance with product defects.

Before contacting IBM Software Support, your company must have an active IBM software maintenance contract, and you must be authorized to submit problems to IBM. The type of software maintenance contract that you need depends on the type of product you have:

- For IBM distributed software products (including, but not limited to, Tivoli, Lotus®, and Rational® products, as well as DB2® and WebSphere® products that run on Windows or UNIX operating systems), enroll in Passport Advantage® in one of the following ways:
 - **Online:** Go to the Passport Advantage Web page ([http://www.lotus.com/services/passport.nsf/WebDocs/ Passport_Advantage_Home](http://www.lotus.com/services/passport.nsf/WebDocs/Passport_Advantage_Home)) and click **How to Enroll**
 - **By phone:** For the phone number to call in your country, go to the IBM Software Support Web site (<http://techsupport.services.ibm.com/guides/contacts.html>) and click the name of your geographic region.
- For IBM eServer™ software products (including, but not limited to, DB2 and WebSphere products that run in zSeries®, pSeries®, and iSeries® environments), you can purchase a software maintenance agreement by working directly with an IBM sales representative or an IBM Business Partner. For more information about support for eServer software products, go to the IBM Technical Support Advantage Web page (<http://www.ibm.com/servers/eserver/techsupport.html>).

If you are not sure what type of software maintenance contract you need, call 1-800-IBMSERV (1-800-426-7378) in the United States or, from other countries, go to the contacts page of the IBM Software Support Handbook on the Web (<http://techsupport.services.ibm.com/guides/contacts.html>) and click the name of your geographic region for phone numbers of people who provide support for your location.

Follow the steps in this topic to contact IBM Software Support:

1. Determine the business impact of your problem.
2. Describe your problem and gather background information.
3. Submit your problem to IBM Software Support.

Determine the business impact of your problem

When you report a problem to IBM, you are asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem you are reporting. Use the following criteria:

Severity 1	Critical business impact: You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.
Severity 2	Significant business impact: The program is usable but is severely limited.
Severity 3	Some business impact: The program is usable with less significant features (not critical to operations) unavailable.
Severity 4	Minimal business impact: The problem causes little impact on operations, or a reasonable circumvention to the problem has been implemented.

Describe your problem and gather background information

When explaining a problem to IBM, be as specific as possible. Include all relevant background information so that IBM Software Support specialists can help you solve the problem efficiently. To save time, know the answers to these questions:

- What software versions were you running when the problem occurred?
- Do you have logs, traces, and messages that are related to the problem symptoms? IBM Software Support is likely to ask for this information.
- Can the problem be re-created? If so, what steps led to the failure?
- Have any changes been made to the system? (For example, hardware, operating system, networking software, and so on.)
- Are you currently using a workaround for this problem? If so, please be prepared to explain it when you report the problem.

Submit your problem to IBM Software Support

You can submit your problem in one of two ways:

- **Online:** Go to the "Submit and track problems" page on the IBM Software Support site (<http://www.ibm.com/software/support/probsub.html>). Enter your information into the appropriate problem submission tool.
- **By phone:** For the phone number to call in your country, go to the contacts page of the IBM Software Support Handbook on the Web (techsupport.services.ibm.com/guides/contacts.html) and click the name of your geographic region.

If the problem you submit is for a software defect or for missing or inaccurate documentation, IBM Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Software Support provides a workaround for you to implement until the APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the IBM product support Web pages daily, so that other users who experience the same problem can benefit from the same resolutions.

For more information about problem resolution, see Searching knowledge bases and Obtaining fixes.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information in softcopy form, the photographs and color illustrations might not appear.

Trademarks

IBM, the IBM logo, developerWorks, eServer, iSeries, Lotus, Passport Advantage, pSeries, Rational, Redbooks, Tivoli, the Tivoli logo, Tivoli Enterprise, Tivoli Enterprise Console, NetView, Wake on LAN, AIX, AS/400, DB2, OS/2, OS/400, WebSphere, zSeries are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Lotus is a trademark of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.



Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

A

activity. An operation in an activity plan that is performed on a set of targets on a specific schedule and which can depend upon the execution of other activities. See also activity plan.

activity plan. A set of activities performed on a set of targets on a specified schedule. See also activity.

Activity Plan Editor. A tool used to generate and update an activity plan. See also activity plan.

Activity Plan Monitor. A tool used to submit, monitor, and control the execution of an activity plan. See also activity plan.

admin role. See authorization role.

administrator. See Tivoli administrator.

authorization role. In a Tivoli environment, a role assigned to Tivoli administrators to enable them to perform their assigned systems management tasks. A role may be granted over the entire Tivoli management region or over a specific set of resources, such as those contained in a policy region. Examples of authorization roles include super, senior, admin, and user. See also role.

B

byte-level differencing. The process of detecting the differences, or the delta, between the software package to be installed and the base software package, and creating a delta software package. See also delta install.

C

commit phase. In software distribution, the phase in which previously prepared actions are committed, causing all of the updates to take effect. See also preparation phase.

configuration repository. In a Tivoli environment, a RIM repository that contains information that is collected or generated and stored by inventory scans and software distributions.

corequisite dependency. A condition in which a configuration change can be performed only if other required changes are also performed as part of the specified change sequence.

D

default policy. In a Tivoli environment, a set of resource property values that are assigned to a resource when the resource is created.

delta install. The process of creating the software package that contains only the delta between the base software package and the software package to be installed. By creating and distributing a delta software package, network traffic is reduced.

device. Any non-client, non-server part of a network managed by Tivoli software, including, but not limited to, Palm devices, handheld PCs, cable set-top boxes, and other pervasive devices.

device agent program. A program that resides on a device, functions as a client, and processes jobs as its main task. The device agent program also contains the initial communications data used during enrollment of a device.

device class. A named set of characteristics applied to a group of devices. Each device class has a unique name and represents a device type.

discover. To identify resources within a network environment.

Distribution Status console. An MDist 2 interface provided by Tivoli Management Framework that enables administrators to monitor and control distributions across a network. See also MDist 2.

domain. A user-defined subdivision that is based on groups and hierarchies of components in subsystems. Multiple domains can exist and be nested within or overlap other domains.

E

endpoint. In a Tivoli environment, the agent that is the ultimate recipient for any type of Tivoli operation. See also target.

endpoint list. In a Tivoli environment, a list of all endpoints in a Tivoli management region with their assigned gateways.

exrequisite dependency. A condition in which a configuration change can be performed only if a specified condition does not exist. For example, a software package will be installed on targets only where a package with an exrequisite dependency is not installed.

G

gateway. Software that provides services between the endpoints and the rest of the Tivoli environment.

J

job. A resource consisting of a task and its preconfigured execution parameters. Among other things, the parameters specify the set of hosts on which the job is to execute.

M

managed node. In a Tivoli environment, a computer system on which Tivoli Management Framework is installed. Contrast with endpoint.

managed resource. In a Tivoli environment, a database object that represents a resource and is governed by policies. See policy.

management region. In Tivoli NetView, the set of managed objects on a particular map that defines the extent of the network that is being actively managed. The management region may vary across maps.

management server. In Tivoli Business Systems Manager, a set of Windows NT services that facilitates the processing of event data and user requests.

MDist. A multiplexed distribution service provided by Tivoli Management Framework that enables efficient transfer of data to multiple targets. Another multiplexed distribution service, MDist 2, provides additional management features. See also MDist 2.

MDist 2. A multiplexed distribution service provided by Tivoli Management Framework that enables efficient transfer of data to multiple targets. Administrators can monitor and control a distribution throughout its life cycle. Another multiplexed distribution service, MDist, lacks these management features. See also Distribution Status console.

N

notice. In a Tivoli environment, a message generated by a systems management operation that contains information about an event or the status of an application. Notices are stored in notice groups.

notice group. In a Tivoli environment, an application- or operation-specific container that stores and displays notices that pertain to specific Tivoli functions. The Tivoli bulletin board is comprised of notice groups. A Tivoli administrator can subscribe to one or more notice groups. The administrator's bulletin board contains only the notices that reside in a notice group to which the administrator is subscribed. See also notice.

O

open registration. A registration process in which any user can register their own workstation as a client node with the server.

P

PDA. See personal data assistant.

personal data assistant (PDA). A handheld device that is used for personal organization tasks (such as calendaring, note-taking, and recording telephone and fax numbers), and networking functions such as e-mail and synchronization.

pervasive device. An information appliance from which users can access a broad range of network-based services, including Internet-based e-commerce services.

policy. A set of rules that are applied to managed resources.

policy region. A group of managed resources that share one or more common policies and which model the management or organizational structure of a network computing environment. Administrators use policy regions to group similar resources, to define access to the resources, to control the resources, and to associate rules for governing the resources.

preparation phase. In software distribution, the phase in which each action in a software package prepares the conditions for the successful execution of an install or remove operation. If the preparation phase fails, the target system is returned to its original state. See also commit phase.

profile. In a Tivoli environment, a container for application-specific information about a particular type of resource. A Tivoli application specifies the template for its profiles, which includes information about the resources that can be managed by that Tivoli application.

profile manager. In a Tivoli environment, a container for profiles that links the profiles to a set of resources, called subscribers. Tivoli administrators use profile managers to organize and distribute profiles. A profile manager can operate in the dataless mode or database mode.

Q

query. In a Tivoli environment, a combination of statements that are used to search the configuration repository for systems that meet certain criteria. The query object is created within a query library. See also query library.

query library. In a Tivoli environment, a facility that provides a way to create and manage Tivoli queries. See also *query*.

R

RDBMS Interface Module (RIM). In Tivoli Management Framework, the module in the distributed object database that contains information about the installation of the relational database management system (RDBMS).

reference model. In the context of Tivoli software, the model configuration for a system, or set of systems, that is used to maintain consistent configurations in a distributed environment.

resource. A hardware, software, or data entity that is managed by Tivoli software. See also *managed resource*.

resource manager. An extension of the Tivoli Management Framework that manages resources, such as pervasive devices and users.

RIM. See *RDBMS Interface Module*.

RIM host. In a Tivoli environment, the managed node on which one or more RIM objects is installed. See also *RIM object*.

RIM object. An object that provides the attributes and methods that enable applications to access an RDBMS.

RIM repository. In a Tivoli environment, a relational database that contains information that is collected or generated by Tivoli applications. Examples of a RIM repository include the configuration repository and the event database.

role. A job function that identifies the tasks that a user can perform and the resources to which a user has access. A user can be assigned one or more roles.

S

senior role. See *authorization role*.

software package. In software distribution, a compressed text file that describes the actions to perform on the target system to which it is distributed. Previously known as *file package*. See also *software package definition*.

software package definition. In software distribution, a non-compressed text file that describes the actions to perform on the target system to which it is distributed. Previously known as *file package definition*. See also “*software package*.”

Software Package Editor. In software distribution, the tool that is used for creating and customizing software packages. See also “*software package*.”

stanza. In software distribution, a section of a software package that defines either a specific action to be performed on that the software package or a set of conditions under which actions are to be performed on the software package. The complete software package is a stanza that contains a hierarchy of many different stanzas. See also “*software package*.”

subscriber. In a Tivoli environment, a resource that is subscribed to a profile manager.

subscription. In a Tivoli environment, the process of identifying the subscribers to which profiles will be distributed.

subscription list. In a Tivoli environment, a list that identifies the subscribers to a profile manager. A profile manager can be included in a subscription list in order to subscribe several resources simultaneously rather than adding each resource individually. In Tivoli Plus modules, a profile manager functions as a subscription list.

super role. See *authorization role*.

T

target. The final destination for an action or operation.

target filter. A filter that is defined for each capability set and is based on the attributes defined for a target instance.

task. (1) An activity that has business value, is initiated by a user, and is performed by software. (2) In a Tivoli environment, the definition of an action that must be routinely performed on various managed resources throughout the network. A task defines the executables to be run when the task is executed, the authorization role required to execute the task, and the user or group name under which the task will execute.

task library. In a Tivoli environment, a container in which a Tivoli administrator can create and store tasks and jobs.

Tivoli administrator. In a Tivoli environment, a system administrator who has been authorized to perform systems management tasks and manage policy regions in one or more networks.

Tivoli desktop. In the Tivoli environment, the desktop that system administrators use to manage their network computing environments.

Tivoli environment. The configured Tivoli applications installed in a Tivoli management region. See also *Tivoli management region*.

Tivoli management gateway. In a Tivoli environment, a system that enables bidirectional communication with Tivoli management agents.

Tivoli management region. The Tivoli server and the set of managed node gateways and endpoints that it serves. An organization can have more than one region. A Tivoli management region addresses the physical connectivity of resources whereas a policy region addresses the logical organization of resources.

Tivoli management region role. In a Tivoli environment, the role an administrator has in the local Tivoli management region (region) and any connected region. The region role propagates the assigned authorization level to all resources in the region. For example, if a Tivoli administrator has a senior role in a region, then the administrator has the senior role over every resource in that region.

Tivoli management region server (Tivoli server). The server for a specific Tivoli management region that holds or references the complete set of Tivoli software, including the full object database.

Tivoli server. See Tivoli management region server.

transactional mode. In software distribution, a mode of operation in which install and remove operations occur in two phases: the preparation phase and the commit phase. See also preparation phase and commit phase.

U

undoable mode. In software distribution, a mode of operation in which committed actions can be rolled back because a backup copy was saved.

user. A person who uses Tivoli software and is assigned one or more roles. Any person, organization, process, device, program, protocol, or system that uses a service provided by others.

user role. See authorization role.

V

validation policy. In a Tivoli environment, the policy that ensures that all resources in a policy region comply with the region's established policy. Validation policy prevents Tivoli administrators from creating or modifying resources that do not conform to the policy of the policy region in which the resources were created. Contrast with default policy.

W

Web object. A resource, such as a software package, inventory profile, or reference model, that is available using the Web Interface of IBM Tivoli Configuration Manager.

Index

Special characters

- \$SoftwareDependency
 - Software Distribution parameter in Activity Planner 83
- %variableName
 - Software Distribution parameter in Activity Planner 83

A

- accept
 - Software Distribution operation parameters in Activity Planner 72
- accessibility xvii
- action_entry, element in actions table 156
- actions table in Change Manager 156
- activities
 - canceling 51
 - cloning 22
 - conditioning
 - completion 33
 - depot 33
 - failure 33
 - overview 33
 - success 33
 - targets 33
 - data moving operations 22, 71
 - defining 20
 - deleting 44
 - depot management operations 70
 - elements
 - execution_window 66
 - execution time window 37, 65
 - filtering 47
 - final 21
 - Inventory operations 24
 - pausing 51
 - Pristine Manager
 - defining 25
 - Pristine Manager operations 24
 - removing 44
 - restarting 51
 - resuming 51
 - scheduling 36
 - Software Distribution
 - defining 23, 24
 - Software Distribution operations 70
 - sorting 36
 - supported
 - for Inventory 84
 - for Software Distribution 70
 - targets
 - assigning 26
 - directory query library subscribers 29
 - excluding 27
 - Inventory subscribers 28
 - pervasive device resource groups 30
- activities (continued)
 - targets (continued)
 - pristine 31
 - profile manager subscribers 29
 - profile subscribers 29
 - query library subscribers 29
 - resolving at execution 27, 42
 - resolving at submission 27
 - specifying in file 28
 - specifying in list 27
 - user resource groups 31
 - variables 32
 - task library task 69
- activity
 - activity plan element 61
- activity element
 - application 64
 - condition 65
 - definition 63
 - description 64
 - excl_targets 64
 - execution_window 64
 - FailOnPostScript 65
 - name 64
 - operation 65
 - parameters 65
 - target_resource_type 64
 - targets 64
 - targets_comparison 64
- activity plan
 - Pristine Manager 276
- activity plan status
 - cancel pending 45
 - canceled 45
 - canceled by condition 45
 - failed 45
 - paused 45
 - registered 45
 - started 45
 - starting 45
 - successful 45
 - waiting 45
- Activity Planner
 - accept, Software Distribution operation parameters 72
 - administrative tasks
 - commands for performing 90
 - before you start 5
 - command line interface
 - trace file 136
 - using 57
 - commands
 - for administrative tasks 90
 - for database management 88
 - for managing 88
 - for managing error levels 89
 - for managing return values 89
 - for monitoring 89
 - for registering application plug-ins 90
 - return values 130
- Activity Planner (continued)
 - commit, Software Distribution operation parameters 75
 - Configuration Manager components, enabling for 7
 - core log file 135
 - database
 - commands for managing 88
 - drafts and templates 16
 - DB2 deadlock, managing 138
 - default trace file 137
 - delete, Software Distribution operation parameters 80
 - environment 5
 - error levels
 - commands for managing 89
 - executer trace file 136
 - handler trace file 136
 - install, Software Distribution operation parameters 73
 - installation notes 139
 - Inventory
 - operation parameters 84
 - operations parameter values 85
 - supported activities 84
 - load, Software Distribution operation parameters 77
 - log 135
 - login 19
 - main trace file 136
 - monitor trace file 137
 - multiple domains, managing 139
 - operations 19
 - ExecuteTask 70
 - plug-ins, downloading 137
 - Pristine Manager
 - operation parameters 87
 - operations parameter values 87
 - remove, Software Distribution operation parameters 74
 - retrieve, Software Distribution operation parameters 79
 - return values
 - commands for managing 89
 - roles 6
 - scan, Inventory operation parameters 84
 - send, Software Distribution operation parameters 78
 - Software Distribution
 - operations parameter values 80
 - supported activities 70
 - startup trace files 135
 - trace level
 - setting 131
 - understanding the environment 5
 - undo, Software Distribution operation parameters 76
 - unload, Software Distribution operation parameters 77
 - user roles 5

- activity planner configuration file
 - apm.ini 8
- Activity Planner endpoint variable
 - APM_CLI_TIMEOUT 15
- Activity Planner plug-ins
 - downloading 137
- Activity Planner RIM object
 - traces 131
- Activity Planner server variables 15
 - APM_DEBUG 15
 - APM_HOSTNAME 15
 - APM_KERNEL_LINUX 15
 - APM_PORT 15
 - APM_THREADS_FLAG 15
 - APM_TIMEOUT 15
- activity plans
 - activity element
 - application 64
 - condition 65
 - description 64
 - excl_targets 64
 - execution_window 64
 - FailOnPostScript 65
 - name 64
 - operation 65
 - parameters 65
 - target_resource_type 64
 - targets 64
 - targets_comparison 64
 - authorization roles 49
 - cancel at cutoff option 39
 - canceling 51
 - cleaning up database 54
 - commands
 - for managing 88
 - for monitoring 89
 - complete not after option 39
 - conditioning
 - completion 66
 - depot 67
 - failure 67
 - for_targets 67
 - results 66
 - success 67
 - syntax 66
 - targets 67
 - type_of_condition 67
 - using 66
 - custom variables, using 25
 - database
 - commands for managing 88
 - drafts and templates 16
 - saving to 43
 - definition file 16
 - overview 57
 - structure 57
 - syntax 57
 - deleting 44
 - distribution status console 53
 - draft, saving as 43
 - editor
 - GUI 19
 - overview 16
 - elements
 - activity 61
 - activity_plan root 58
 - activity, definition 63
- activity plans (continued)
 - elements (continued)
 - cancel_at_cutoff 59
 - compl_not_after 59
 - description 58
 - excl_targets 61
 - final_activity 60
 - frequency_information 61
 - mail_id 60
 - name 58
 - notify_date 60
 - post_notice 60
 - priority 58
 - start_not_before 59
 - stop_on_error 60
 - submit_paused 59
 - target_resource_type 60
 - targets 61
 - targets_computation 61
 - variable 61
 - error handling 39
 - frequency information 62
 - frequency_information
 - date_interval 62
 - days_of_month 62
 - days_of_week 62
 - expiration_date 63
 - stop_on_overlap 62
 - stop_rec_on_error_level 62
 - time_interval 62
 - loading 43
 - locked 43
 - managing with command line 88
 - monitor
 - GUI 19
 - overview 16
 - naming 38
 - notification date option 38
 - notification of execution 38
 - pausing 51
 - previewing in Change Manager 189
 - priority 38
 - properties notebook 37
 - properties, defining 37
 - recovering 52
 - recursion
 - error handling 41
 - interrupting 41
 - overview 40
 - removing 44
 - restarting 51
 - resuming 51
 - return codes 40, 41
 - saving 43
 - scheduling 38
 - periodically 40
 - time interval 38
 - specifying targets 42
 - start not before option 39
 - status
 - deleting 53
 - overview 45
 - updating 53
 - stop on error option 39
 - submit paused option 38
 - submitting
 - in Activity Planner 49
- activity plans (continued)
 - submitting (continued)
 - in Change Manager 191
 - supported encoding for xml file 43
 - targets
 - by file 68
 - by name 68
 - definition 68
 - directory query library
 - subscriber 69
 - excluding 27
 - Inventory subscriber 68
 - profile subscriber 68
 - query library subscriber 68
 - resolution 41
 - using variables 69
 - template, saving as 43
 - updating 50
 - updating status 53
 - using with Change Manager 188
 - variables, using 25, 63
 - XML file, saving as 43
- activity_plan root element 58
- adding elements to a reference
 - model 176, 177, 178
- adding operating system element to
 - reference model, Pristine Manager 257
- admin role
 - Activity Planner 7
- ADS certificate
 - Pristine Manager 277
- ADS servers
 - environment variables, Pristine
 - Manager 255
- AllowDefer
 - Software Distribution parameter in
 - Activity Planner 80
- AllowReject
 - Software Distribution parameter in
 - Activity Planner 80
- APM_Admin role 7
- APM_CLI_TIMEOUT
 - Activity Planner endpoint
 - variable 15
- APM_DEBUG
 - Activity Planner server variables 15
- APM_Edit role 7
- APM_HOSTNAME
 - Activity Planner server variables 15
- APM_KERNEL_LINUX
 - Activity Planner server variables 15
- APM_KERNEL_LINUX variable 140
- APM_Manage role 7
- APM_PORT
 - Activity Planner server variables 15
- APM_THREADS_FLAG
 - Activity Planner server variables 15
- APM_THREADS_FLAG variable 140
- APM_TIMEOUT
 - Activity Planner server variables 15
- APM_View role 6, 7
- apm.ini
 - cache_global_target_info
 - parameter 10
 - cache_local_target_info parameter 10
 - cancel_unique_targets 12
 - checkpoint_restart 11

- apm.ini (*continued*)
 - commit_interval 12
 - configuration file 8
 - db_retry_count 11
 - deep_gc_interval parameter 11
 - ENGINE_TUNING section 9
 - executer_max_idle_time
 - parameter 11
 - executer_max_threads parameter 11
 - executer_min_threads parameter 11
 - global_cache_max_size parameter 10
 - global_cache_refresh_timeout
 - parameter 11
 - inventory_rim_name 14
 - load_paused_plans_at_startup
 - parameter 10
 - parameters 8
 - pre_loading_tmf_objects_threshold
 - parameter 10
 - retrieve_gateways_info 12
 - retry_wait_interval 11
 - trace_files_num=3 14
 - use_cached_targets_threshold 12
- apmJNI.ini file
 - apmlog file 135
 - Java Virtual Machine parameters 14
 - JVM memory allocation pool 135
 - logging information 135
 - maxHeapSize 14
 - minHeapSize 14
- application
 - element in activity definition 64
- applying reference models with Web Interface 319
- Argument
 - ExecuteTask
 - in Activity Planner 70
- Assign Event Group to Console
 - dialog 229
- assigning a priority
 - conditioning 148
- assigning pristine subscribers to reference model, Pristine Manager 257
- associating endpoints, resource gateway 355
- attribute
 - in Change Manager dependency
 - element 165
 - in Change Manager element
 - element 165
 - in Change Manager model
 - element 164
 - in Change Manager subscriber
 - element 166
- authorization roles 6
 - activity plans 49
 - admin
 - Activity Planner 7
 - APM_Admin 7
 - APM_Edit 7
 - APM_Manage 7
 - APM_View 6, 7
 - CCM_Admin 153
 - CCM_Edit 152
 - CCM_Manage 152
 - CCM_View 152

- authorization roles (*continued*)
 - queries
 - editing 420
 - running 422
 - RIM_update 6, 7
 - RIM_view 6, 7
 - senior
 - Activity Planner 7
 - super
 - Activity Planner 7
- AutoAccept
 - Software Distribution parameter in
 - Activity Planner 81
- AutoCommit
 - Software Distribution parameter in
 - Activity Planner 81

B

- books
 - see publications xv, xvii
- Boolean operators
 - conditioning by 35, 67

C

- cancel at cutoff option, activity plans 39
- cancel_at_cutoff
 - activity plan element 59
- CCM_Admin role 153
- CCM_Edit role 152
- CCM_Manage role 152
- CCM_View role 152
- change management status 161
- Change Manager
 - and activity plans
 - overview 188
 - previewing 189
 - submitting 191
 - before you start 147
 - checking the configuration file 215
 - command line interface 195
 - commands 195
 - concepts 148
 - conditioning 148
 - confccm.xml file 153, 215
 - backup 153
 - configuration file 153
 - backup 153
 - creating reports 192
 - dependency
 - defining 179
 - desired state
 - setting 178
 - differencing 151
 - elements
 - action_entry 156
 - attribute, in dependency
 - element 165
 - attribute, in element element 165
 - attribute, in model element 164
 - attribute, in subscriber
 - element 166
 - configuration 161
 - current_state 156

- Change Manager (*continued*)
 - elements (*continued*)
 - dependency, in element
 - element 165
 - description 148
 - description, in model element 164
 - desired_state 156
 - desired_state, in dependency
 - element 165
 - desired_state, in element
 - element 164
 - element, in model element 164
 - excluded, in subscriber
 - element 166
 - Inventory 149, 161
 - model 163
 - model, in model element 164
 - modifying 187
 - name 157
 - name, in dependency
 - element 165
 - name, in element element 164
 - name, in model element 163
 - name, in subscriber element 165
 - operating system 149, 161
 - operation 156
 - overview 148
 - parameter 156
 - removing 187
 - Software Distribution 148
 - subscriber 165
 - subscriber, in model element 164
 - subscriber, in subscriber
 - elements 166
 - type, in dependency element 165
 - type, in element element 164
 - type, in subscriber element 165
 - value 157
 - version, in model element 164
 - enabling Configuration Manager
 - components for 153
 - environment 151
 - GUI 171
 - invtable.xml 160
 - login 171
 - logs 213
 - managing multiple domains 219
 - ostable.xml 160
 - overview 147
 - performing operations 152
 - previewing an activity plan 189
 - reference models
 - adding elements 176
 - adding Inventory elements 176, 177
 - adding operating system
 - elements 177
 - adding Software Distribution
 - elements 178
 - assigning subscribers 180
 - concepts 161
 - creating 172
 - creating from target 173
 - example 161
 - exporting 175
 - importing 175
 - managing 195

- Change Manager *(continued)*
 - reference models *(continued)*
 - modifying 186
 - removing 187
 - schematic representation 166
 - XML format 163
 - reports
 - creating 192
 - reference models 193
 - target 193
 - stable.xml file 155
 - states and actions 155
 - submitting an activity plan 191
 - subscribers
 - assigning to a reference model 180
 - CSV 149, 182
 - description 149
 - directory query library 150, 184
 - group 150
 - Inventory 150, 181
 - modifying 188
 - pristine target 150
 - profile manager 150, 182
 - query library 150, 183
 - removing 188
 - static 149, 183
 - Web Interface 150, 180
 - targets
 - modifying 188
 - removing 188
 - traces 213
 - troubleshooting 213
 - understanding the environment 151
 - user roles 148, 152
 - using the GUI 171
- Change Manager GUI
 - starting 171
- checkpoint restart 140
- checkpoint_restart
 - error recovery 140
 - performance improvement 11
- child reference models
 - creating 173
 - overview 148
- cleanup activity plans database 54
- CLI
 - using xix
- cloning activities 22
- cmosep.properties
 - Pristine Manager
 - troubleshooting 273
- CodePageTranslation
 - Software Distribution parameter in Activity Planner 81
- command line
 - syntax xix
 - using xix
- command line interface
 - Pristine Manager 259
- command line interface (CLI) xix
- commands
 - Activity Planner
 - return values 130
 - help for xxi
 - syntax xix
 - using xix
- commands *(continued)*
 - wapmfltr 91
 - wapmgui 94
 - wapmplugin 95
 - wapmrpm 97
 - wccmgui 197
 - wccmplugin 198
 - wcntpln 98
 - using to interrupt an activity plan 41
 - wcrtdirctx 434
 - wcrtdirquery 427
 - wcrtdirlib 426
 - wdelpln 100
 - wdelrmod 200
 - wdelstat 101
 - wexppln 102
 - wexprmod 202
 - wgenpln 103
 - wgetdirctx 436
 - wgetdirquery 431
 - wgeterrlev 106
 - wimppln 107
 - wimprmod 204
 - wlstpln 108
 - wlstrmod 206
 - wlstrep 208
 - wmanagedir 439
 - wmonpln 110
 - woselement 260
 - wpristine 262, 277
 - wpristineexport 264
 - wpristinegroup 265
 - wpristinemachine 267
 - wpristinesrv 270
 - wresetlev 113
 - wresgrp 388
 - wresgw 391
 - wresource 394
 - wrundirquery 432
 - wrunquery 421
 - wsetapmpw 114
 - wsetdirctx 437
 - wsetdirctxpw 438
 - wsetdirquery 429
 - wseterrlev 115
 - using to map return codes 40, 41
 - wsfdpln 117
 - wstartapm 120
 - wstopapm 121
 - wsubpln 122
 - wsyncrmod 210
 - wtrcpm 132
 - wtrccm 217
 - wunlockpln 126
 - wupdpln 127
 - wwweb 288
 - wwwebcfg 303
 - wwwebplugin 283
- commit
 - Software Distribution operation parameters in Activity Planner 75
- compl_not_after
 - activity plan element 59
- complete not after option, activity plans 39
- COMPUTER
 - Inventory table updated by Web Interface 298
- computer name
 - creating machines 239
- condition
 - element in activity definition 65
- conditioning
 - activities 33
 - assigning a priority 148
 - Boolean operators 35, 67
 - Change Manager 148
 - completion 33, 66
 - depot 33, 67
 - failure 33, 67
 - for_targets 67
 - performance improvement 33, 67
 - results 66
 - success 33, 67
 - syntax 66
 - targets
 - all 33, 67
 - single 33, 67
 - type_of_condition 67
 - using 66
- confccm.xml file 153, 215
- configuration element
 - setting priority 189
- configuration elements
 - in Change Manager 161
 - modifying in Change Manager 187
 - removing in Change Manager 187
- configuration file
 - Change Manager, checking 215
- configuring Pristine Manager 224
- connection problems 411
- conventions
 - typeface xviii
- cradle, disabling auto-connection for Palm OS devices 341
- Create Console dialog 229
- creating
 - child reference models 173
 - devices 357
 - directory query library 417
 - dynamic resource group 370, 389
 - groups, Pristine Manager 245
 - machines Environment page, Pristine Manager 241
 - machines General page, Pristine Manager 237
 - machines Network page, Pristine Manager 238
 - machines Tivoli page, Pristine Manager 240
 - machines, Pristine Manager 236
 - operating system elements General page, Pristine Manager 249
 - operating system elements Images page, Pristine Manager 250
 - operating system elements, Pristine Manager 248
 - reference models 172
 - reference models from target 173
 - resources 357
 - servers General page, Pristine Manager 235

- creating (*continued*)
 - servers, Pristine Manager 233
 - static resource group 370
- creating devices
 - endpoint name 358
 - local address 358
 - resource label 358
- creating machines
 - computer name 239
 - endpoint label 238
 - from CLI (Pristine Manager) 269
 - host name 239
 - IP address 239
 - MAC address 238
 - SMBIOS GUID 238
- creating operating system elements
 - from CLI 261
- creating reference model, Change Manager for Pristine Manager 257
- creating servers
 - from CLI (Pristine Manager) 271
- creating WinCE devices
 - local address 359
- CSV subscribers
 - in Change Manager 149, 182
- current states
 - table 157
- current_state, element in actions
 - table 156
- custom variables
 - using in activity plans 25
- customer support
 - see Software Support 446
- customization parameters
 - devices 377
 - Palm OS devices 377
 - Windows CE devices 382
- customizing
 - actions for Resource Manager 375
 - Palm OS devices
 - keywords 377
 - Windows CE devices
 - keywords 382
- customizing devices
 - keyword-value pairs 377
 - software package definition file 377
 - Software Package Editor 377

D

- data moving
 - Activity Planner operations 22
- data moving operations
 - activities 71
 - data moving variables 25
 - Software Distribution variables 25
 - using variables 25
- database tables, Pristine Manager 225
- databases (Pristine Manager)
 - IMAGE table 225
 - MACHINE table 225
 - MACHINE_ENV table 225
 - OSELEMENT table 225
 - OSELEMENT_IMAGE table 225
 - PLUGIN table 225
 - PLUGIN_CREATE_KEYS table 225
 - PLUGIN_ENVIRONMENT table 225

- databases (Pristine Manager) (*continued*)
 - PRISTINE_ACTIVITY table 225
 - PRISTINE_ROLE table 225
 - ROLE_SUBSCRIBER table 225
 - SERVER table 225
 - SERVER_ENVIRONMENT table 225
 - TMR_NAME table 226
- databases, enabling Pristine Manager 224
- date_interval, in frequency information
 - for activity plans 62
- days_of_month, in frequency information
 - for activity plans 62
- days_of_week, in frequency information
 - for activity plans 62
- DB2 schema pristine_db2_schema.sql
 - Pristine Manager 224
- DB2 script file pristine_db2_admin.sql
 - Pristine Manager 224
- Deadline
 - Inventory parameter in Activity Planner 85
 - Software Distribution parameter in Activity Planner 81
- default policy, enabling on resource group 389
- DefaultAction
 - Software Distribution parameter in Activity Planner 81
- DefaultTimeout
 - Software Distribution parameter in Activity Planner 81
- defining
 - environment variables, Pristine Manager 252, 254
- delete
 - Software Distribution operation parameters in Activity Planner 80
- deleting
 - activities 44
 - activity plan 44
 - devices 361
 - elements in Change Manager 187
 - reference models 187
 - resource gateways 411, 412
 - resources
 - from command line 399
 - from Tivoli desktop 361
 - using script 399
 - subscribers in Change Manager 188
 - targets in Change Manager 188
- delstat_commit_interval
 - performance improvement 12
- dependency
 - defining for Software Distribution element 179
 - in Change Manager element element 165
- dependency element
 - in xml format reference models 165
- DependencyCheck
 - Software Distribution parameter in Activity Planner 81
- depot management operations
 - activities 70
- description
 - activity plan element 58

- description (*continued*)
 - element in activity definition 64
 - in Change Manager model element 164
- desired state
 - setting 178
- desired states
 - table 157
- desired_state
 - element in actions table 156
 - in Change Manager dependency element 165
 - in Change Manager element element 164
- DestPath
 - Software Distribution parameter in Activity Planner 81
- DestPostScript
 - Software Distribution parameter in Activity Planner 81
- DestPreScript
 - Software Distribution parameter in Activity Planner 81
- devices
 - creating 357
 - from Tivoli desktop 357
 - customization parameters 377
 - customizing
 - keywords 377
 - using software package definition file 377
 - using Software Package Editor 377
 - deleting 361
 - discovering 356
 - enrolling automatically, from Tivoli desktop 355
 - finding 373
 - grouping 369
 - installing and configuring 331
 - listing 373
 - modifying 360
 - overview 327
 - Palm OS
 - administrative tasks 337
 - changing user ID of 413
 - configuration parameters, changing 337
 - cradle, disabling
 - auto-connection 341
 - installing 331
 - installing, conduit communication with device agent 332
 - installing, configuration file preparation 334
 - installing, device agent files 332
 - installing, planning 332
 - installing, server connection using cradle 331
 - installing, using HotSync Manager 331
 - installing, Web Gateway auto-connection 332
 - modem connection, manual setup 339
 - network attachment option, changing 338

- devices (*continued*)
 - Palm OS (*continued*)
 - PDBGenerator utility setup 336
 - security manager, using 342
 - SSL 341
 - subtypes 359
 - Windows CE
 - administrative tasks 349
 - installing, configuring the device agent 344
 - installing, device agent files 343
 - installing, host PC communications 342
 - installing, overview 342
 - installing, planning 342
 - security manager, using 349
 - SSL 349
 - Dialogs
 - Assign Event Group to Console 229
 - Create Console 229
 - Tivoli Enterprise Console 229
 - differencing, in Change Manager 151
 - directory context, Enterprise Directory 418
 - directory names, notation xix
 - directory query
 - creating 418
 - editing 420
 - running 421
 - directory query library subscribers
 - as targets for activities 29
 - assigning 184
 - in activity plans 69
 - in Change Manager 150
 - DisconnectedOperation
 - Inventory parameter in Activity Planner 85
 - Software Distribution parameter in Activity Planner 81
 - discovering resources
 - from Tivoli desktop 356
 - overview of 356
 - Disposable
 - Software Distribution parameter in Activity Planner 81
 - distribution status console 53
 - DistributionMode
 - Software Distribution parameter in Activity Planner 81
 - DMS_PROXY_HOSTNAME keyword
 - in variables.xml file 411
 - DMSMsgn.log
 - Tivoli Web Gateway 405
 - Web Interface 300
 - DNS problems 411
 - dynamic resolution of targets 41
 - dynamic resource group 369
 - adding resources 372
 - creating 370
- ## E
- editor, activity plans
 - GUI 19
 - overview 16
 - education
 - see Tivoli technical training xviii
 - element
 - in Change Manager model element 164
 - element section
 - in xml format reference models 164
 - elements
 - activity plans
 - activity 61
 - cancel_at_cutoff 59
 - compl_not_after 59
 - description 58
 - excl_targets 61
 - final_activity 60
 - frequency_information 61
 - mail_id 60
 - name 58
 - notify_date 60
 - post_notice 60
 - priority 58
 - start_not_before 59
 - stop_on_error 60
 - submit_paused 59
 - target_resource_type 60
 - targets 61
 - targets_computation 61
 - variable 61
 - modifying in Change Manager 187
 - removing in Change Manager 187
 - Software Distribution 161
 - EnableMulticast
 - Software Distribution parameter in Activity Planner 81
 - EnableTimeout
 - Software Distribution parameter in Activity Planner 81
 - enabling
 - Tivoli Enterprise Console integration 226
 - endpoint label
 - creating machines 238
 - endpoint name
 - creating devices 358
 - endpoint registration
 - wresgw 355
 - endpoints
 - deleting when resource gateways 411, 412
 - registering as resource gateways 355
 - Resource Manager 327
 - roaming 22
 - roaming option 45
 - endpoints on web clients
 - distributing software packages 298, 306
 - TME_OBJECT_LABEL field 298, 306
 - enrolling devices automatically 355
 - Enterprise Directory
 - command line overview 425
 - creating a directory query library 417
 - directory context 418
 - directory query
 - creating 418
 - editing 420
 - running 421
 - Inventory, using with 417
 - Enterprise Directory (*continued*)
 - overview 417
 - Software Distribution, using with 417
 - environment variables
 - adding, Pristine Manager 252, 254
 - ADS servers, Pristine Manager 255
 - installation job (ADS server), Pristine Manager 255
 - response file, Pristine Manager 255
 - running commands, Pristine Manager 256
 - servers, Pristine Manager 235
 - environment variables (Pristine Manager)
 - PRISTINE_GUIRUNONCE 253
 - PRISTINE_COLORDEPTH 252
 - PRISTINE_COMPUTERNAME 252
 - PRISTINE_DNSSERVER_ADDRESS 252
 - PRISTINE_DOMAINNAME 253
 - PRISTINE_ENDPOINT_GWPORT 253
 - PRISTINE_ENDPOINT_OPTIONS 253
 - PRISTINE_ENDPOINT_LABEL 253
 - PRISTINE_ENDPOINT_PORT 253
 - PRISTINE_GATEWAY_ADDRESS 253
 - PRISTINE_HOSTNAME 253
 - PRISTINE_IPADDRESS 253
 - PRISTINE_MAC_ADDRESS 254
 - PRISTINE_SCREENHEIGHT 254
 - PRISTINE_SCREENWIDTH 254
 - PRISTINE_SERVER_GUIRUNONCE 254
 - PRISTINE_SMBIOS_GUID 254
 - PRISTINE_SUBNETMASK 254
 - PRISTINE_TMA_SETUP_PATH 254, 274
 - PRISTINE_TMR_SERVER 254
 - environment variables, notation xix
 - environment variables, Pristine Manager 252
 - error handling
 - in activity plans 39
 - in recursive activity plans 41
 - error recovery
 - checkpoint restart 140
 - excl_targets
 - activity plan element 61
 - element in activity definition 64
 - ExecuteTask
 - Argument, in Activity Planner 70
 - ExecutionMode, in Activity Planner 70
 - Library, in Activity Planner 70
 - OutputFile, in Activity Planner 70
 - OutputFormat, in Activity Planner 70
 - OutputHost, in Activity Planner 70
 - parameters, in Activity Planner 70
 - StageCount, in Activity Planner 70
 - StageInterval, in Activity Planner 70
 - Task, in Activity Planner 70
 - Timeout, in Activity Planner 70
 - execution time window
 - activities 37

- execution time window *(continued)*
 - for activities 65
- execution_window
 - element in activity definition 64
- ExecutionMode
 - ExecuteTask
 - in Activity Planner 70
- ExecutionTimeout
 - Inventory parameter in Activity Planner 85
 - Software Distribution parameter in Activity Planner 81
- ExecutionTimeoutUnit
 - Inventory parameter in Activity Planner 85
 - Software Distribution parameter in Activity Planner 81
- expiration_date, in frequency information for activity plans 63
- exporting a reference model 175
- exporting machine list
 - from command line (Pristine Manager) 269
- exporting machines (Pristine Manager)
 - from CLI 264

F

- FailOnPostScript
 - element in activity definition 65
- failure, activity plan
 - Pristine Manager 278
- failure, avoiding
 - Pristine Manager 277
- failure, installation
 - Pristine Manager 277
- FileName
 - Software Distribution parameter in Activity Planner 81
- files
 - activity plan definition file 57
 - tecad_pristine.baroc file (Pristine Manager) 230
 - tecad_sd.conf 226, 227
 - tecad_sdnew.baroc 226
- Filter keyword 228
- filtering
 - activities 47
 - plans 47
 - pristine machines 242
 - targets 47
- final activity 21
- final_activity
 - activity plan element 60
- finding resources
 - from command line 389
 - from Tivoli Desktop 373
- fixes, obtaining 445
- Force
 - Inventory parameter in Activity Planner 85
 - Software Distribution parameter in Activity Planner 81
- frequency_information
 - activity plan element 61
 - activity plans 62, 63
 - targets_resolution 63

- FromCD
 - Software Distribution parameter in Activity Planner 81
- FromDepot
 - Software Distribution parameter in Activity Planner 81
- FromFileServer
 - Software Distribution parameter in Activity Planner 81

G

- gateway information
 - performance improvement 12
- group
 - subscribers
 - in Change Manager 150
- group manager, Pristine Manager 224
- group subscriber, Pristine Manager 224
- grouping
 - pristine machines 243
- grouping machines
 - from CLI 265
- grouping resources
 - from command line 389
 - from Tivoli desktop 370
 - in dynamic resource group 372
 - in static device group 389
 - in static resource group 266, 371, 389
 - overview of 369
 - using default policy 389
- groups
 - creating, Pristine Manager 245

H

- heap size
 - Java Virtual Machine 14
- help for commands xxi
- host name
 - creating machines 239
- HotSync Manager
 - for Palm OS devices installation 331

I

- IBM agent
 - configuring 344
 - installing 343
- Ignore
 - Software Distribution parameter in Activity Planner 81
- IMAGE table (Pristine Manager) 225
- importing a reference model 175
- information centers, searching to find software problem resolution 445
- Informix schema pristine_infx_schema.sql
 - Pristine Manager 224
- Informix script file
 - pristine_infx_admin.sql
 - Pristine Manager 224
- initialization
 - nokia 362
- install
 - Software Distribution operation parameters in Activity Planner 73

- installation job (ADS server)
 - environment variables, Pristine Manager 255
- installing
 - interconnected regions, Pristine Manager 277
 - Pristine Manager 278
 - software packages with Web Interface 313
 - without share (ADS), Pristine Manager 274
 - without share (RIS), Pristine Manager 274
- installing on pristine machines 223
- concepts 223
- Internet, searching to find software problem resolution 445
- Inventory
 - Activity Planner
 - operation parameters 84
 - operations 24
 - operations parameter values 85
 - supported activities 84
 - Change Manager data elements 149
 - Change Manager scan elements 149
 - COMPUTER table updated by Web Interface 298
 - elements
 - adding to reference models 176, 177
 - data 161
 - scan 161
 - Enterprise Directory, using with 417
 - Resource Manager, using with 353
 - scan operation parameters in Activity Planner 84
 - SD_H_INST table updated by Web Interface 298
 - SD_INST table updated by Web Interface 298
 - subscribers
 - as targets for activities 28
 - in activity plans 68
 - subscribers in Change Manager 181
 - subscribers, in Change Manager 150
 - Web Interface, using with 281
- inventory profiles
 - publishing 286
 - unpublishing 286
- inventory scans
 - running with Web Interface 316
- invtable.xml file in Change Manager 160
- IP address
 - creating machines 239
- IP address, static
 - Pristine Manager 278

J

- java plug-in
 - automatic download 285
- Java Virtual Machine
 - heap size 14
 - memory allocation pool 14
- Java Virtual Machine parameters
 - apmJNI.ini file 14
 - heap size 14

Java Virtual Machine parameters
(continued)

- maxHeapSize 14
- memory allocation pool 14
- minHeapSize 14

JVM heap size 14
JVM memory allocation pool 14

K

keyword, Filter 228

keywords

- customizing

- Windows CE devices 382

- customizing devices

- overview 377

- customizing Palm OS devices 377

keywords (Pristine Manager) 254, 274

keywords (Pristine Manager) 252, 253, 254

- PRISTINE_ENDPOINT_
OPTIONS 253

knowledge bases, searching to find
software problem resolution 445

L

Label

- Software Distribution parameter in
Activity Planner 82

large Change Manager distributions
performance improvement 219

LenientDistribution

- Inventory parameter in Activity
Planner 85

- Software Distribution parameter in
Activity Planner 82

Library

- ExecuteTask

- in Activity Planner 70

Linux version

- troubleshooting 140

listing machines

- from command line (Pristine
Manager) 266, 269

listing operating system elements

- from command line (Pristine
Manager) 261

listing resources

- from command line 389

- from Tivoli desktop 373

listing servers

- from command line (Pristine
Manager) 271

load

- Software Distribution operation
parameters in Activity Planner 77

local address

- creating devices 358

- creating WinCE devices 359

- format, examples of 397

locked activity plans 43

log files

- Activity Planner 135

- Change Manager 213

login, Activity Planner 19

login, Change Manager 171

M

MAC address

- creating machines 238

machine manager, Pristine Manager 223

MACHINE table (Pristine Manager) 225

MACHINE_ENV table (Pristine
Manager) 225

machines

- computer name, creating 239

- creating environment variables,
Pristine Manager 241

- creating for Pristine Manager
(CLI) 269

- creating general, Pristine
Manager 237

- creating network, Pristine
Manager 238

- creating Tivoli, Pristine Manager 240

- creating, Pristine Manager 236

- endpoint label, creating 238

- exporting list (Pristine Manager), from
command line 269

- exporting Pristine Manager (CLI) 264

- grouping for Pristine Manager 265

- host name, creating 239

- IP address, creating 239

- listing (Pristine Manager), from
command line 266, 269

- MAC address, creating 238

- modifying from CLI (Pristine
Manager) 269

- SMBIOS GUID, creating 238

mail_id

- activity plan element 60

manuals

- see publications xv, xvii

maxHeapSize

- apmJNI.ini file 14

memory allocation pool

- Java Virtual Machine 14

Microsoft SQL schema file

- pristine_ms_sql_schema.sql

- Pristine Manager 224

Microsoft SQL script file

- pristine_ms_sql_admin.sql

- Pristine Manager 224

minHeapSize

- apmJNI.ini file 14

MobileEscalateDate

- Inventory parameter in Activity
Planner 85

- Software Distribution parameter in

- Activity Planner 82

MobileEscalateMessage

- Inventory parameter in Activity

- Planner 85

- Software Distribution parameter in

- Activity Planner 82

MobileForceMandatory

- Inventory parameter in Activity

- Planner 85

- Software Distribution parameter in

- Activity Planner 82

MobileHidden

- Inventory parameter in Activity
Planner 85

- Software Distribution parameter in
Activity Planner 82

MobileMandatoryDate

- Inventory parameter in Activity
Planner 86

- Software Distribution parameter in
Activity Planner 82

model

- in Change Manager model
element 164

model element

- in xml format reference models 163

modem connection

- manual setup for Palm OS
devices 339

modifying

- devices 360

- pristine machines, multiple 243

- resources

- from command line 399

- from Tivoli Desktop 360

- sample script for 401

- using script 398

modifying machines

- from CLI (Pristine Manager) 269

modifying operating system elements

- from CLI (Pristine Manager) 261

monitor, activity plans

- GUI 19

- overview 16

msg_cmosep.log

- Pristine Manager

- troubleshooting 273

multiple domains, managing in Change
Manager 219

MVS pristine_db2_mvs_custom_
schema.sql

- Pristine Manager 224

MVS script file pristine_db2_mvs_
admin.sql

- Pristine Manager 224

N

name

- activity plan element 58

- element in actions table 157

- element in activity definition 64

- in Change Manager dependency

- element 165

- in Change Manager element

- element 164

- in Change Manager model

- element 163

- in Change Manager subscriber

- element 165

naming conventions

- special characters 419

network attachment option changing for
Palm OS devices 338

nokia

- connecting to Tivoli Web

- Gateway 362

- nokia (*continued*)
 - creating in Tivoli Resource Manager 362
 - initialization 362
 - provisioning 362
- notation
 - environment variables xix
 - path names xix
 - typeface xix
- notebook
 - activity plan properties 37
- notification date option, activity plans 38
- notification of plan execution 38
- notify_date
 - activity plan element 60
- notify_ext_directly
 - performance improvement 9
- notify_interval
 - performance improvement 10, 35, 67
- NotifyInterval
 - Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 82
- NotifyIntervalUnit
 - Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 82
- number of APM trace files
 - trace_files_num=3 131

O

- online publications
 - accessing xvii
- operating system
 - elements 161
 - adding to reference models 177
 - installing pristine machines, mode 238
- operating system element
 - adding to reference model 257
 - Change Manager 149
- operating system element manager, Pristine Manager 224
- operating system element, Pristine Manager 223
- operating system elements
 - creating for Pristine Manager (CLI) 261
 - creating, Pristine Manager 248, 249, 250
 - listing (Pristine Manager), from command line 261
 - modifying from CLI (Pristine Manager) 261
- operation
 - element in actions table 156
 - element in activity definition 65
- operation parameters
 - Activity Planner
 - supported for Inventory 84
 - supported for Pristine Manager 87

- operations
 - Activity Planner
 - Software Distribution 22
 - parameters 71
- Oracle schema file
 - pristine_ora_schema.sql
 - Pristine Manager 224
- Oracle script file pristine_ora_admin.sql
 - Pristine Manager 224
- ordering publications xvii
- OrigPath
 - Software Distribution parameter in Activity Planner 82
- OrigPostScript
 - Software Distribution parameter in Activity Planner 82
- OrigPreScript
 - Software Distribution parameter in Activity Planner 82
- OSElement
 - Pristine Manager parameter in Activity Planner 87
- OSELEMENT table (Pristine Manager) 225
- OSELEMENT_IMAGE table (Pristine Manager) 225
- ostable.xml file in Change Manager 160
- OutputFile
 - ExecuteTask
 - in Activity Planner 70
- OutputFormat
 - ExecuteTask
 - in Activity Planner 70
- OutputHost
 - ExecuteTask
 - in Activity Planner 70

P

- Palm OS devices
 - administrative tasks 337
 - changing user ID of 413
 - configuration parameters, changing 337
 - cradle, disabling auto-connection 341
 - customizing
 - keyword-value pairs 377
 - installing 331
 - installing, conduit communication with device agent 332
 - installing, device agent files 332
 - installing, planning 332
 - installing, server connection using cradle 331
 - installing, using HotSync Manager 331
 - installing, Web Gateway auto-connection 332
 - modem connection, manual setup 339
 - network attachment option, changing 338
 - PDBGenerator utility setup 336
 - security manager, using 342
 - SSL 341
- Palm OS devices, configuration file preparation 334

- parameter
 - element in actions table 156
- parameter element in stable.xml file 156
- parameters
 - element in activity definition 65
 - Software Distribution operations 71
 - task library operations 69
- parent reference models
 - overview 148
- password
 - tivapm user 139
- path names, notation xix
- pdaemon.log
 - Pristine Manager
 - troubleshooting 273
- PDBGenerator utility setup for Palm OS devices 336
- performance improvement
 - checkpoint restart 140
 - checkpoint_restart 11
 - conditioning 33, 67
 - delstat_commit_interval 12
 - gateway information 12
 - httpd.conf file 409
 - large Change Manager distributions 219
 - notify_ext_directly 9
 - notify_interval 10, 35, 67
 - target resolution 27, 42
 - target variables 32
 - Tivoli Web Gateway configuration 408
 - web server on AIX systems 409
 - web server on Solaris systems 409
- periodically scheduling, activity plans 40
- pervasive devices
 - overview 327
 - resource groups
 - as targets for activities 30
- plans
 - filtering 47
 - targets
 - resolving at submission 42
- plug-ins
 - Activity Planner commands for registering applications 90
- PLUGIN table (Pristine Manager) 225
- PLUGIN_CREATE_KEYS table (Pristine Manager) 225
- PLUGIN_ENVIRONMENT table (Pristine Manager) 225
- pmanager.log
 - Pristine Manager
 - troubleshooting 273
- post_notice
 - activity plan element 60
- prerequisites, Pristine Manager 224
- priority
 - activity plan element 58
 - for activity plans 38
 - setting 189
- Priority
 - Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 82

- priority level 189
- priority set 189
- pristine machines
 - advantages installing 223
 - concepts 223
 - filtering 242
 - grouping 243
 - installing, overview 223
 - modifying multiple 243
- Pristine Manager
 - activity plan 276
 - activity plan failure 278
 - Activity Planner
 - defining 25
 - operation parameters 87
 - operations 24
 - operations parameter values 87
 - adding operating system element to reference model 257
 - ADS certificate 277
 - advantages 223
 - assigning pristine subscribers to reference model 257
 - before you start 233
 - command line interface 259
 - concepts 223
 - configuring 224
 - creating groups 245
 - creating machines 236
 - creating operating system elements 248
 - creating reference model, Change Manager 257
 - creating servers 233
 - database tables 225
 - DB2, pristine_db2_admin.sql script file 224
 - DB2, pristine_db2_schema.sql schema 224
 - enabling databases 224
 - environment variables, adding 252, 254
 - environment variables, ADS servers 255
 - environment variables, installation job (ADS server) 255
 - environment variables, response file 255
 - environment variables, running commands 256
 - failure of installation 277
 - failure, avoiding 277
 - group manager, definition 224
 - group subscriber, definition 224
 - IMAGE table 225
 - Informix, pristine_infx_admin.sql script file 224
 - Informix, pristine_infx_schema.sql schema 224
 - installing 278
 - installing interconnected regions 277
 - installing, without share (ADS) 274
 - installing, without share (RIS) 274
 - integrating with Tivoli Enterprise Console 226
 - machine manager, definition 223
 - MACHINE table 225
- Pristine Manager (*continued*)
 - MACHINE_ENV table 225
 - machines, Environment page 241
 - machines, General page 237
 - machines, Network page 238
 - machines, Tivoli page 240
 - Microsoft SQL,
 - pristine_ms_sql_admin.sql script file 224
 - Microsoft SQL,
 - pristine_ms_sql_schema.sql schema 224
 - MVS, pristine_db2_mvms_admin.sql 224
 - MVS, pristine_db2_mvms_custom_schema.sql 224
 - operating system element manager, definition 224
 - operating system element, definition 223
 - operating system elements, General page 249
 - operating system elements, Images page 250
 - Oracle, pristine_oracle_admin.sql script file 224
 - Oracle, pristine_oracle_schema.sql schema file 224
 - OSELEMENT table 225
 - OSELEMENT_IMAGE table 225
 - overview 223
 - PLUGIN table 225
 - PLUGIN_CREATE_KEYS table 225
 - PLUGIN_ENVIRONMENT table 225
 - prerequisites 224
 - pristine server, definition 223
 - pristine target subscriber, definition 223
 - PRISTINE_ACTIVITY table 225
 - PRISTINE_ROLE table 225
 - reference models
 - subscribing pristine targets 185, 186
 - ROLE_SUBSCRIBER table 225
 - running script for Tivoli Enterprise Console 226
 - schema files 224
 - script files 224
 - server manager, definition 223
 - SERVER table 225
 - SERVER_ENVIRONMENT table 225
 - servers, Environment page 235
 - servers, General page 235
 - static IP address 278
 - submitting plan 256
 - submitting plan, Activity Planner 256, 257
 - subscribers
 - pristine targets to reference model 185, 186
 - Sybase, pristine_syb_admin.sql 224
 - Sybase, pristine_syb_schema.sql 224
 - TMR_NAME table 226
 - trace information 273
 - troubleshooting 273
- Pristine Manager RIM object traces 273
- Pristine Manager tool
 - reference models
 - adding group subscribers 185, 186
 - assigning subscribers 185
 - subscribers
 - assigning group to a reference model 185, 186
 - assigning to a reference model 185
- Pristine Manager troubleshooting
 - cmosep.properties 273
 - msg_cmosep.log 273
 - pdaemon.log 273
 - pmanager.log 273
 - trace_cmosep.log 273
- pristine mode
 - installing operating system 238
- pristine server, Pristine Manager 223
- pristine subscribers
 - assigning to reference model 257
- pristine target subscribers
 - in Change Manager 150
- pristine target subscriber, Pristine Manager 223
- pristine target subscribers
 - targets
 - activities 31
- PRISTINE_ACTIVITY table (Pristine Manager) 225
- PRISTINE_COLORDEPTH, variables (Pristine Manager) 252
- PRISTINE_COMPUTERNAME, environment variables (Pristine Manager) 252
- pristine_db2_admin.sql, Pristine Manager 224
- pristine_db2_mvms_admin.sql, Pristine Manager 224
- pristine_db2_mvms_custom_schema.sql, Pristine Manager 224
- pristine_db2_schema.sql, Pristine Manager 224
- PRISTINE_DNSSERVER_ADDRESS, environment variables (Pristine Manager) 252
- PRISTINE_DOMAINNAME, variables (Pristine Manager) 253
- PRISTINE_ENDPOINT_OPTIONS, environment variables (Pristine Manager) 253
- PRISTINE_ENDPOINT_GWPORT, environment variables (Pristine Manager) 253
- PRISTINE_ENDPOINT_LABEL, environment variables (Pristine Manager) 253
- PRISTINE_ENDPOINT_PORT, environment variables (Pristine Manager) 253
- PRISTINE_GATEWAY_ADDRESS, environment variables (Pristine Manager) 253
- PRISTINE_GUIRUNONCE, variables (Pristine Manager) 253

PRISTINE_HOSTNAME, environment variables (Pristine Manager) 253
 pristine_infx_admin.sql, Pristine Manager 224
 pristine_infx_schema.sql, Pristine Manager 224
 PRISTINE_IPADDRESS, environment variables (Pristine Manager) 253
 PRISTINE_MAC_ADDRESS, environment variables (Pristine Manager) 254
 pristine_ms_sql_admin.sql, Pristine Manager 224
 pristine_ms_sql_schema.sql, Pristine Manager 224
 pristine_oracle_admin.sql, Pristine Manager 224
 pristine_oracle_schema.sql, Pristine Manager 224
 PRISTINE_ROLE table (Pristine Manager) 225
 PRISTINE_SCREENHEIGHT, environment variables (Pristine Manager) 254
 PRISTINE_SCREENWIDTH, environment variables (Pristine Manager) 254
 PRISTINE_SERVER_GUIRUNONCE, environment variables (Pristine Manager) 254
 PRISTINE_SUBNETMASK, environment variables (Pristine Manager) 254
 pristine_syb_admin.sql, Pristine Manager 224
 pristine_syb_schema.sql, Pristine Manager 224
 PRISTINE_TMA_SETUP_PATH, environment variables (Pristine Manager) 254, 274
 PRISTINE_TMR_SERVER, environment variables (Pristine Manager) 254
 problem determination
 describing problem for IBM Software Support 447
 determining business impact for IBM Software Support 447
 submitting problem to IBM Software Support 447
 profile manager subscriber
 in Change Manager 150
 profile manager subscribers
 assigning 182
 targets
 activities 29
 profile subscribers
 in activity plans 68
 targets
 activities 29
 properties
 activity plans 37
 of reference models 172, 175
 publications xv
 accessing online xvii
 ordering xvii
 published_inv_profiles_query 292
 published_packages_query 292
 published_refmods_query 292

Q

queries
 defining 370
 saving results 422
 query library subscribers
 assigning 183
 in activity plans 68
 in Change Manager 150
 targets
 activities 29
 querying resource group 389

R

Reboot
 Software Distribution parameter in Activity Planner 83
 recursion
 submitted activity plans 51
 recursion in activity plans
 interrupting 41
 overview 40
 Recursive
 Software Distribution parameter in Activity Planner 83
 reference model
 adding operating system element 257
 assigning pristine subscribers 257
 reference models
 adding elements 176
 Inventory 176, 177
 operating system 177
 Software Distribution 178
 applying with Web Interface 319
 assigning group subscribers 185, 186
 assigning subscribers 180, 185
 concepts 161
 creating 172
 creating from target 173
 example 161
 exporting 175
 importing 175
 in xml format
 dependency element 165
 element section 164
 model element in Change Manager 163
 subscriber element 165
 managing 195
 modifying in Change Manager 186
 overview 148
 properties 172, 175
 publishing 286
 removing 187
 report in Change Manager 193
 schematic representation 166
 subscribing pristine targets 185, 186
 synchronizing with Web Interface 319
 unpublishing 286
 viewing with Web Interface 317
 XML format 163
 registering resource types 354
 RegisterPervasive.sh 354
 RegisterUser.sh 354

RelativeDeadline
 Inventory parameter in Activity Planner 86
 Software Distribution parameter in Activity Planner 83
 remove
 Software Distribution operation parameters in Activity Planner 74
 removing
 activities 44
 activity plan 44
 elements in Change Manager 187
 reference models 187
 subscribers in Change Manager 188
 targets in Change Manager 188
 reports
 Change Manager, creating 192
 reference models 193
 target 193
 resource gateways
 description 327
 supported applications 327
 resource group
 creating
 dynamic 389
 with script 403
 with Tivoli policy facility 369
 does not exist 413
 dynamic 369, 372
 enabling default policy on 389
 grouping
 from command line 389
 from Tivoli desktop 370
 overview of 369
 pervasive devices, specifying as targets of activities 30
 querying 389
 scripts for subscribers 402
 static 266, 369, 371, 389
 users, specifying as targets of activities 31
 Resource Group Members dialog, greyed out 413
 resource label
 creating devices 358
 Resource Manager
 configuration 327
 customizing actions 375
 database 327
 devices
 installing and configuring 331
 Inventory, using with 353
 overview 327
 resources, types 327
 Software Distribution, using with 353
 software packages for devices 353
 Tivoli gateways 327
 Tivoli servers 327
 troubleshooting 405
 users
 overview 328
 Web server cluster 328
 resource types
 registering 354
 resources
 creating 357

resources (continued)

- from command line 399
 - from Tivoli desktop 357
 - multiple, from file 399
 - with script 398
 - deleting
 - from command line 399
 - from Tivoli desktop 361
 - discovering
 - from Tivoli desktop 356
 - overview of 356
 - finding
 - from command line 389
 - from Tivoli Desktop 373
 - grouping 369
 - from command line 389
 - from Tivoli desktop 370
 - in dynamic resource group 372
 - in static resource group 266, 371, 389
 - overview of 369
 - using default policy 389
 - listing
 - from command line 389
 - from Tivoli desktop 373
 - modifying
 - from command line 399
 - from Tivoli Desktop 360
 - types in Resource Manager 327
 - users 368
- response file
- environment variables, Pristine Manager 255
- retrieve
- Software Distribution operation parameters in Activity Planner 79
- RetryUnicast
- Software Distribution parameter in Activity Planner 83
- return codes for activity plans 40, 41
- return values for Activity Planner commands 130
- rg_def_subscribers
- Tivoli policy facility 369
- rg_val_subscribers
- Tivoli policy facility 369
- RIM_update role 6, 7
- RIM_view role 6, 7
- RIS servers
- MAC address for machines 238
- RoamEp
- Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 83
- roaming endpoints 22, 45
- ROLE_SUBSCRIBER table (Pristine Manager) 225
- ## roles
- Activity Planner 5, 6
 - Activity Planner operations 6
 - admin
 - Activity Planner 7
 - APM_Admin 7
 - APM_Edit 7
 - APM_Manage 7
 - APM_View 6, 7

roles (continued)

- Change Manager 148, 152
- Change Manager operations 152
- RIM_update 6, 7
- RIM_view 6, 7
- senior
 - Activity Planner 7
- super
 - Activity Planner 7
- WebUI_Admin 281

S

- saving activity plans 43
- scan
- Inventory operation parameters in Activity Planner 84
- scheduling
- activities 36
 - activity plans 38
- schema files
- Pristine Manager 224
- script files
- Pristine Manager 224
- scripts
- for creating resources 398
 - for deleting resources 399
 - for modifying resources 398
- samples 401
- for resource group subscribers 402
 - for wresource command to create resource group 403
- writing 370
- SD_H_INST
- Inventory table updated by Web Interface 298
- SD_INST
- Inventory table updated by Web Interface 298
- security manager
- for Palm OS devices 342
 - for Windows CE 349
- send
- Software Distribution operation parameters in Activity Planner 78
- SendTimeout
- Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 83
- SendTimeoutUnit
- Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 83
- senior role
- Activity Planner 7
- server manager, Pristine Manager 223
- SERVER table (Pristine Manager) 225
- SERVER_ENVIRONMENT table (Pristine Manager) 225
- servers
- creating for Pristine Manager (CLI) 271
 - creating general, Pristine Manager 235

servers (continued)

- creating, Pristine Manager 233
 - environment variables, Pristine Manager 235
 - listing (Pristine Manager), from command line 271
 - setting timeout for Pristine Manager (CLI) 263
 - setting trace level for Pristine Manager (CLI) 263
 - setting trace size for Pristine Manager (CLI) 263
 - setting timeout (Pristine Manager) from CLI 263
 - setting trace (Pristine Manager) level, from CLI 263
 - size, from CLI 263
- SMBIOS GUID
- creating machines 238
- Software Distribution
- accept operation parameters in Activity Planner 72
 - activities
 - defining 23
 - Activity Planner
 - defining 24
 - operations 22
 - operations parameter values 80
 - supported activities 70
 - Change Manager
 - elements 148
 - commit operation parameters in Activity Planner 75
 - delete operation parameters in Activity Planner 80
 - elements 161
 - Enterprise Directory, using with 417
 - install operation parameters in Activity Planner 73
 - load operation parameters in Activity Planner 77
 - operations
 - activities 70
 - remove operation parameters in Activity Planner 74
 - Resource Manager, using with 353
 - retrieve operation parameters in Activity Planner 79
 - send operation parameters in Activity Planner 78
 - undo operation parameters in Activity Planner 76
 - unload operation parameters in Activity Planner 77
 - Web Interface, using with 281
- software packages
- distributing to endpoints on web clients 298, 306
 - for devices 353
 - installing with Web Interface 313
 - large distributions 306
 - publishing 286
 - uninstalling with Web Interface 315
 - unpublishing 286
 - verifying with Web Interface 314
- Software Support
- contacting 446

- Software Support (*continued*)
 - describing problem for IBM Software Support 447
 - determining business impact for IBM Software Support 447
 - submitting problem to IBM Software Support 447
- SoftwareDependency
 - Software Distribution parameter in Activity Planner 83
- SoftwarePackage
 - Software Distribution parameter in Activity Planner 83
- sorting activities 36
- SourceHost
 - Software Distribution parameter in Activity Planner 83
- special characters, names of Tivoli resources 419
- SSL
 - for Palm OS devices 341
 - with Windows CE devices 349
- stable.xml file
 - default settings 157
 - in Change Manager 155
 - parameter element 156
 - table_entry element 156
- StageCount
 - ExecuteTask
 - in Activity Planner 70
- StageInterval
 - ExecuteTask
 - in Activity Planner 70
- start not before option, activity plans 39
- start_at, element in execution_window element 66
- start_not_before
 - activity plan element 59
- states, table in Change Manager 157
- static resolution of targets 41
- static resource group 369
 - adding resources 371
 - creating 370
- static subscriber 183
- status, activity plans 45
- stop on error option
 - activity plans 39
 - in recursive activity plans 41
- stop_on_error
 - activity plan element 60
- stop_on_overlap, in frequency
 - information for activity plans 62
- stop_rec_on_error_level, in frequency
 - information for activity plans 62
- submit paused option, activity plans 38
- submit_paused
 - activity plan element 59
- submitted activity plans
 - recursion 51
- submitting
 - activity plans 49
 - an activity plan in Change Manager 191
- submitting plan, Activity Planner for Pristine Manager 256, 257
- submitting plan, Pristine Manager 256

- subscriber
 - in Change Manager model element 164
 - in Change Manager subscriber elements 166
- subscriber element
 - in xml format reference models 165
- subscribers
 - assigning group to reference model 185, 186
 - assigning to reference model 180, 185
 - CSV
 - Change Manager 149
 - description
 - Change Manager 149
 - directory query library
 - as targets for activities 29
 - Change Manager 150
 - group
 - Change Manager 150
 - Inventory
 - as targets for activities 28
 - Change Manager 150
 - modifying in Change Manager 188
 - pristine
 - for activities 31
 - pristine target
 - Change Manager 150
 - for activities 31
 - pristine targets to reference model 185, 186
 - profile
 - as targets for activities 29
 - profile manager
 - as targets for activities 29
 - Change Manager 150
 - query library
 - as targets for activities 29
 - Change Manager 150
 - removing in Change Manager 188
 - static
 - Change Manager 149
 - Web Interface
 - Change Manager 150
 - Web Interface, assigning in Change Manager 180
- subtypes, devices 359
- successful targets 230
- super role
 - Activity Planner 7
- suspend_at, element in execution_window element 66
- Sybase schema pristine_syb_schema.sql
 - Pristine Manager 224
- Sybase script file pristine_syb_admin.sql
 - Pristine Manager 224
- synchronizing
 - reference models with Web Interface 319
- syntax command line xix
- SystemErr.log
 - Tivoli Web Gateway 405
 - Web Interface 300
- SystemOut.log
 - Tivoli Web Gateway 405
 - Web Interface 300

T

- table_entry element in stable.xml file 156
- target variables
 - \$(DEPOT_LIST) 32
 - \$(ORIGINATOR_LIST) 32
 - \$(TARGET_LIST) 32
 - performance improvement 32
- target_resource_type
 - activity plan element 60
 - element in activity definition 64
- targets
 - activity plan element 61
 - activity plans
 - by file 68
 - by name 68
 - directory query library
 - subscriber 69
 - Inventory subscriber 68
 - profile subscriber 68
 - query library subscriber 68
 - using variables 69
- assigning for activities 26
- canceling 51
- conditioning in activities 33
- definition in activity plans 68
- directory query library
 - subscribers for activities 29
- dynamic resolution 41
- element in activity definition 64
- excluding from
 - activities 27
 - activity plans 27
- file specification
 - for activities 28
- filtering 47
- Inventory subscribers
 - for activities 28
- list specification
 - for activities 27
- modifying in Change Manager 188
- pausing 51
- pervasive device resource groups
 - for activities 30
- pristine
 - for activities 31
- profile manager subscribers
 - for activities 29
- profile subscribers
 - for activities 29
- query library subscribers
 - for activities 29
- removing in Change Manager 188
- report in Change Manager 193
- resolution 41
- resolving at activity execution
 - performance improvement 27, 42
- resolving at plan submission
 - performance improvement 27, 42
- restarting 51
- resuming 51
- specification in activity plans 42
- static resolution 41
- user resource groups
 - for activities 31
- variables
 - built-in, for activities 32

- targets (*continued*)
 - variables (*continued*)
 - using for activities 32
- targets_comparison
 - element in activity definition 64
- targets_computation
 - activity plan element 61
- targets_resolution 63
- targets_resolution, in frequency information for activity plans 63
- Task
 - ExecuteTask
 - in Activity Planner 70
- task library operations parameters
 - values 70
- task library parameters 69
- task library tasks 22
- task-type activity 69
- tecad_pristine.baroc file (Pristine Manager) 230
- tecad_sd.conf file 226, 227
- tecad_sdnew.baroc file 226
- time interval scheduling, activity plans 38
- time_interval, in frequency information for activity plans 62
- Timeout
 - ExecuteTask
 - in Activity Planner 70
- tivapm
 - password 139
- Tivoli Enterprise Console
 - assigning administration roles 229
 - assigning event groups 229
 - assigning operators 229
 - configuring Pristine Manager for 226
 - creating a console 229
 - dialog 229
 - running script, Pristine Manager 226
 - tecad_sd.conf file 227
- Tivoli gateways
 - Resource Manager 327
- Tivoli policy facility
 - rg_def_subscribers 369
 - rg_val_subscribers 369
- Tivoli policy facility, resource group 369
- Tivoli query facility, defining queries 370
- Tivoli servers
 - Resource Manager 327
- Tivoli software information center xvii
- Tivoli technical training xviii
- Tivoli Web Gateway
 - DMSMsgn.log 405
 - SystemErr.log 405
 - SystemOut.log 405
 - TraceDMSn.log 405
 - TWGapi.log 405
- Tivoli Web Gateway configuration
 - performance improvement 408
 - traceConfig.properties file 408
 - Transaction.properties file 407
- TME_OBJECT_LABEL field
 - endpoints on web clients 298, 306
- TMR_NAME table (Pristine Manager) 226

- trace_cmosep.log
 - Pristine Manager
 - troubleshooting 273
- trace_files_num=3
 - number of APM trace files 131
- traceConfig.properties file
 - Tivoli Web Gateway
 - configuration 408
- TraceDMSn.log
 - Tivoli Web Gateway 405
 - Web Interface 300
- traces
 - Activity Planner RIM object 131
 - Change Manager 213
 - Pristine Manager 273
 - Pristine Manager RIM object 273
 - Tivoli Resource Manager 405
- training, Tivoli technical xviii
- Transaction.properties file
 - Tivoli Web Gateway
 - configuration 407
- Transactional
 - Software Distribution parameter in Activity Planner 83
- troubleshooting
 - Pristine Manager 273
- TWGapi.log
 - Tivoli Web Gateway 405
 - Web Interface 300
- type
 - in Change Manager dependency element 165
 - in Change Manager element element 164
 - in Change Manager subscriber element 165, 166
- typeface conventions xviii

U

- undo
 - Software Distribution operation parameters in Activity Planner 76
- Undo
 - Software Distribution parameter in Activity Planner 83
- uninstalling
 - software packages with Web Interface 315
- United Linux
 - Activity Planner troubleshooting 140
- unload
 - Software Distribution operation parameters in Activity Planner 77
- user roles
 - Activity Planner 5
 - Change Manager 148, 152
- UserNotification
 - Software Distribution parameter in Activity Planner 83
- users
 - resource groups
 - as targets for activities 31
 - Resource Manager, overview 328
 - resources
 - grouping 370
 - listing 373

- users (*continued*)
 - resources (*continued*)
 - viewing 368
 - viewing 368

V

- value
 - element in actions table 157
- variable
 - activity plan element 61
- variableName
 - Software Distribution parameter in Activity Planner 83
- variables
 - Activity Planner 15
 - as targets for activities 32
 - built-in
 - in targets for activities 32
 - custom, using in activity plans 25
 - using for targets in activity plans 69
 - using in activity plans 25, 63
- variables, notation for xix
- variables.xml file 411
- verifying
 - software packages with Web Interface 314
- version
 - in Change Manager model element 164
- viewing
 - reference models with Web Interface 317

W

- WakeOnLan
 - Inventory parameter in Activity Planner 86
 - Software Distribution parameter in Activity Planner 83
- wapmfltr 91
- wapmgui 94
- wapmplugin 95
- wapmrin 97
- wccmgui 197
- wccmplugin 198
- wcntpln 98
 - using to interrupt an activity plan 41
- wcrtdirctx 434
- wcrtdirquery 427
- wcrtdirlib 426
- wdelpln 100
- wdelrmod 200
- wdelstat 101
- Web Gateway
 - database 327
 - resource gateway 327
 - servers 328
- web interface
 - customizing
 - webconsole.properties file 311
 - unrestricted environment 292
- Web Interface
 - authorization error 305
 - configuring trace 302

- Web Interface *(continued)*
 - distributing to endpoints on web clients 306
 - distributing to many users 306
 - DMSMsgn.log 300
 - introduction 281
 - Inventory, using with 281
 - logon unsuccessful 307
 - monitoring results 298
 - profile not found error 305
 - published_inv_profiles_query 292
 - published_packages_query 292
 - published_refmods_query 292
 - reference model cannot be applied 307
 - registering plug-ins 282
 - results not returned 305
 - roles 281
 - running inventory scans 316
 - setting up 281
 - Software Distribution, using with 281
 - software package not visible 307
 - starting 309
 - subscribers, assigning in Change Manager 180
 - subscribers, in Change Manager 150
 - SystemErr.log 300
 - SystemOut.log 300
 - TraceDMSn.log 300
 - transaction timeout 306
 - troubleshooting 305
 - troubleshooting for
 - administrator 300
 - troubleshooting for users 321
 - TWGapi.log 300
 - updating COMPUTER Inventory table 298
 - updating SD_H_INST Inventory table 298
 - updating SD_INST Inventory table 298
 - user problems 306
 - using 309
 - Web objects not visible 306
 - webui_query 292
- web objects
 - unrestricted access 292
- Web objects
 - determining what has been published 292
 - publishing 285, 286
 - unpublishing 285, 286
 - using 312
- Web user
 - providing information and assistance 294
- webconsole.properties file
 - customizing 311
- WebUI_Admin role 281
- webui_query 292
- wexppln 102
- wexprmod 202
- wgenpln 103
- wgetdirctx 436
- wgetdirquery 431
- wgeterrlev 106
- wimppln 107
- wimprmod 204
- WinCE devices
 - local address 359
- Windows CE devices
 - administrative tasks 349
 - customizing
 - keyword-value pairs 382
 - installing, configuring the device agent 344
 - installing, configuring the IBM agent 344
 - installing, device agent files 343
 - installing, host PC communications 342
 - installing, overview 342
 - installing, planning 342
 - security manager, using 349
 - SSL 349
- wlstpln 108
- wlstrmod 206
- wlstsrep 208
- wmanagedir 439
- wmonpln 110
- woselement 260
- wpristine 262, 277
- wpristineexport 264
- wpristinegroup 265
- wpristinemachine 267
- wpristinesrr 270
- wresetlev 113
- wresgrp 388
- wresgw 391
 - endpoint registration 355
- wresource 394
- wresource action command
 - script 375
- wrundirquery 432
- wsetapmpw 114
- wsetdirctx 437
- wsetdirctxpw 438
- wsetdirquery 429
- wseterrlev 115
 - using to map return codes 40, 41
- wsfdpln 117
- wstartapm 120
- wstopapm 121
- wsubpln 122
- wsyncrmod 210
- wtrcapm 132
- wtrcccm 217
- wunlockpln 126
- wupdpln 127
- wwweb 288
- wwwebcfg 303
- wwwebplugin 283

X

- xml file
 - activity plans definition file 16, 57
 - supported encodings 43



Program Number: 5724-C06

Printed in USA

SC23-4710-05

