

PATROL[®] Adapter for WBEM User Guide

Version 1.2.00

June 16, 2000



Copyright © 1998-2000 BMC Software, Inc., as an unpublished work. All rights reserved.

BMC Software, the BMC Software logos, and all other BMC Software product or service names are registered trademarks or trademarks of BMC Software, Inc. PATROL SCRIPT LANGUAGE IS PROPRIETARY AND CONFIDENTIAL TO BMC SOFTWARE, INC.

PATROL® technology has been issued U.S. Patent No. 5,655,081.

Acrobat is a registered trademark of Adobe Systems, Inc. All other third-party logos and product/trade names are trademarks or registered trademarks of their respective companies.

Restricted Rights Legend

U.S. GOVERNMENT RESTRICTED RIGHTS. UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in FAR Section 52.227-14 Alt. III (g)(3), FAR Section 52.227-19, DFARS 252.227-7014 (b), or DFARS 227.7202, as amended from time to time. Send any contract notices to Contractor/Manufacturer:

BMC Software, Inc.
2101 CityWest Blvd.
Houston TX 77042-2827
USA

Contacting BMC Software

You can access the BMC Software Web site at <http://www.bmc.com>. From this Web site, you can obtain general information about the company, its products, special events, and career opportunities. For a complete list of all BMC Software offices and locations, go to <http://www.bmc.com/corporate/offices.html>.

USA and Canada

Address BMC Software, Inc.
2101 CityWest Blvd.
Houston TX 77042-2827

Telephone 713 918 8800 or
800 841 2031

Fax 713 918 8000

Outside USA and Canada

Telephone (01) 713 918 8800

Fax (01) 713 918 8000

Customer Support

You can obtain technical support by using Response Online™ (support from the Web) or Response On Demand™. To expedite your inquiry, please see “Before Contacting BMC Software,” below.

Response Online

You can obtain technical support from BMC Software 24 hours a day, seven days a week by accessing the technical support Web site at <http://www.bmc.com/support.html>. From this site, you can

- read overviews about support services and programs that BMC Software offers
- find the most current information about BMC Software products
- search a database for problems similar to yours and possible solutions
- order or download product documentation
- report a problem or ask a question
- subscribe to receive e-mail notices when new product versions are released
- find worldwide BMC Software support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Response On Demand

In the USA and Canada, if you need technical support and do not have access to the Web, call 800 537 1813. Outside the USA and Canada, please contact your local support center or your local sales office for assistance.

Before Contacting BMC Software

Before you contact BMC Software, have the following information available so that a technical support analyst can begin working on your problem immediately:

- product information
 - product name
 - product version (release number)
 - license number and password (trial or permanent)
- operating-system and environment information
 - machine type
 - operating system type, version, and service pack or program temporary fix (PTF)
 - system hardware configuration
 - serial numbers
 - related software (database, application, and communication) including type, version, and service pack or PTF
- sequence of events leading to the problem
- commands and options that you used
- messages received (and the time and date that you received them)
 - product error messages
 - messages from the operating system, such as `file system full`
 - messages from related software

Contents

About This Book	xiii
Chapter 1	Introduction
Overview	1-2
System Requirements	1-4
Components	1-4
Installation Information	1-5
Files Installed	1-6
Classes Installed	1-11
Installation of the PATROL_Agent Instance	1-12
Installation of the WBEM KM on a Unix Computer	1-13
Chapter 2	PATROL Models for CIM
Overview	2-2
UML Model Diagrams	2-2
CIM Namespaces	2-3
PATROL Agent Model	2-5
PATROL Agent Classes	2-7
PATROL Agent Associations	2-7
Methods	2-8
PATROL Knowledge Module Model	2-16
PATROL KM Classes	2-19
PATROL KM Associations	2-20
PATROL Event Model	2-21
MOF Files	2-22
Chapter 3	PATROL Providers
Overview	3-2

PATROL Agent Connections	3-3
Error Logging	3-3
PATROL Instance Provider	3-4
Improving Performance	3-4
PATROL Event Provider	3-5
PATROL Method Provider	3-6

Chapter 4

PATROL KM2MOF Tool

Overview	4-2
Using the PATROL KM2MOF Tool	4-4
Command Line Options	4-4
Compiling a .km File	4-5
Compiling a .kml File	4-6
Loading .mof Files into CIMOM	4-7
PATROL KM Properties Mapped to MOF Files	4-7
Messages and Errors	4-10

Chapter 5

PATROL Knowledge Module for WBEM

Overview	5-3
Loading the PATROL Adapter for WBEM KM	5-4
PATROL Adapter for WBEM KM Application Classes	5-5
PATROL WBEM KM Hierarchy	5-6
About the PATROL Adapter for WBEM Application Classes	5-7
WBEM Application Class	5-8
Parameters	5-8
Menu Commands	5-8
WBEM_CIMOM Application Class	5-9
Parameters	5-9
Menu Commands	5-9
WBEM_NAMESPACE Application Class	5-11
Parameters	5-11
Menu Commands	5-11
WBEM_PROVIDER Application Class	5-12
Parameters	5-13
Menu Commands	5-13
WBEM_PROCESS Application Class	5-14
Parameters	5-14
ProcessColl Parameter	5-15
PageFaultsPerSec Parameter	5-16
PageFileBytes Parameter	5-16

PrivTimePercent Parameter	5-17
ProcessorTimePercent Parameter	5-18
ThreadCount Parameter	5-18
UserTimePercent Parameter	5-19
WorkingSet Parameter	5-20
Menu Commands	5-21
Error Messages	5-21

Appendix A

Example Scenarios

PATROL KM Model Example	A-2
PATROL Events Example	A-6
PATROL Agent Model Example	A-12
PATROL KM Development Example	A-15

Appendix B

Troubleshooting

Possible Problems and Solutions	B-2
---	-----

Appendix C

Security

Firewalls	C-2
Passwords	C-2
COM/DCOM Configuration	C-3
PATROL Single Sign-On	C-3
Support for PATROL Single Sign-On	C-3
Changing the Account for the PEM COM Server	C-4
Terminating the PEM COM Server Process	C-6

Appendix D

Registry Settings

Registry Settings for the PATROL Adapter for WBEM	D-2
---	-----

Glossary

Index

Figures

Figure 1-1	CIMOM and the PATROL Adapter for WBEM	1-3
Figure 2-1	PATROL Models Namespace Structure	2-4
Figure 2-2	PATROL Agent Model Diagram	2-6
Figure 2-3	PATROL Knowledge Module Model Diagram	2-17
Figure 2-4	Example PATROL KM Model Using the Oracle KM	2-18
Figure 2-5	PATROL Event Model Diagram	2-21
Figure 3-1	PATROL Provider Interaction Diagram	3-2
Figure 4-1	PATROL KM2MOF Tool Flowchart	4-3
Figure 5-1	WBEM KM Application Class Hierarchy	5-6

Tables

Table 1-1	System Requirements	1-4
Table 1-2	PATROL Adapter for WBEM Components	1-4
Table 1-3	Installed Files	1-6
Table 1-4	Installed Classes	1-11
Table 1-5	Namespace Editing Tools	1-12
Table 1-6	Installed KM Files	1-13
Table 2-1	Diagram Elements	2-2
Table 2-2	CIM Namespaces	2-3
Table 2-3	PATROL Agent Classes	2-7
Table 2-4	PATROL Agent Associations	2-7
Table 2-5	PATROL_Agent::Discover Parameters	2-8
Table 2-6	PATROL_Agent::EncryptPassword Parameters	2-9
Table 2-7	PATROL_Agent::GetVariable Parameters	2-10
Table 2-8	PATROL_Agent::UpdateProvider Parameters	2-10
Table 2-9	PATROL_Agent::Disconnect Parameters	2-11
Table 2-10	PATROL_Agent::AuthenticateOnConnect Parameters	2-11
Table 2-11	PATROL_Event::GetEventHistory Parameters	2-12
Table 2-12	PATROL_Parameter::GetParamAlarmRanges Parameters	2-14
Table 2-13	PATROL_Parameter::GetParamAnnotation Parameters	2-14
Table 2-14	PATROL_Parameter::GetParamHistory Parameters	2-15
Table 2-15	PATROL KM Model Classes	2-19
Table 2-16	PATROL KM Model Associations	2-20
Table 2-17	MOF Files	2-22
Table 3-1	Generated Events	3-6
Table 4-1	Command Line Options	4-5
Table 4-2	PATROL KM Properties Mapped to MOF	4-7
Table 4-3	PATROL KM2MOF Tool Messages	4-10
Table 5-1	PATROL Adapter for WBEM Application Classes	5-7

Table 5-2	WBEM Application Class Menu Summary	5-8
Table 5-3	WBEM_CIMOM Application Class Menu Summary	5-10
Table 5-4	WBEM_NAMESPACE Application Class Menu Summary	5-12
Table 5-5	WBEM_PROVIDER Application Class Menu Summary	5-13
Table 5-6	WBEM_PROCESS Application Class Parameter Summary	5-14
Table 5-7	ProcessColl Parameter Defaults	5-15
Table 5-8	PageFaultsPerSec Parameter Defaults	5-16
Table 5-9	PageFileBytes Parameter Defaults	5-16
Table 5-10	PrivTimePercent Parameter Defaults	5-17
Table 5-11	ProcessorTimePercent Parameter Defaults.	5-18
Table 5-12	ThreadCount Parameter Defaults	5-18
Table 5-13	UserTimePercent Parameter Defaults.	5-19
Table 5-14	WorkingSet Parameter Defaults	5-20
Table 5-15	WBEM_PROCESS Application Menu Summary	5-21
Table 5-16	PATROL Adapter for WBEM KM Error Messages	5-21
Table B-1	Possible Problems and Solutions	B-2
Table D-1	Registry Settings.	D-2

About This Book

This book contains detailed information about PATROL[®] Adapter for WBEM and is intended for administrators, programmers, and PATROL Knowledge Module developers using PATROL and Microsoft WMI in a Windows 2000 (or Windows NT) environment. Use this book with the appropriate PATROL user guide for your console.

This book assumes that you are familiar with the following products and related documentation:

- Microsoft Windows Management Instrumentation (WMI) Software Developer Kit (SDK)
- PATROL[®] Agent and PATROL[®] Console documentation
- PATROL[®] Knowledge Module[™] documentation
- Distributed Management Task Force (DMTF) Common Information Model (CIM) Version 2.2 (for more information, see <http://www.dmtf.org>)

Note

This book assumes that you are familiar with your host operating system. You should also know how to perform basic actions, such as choosing menu commands and dragging and dropping icons, in a window environment.

How This Book Is Organized

This book is organized as follows. In addition, a glossary of terms and an index appear at the end of the book.

Chapter/Appendix	Description
Chapter 1, "Introduction"	Provides an overview of the features and components of the PATROL Adapter for WBEM.
Chapter 2, "PATROL Models for CIM"	Provides an overview of the PATROL Models for CIM.
Chapter 3, "PATROL Providers"	Defines the PATROL Providers for WBEM that are included with the PATROL Adapter for WBEM.
Chapter 4, "PATROL KM2MOF Tool"	Explains the KM2MOF Tool and how to use it.
Chapter 5, "PATROL Knowledge Module for WBEM"	Provides an overview of the PATROL Adapter for WBEM Knowledge Module.
Appendix A, "Example Scenarios"	Describes example scenarios for using the PATROL Adapter for WBEM.
Appendix B, "Troubleshooting"	Describes possible problems that you could encounter while using the PATROL Adapter for WBEM. Solutions to the problems are provided.
Appendix C, "Security"	Describes the security interface for the PATROL Adapter for WBEM and its configuration.
Appendix D, "Registry Settings"	Defines the registry settings for use with the PATROL Adapter for WBEM.

Related Documentation

BMC Software products offer several types of documentation:

- online and printed books
- online Help
- release notes

Online and Printed Books

The books that accompany BMC Software products are available in online format and printed format. You can view online books with Acrobat Reader from Adobe Systems. The reader is provided at no cost, as explained in the “To Access Online Books” section. You can also obtain additional printed books from BMC Software, as explained in the “To Request Additional Printed Books” section.

To Access Online Books

Online books are formatted as Portable Document Format (PDF) files. You can view them, print them, or copy them to your computer by using Acrobat Reader 3.0 or later. You can access online books from the documentation compact disc (CD) that accompanies your product or from the World Wide Web.

In some cases, installation of Acrobat Reader and downloading the online books is an optional part of the product-installation process. For information about downloading the free reader from the Web, go to the Adobe Systems site at <http://www.adobe.com>.

To view any online book that BMC Software offers, visit the support page of the BMC Software Web site at <http://www.bmc.com/support.html>. Log on and select a product to access the related documentation. (To log on, first-time users can request a user name and password by registering at the support page or by contacting a BMC Software sales representative.)

To Request Additional Printed Books

BMC Software provides a core set of printed books with your product order. To request additional books, go to the Web site, <http://www.bmc.com/support.html>.

Release Notes

Printed release notes accompany each BMC Software product. Release notes provide up-to-date information such as

- updates to the installation instructions
- last-minute product information

The latest versions of the release notes are also available on the Web at <http://www.bmc.com/support>.

Conventions

The following conventions are used in this book:

- This book includes special elements called *notes*, *warnings*, *examples*, and *tips*:

Note

Notes provide additional information about the current subject.

Warning

Warnings alert you to situations that can cause problems, such as loss of data, if you do not follow instructions carefully.

Example

An example clarifies a concept discussed in text.

Tip

A tip provides useful information that may improve product performance or make procedures easier to follow.

- All syntax, operating system terms, and literal examples are presented in Courier typeface.
- In instructions, **boldface** type highlights information that you enter. File names, directories, and Web addresses also appear in boldface type.
- The symbol => connects items in a menu sequence. For example, **Actions => Create Test** instructs you to choose the Create Test command from the Actions menu.
- The symbol >> denotes one-step instructions.
- In syntax, path names, or system messages, *italic* text represents a variable, as shown in the following examples:

The table *table_name* is not available.

system/instance/file_name

- In syntax, the following additional conventions apply:
 - A vertical bar (|) separating items indicates that you must choose one item. In the following example, you would choose *a*, *b*, or *c*:

a | b | c
 - An ellipsis (. . .) indicates that you can repeat the preceding item or items as many times as necessary.
 - Square brackets ([]) around an item indicate that the item is optional.

Introduction

This chapter provides you with an overview of the PATROL Adapter for WBEM product.

The following topics are discussed:

Overview	1-2
System Requirements	1-4
Components	1-4
Installation Information	1-5
Files Installed	1-6
Classes Installed	1-11
Installation of the PATROL_Agent Instance	1-12
Installation of the WBEM KM on a Unix Computer	1-13

Overview

The Web-Based Enterprise Management (WBEM) initiative consists of several standards developed by the Distributed Management Task Force (DMTF) to describe and share enterprise management information across platforms. The Common Information Model (CIM) standard describes the underlying, object-oriented structure of the management information. For more information, see the Web site, <http://www.dmtf.org>.

Microsoft refers to its implementation of WBEM as Windows Management Instrumentation (WMI). WMI includes a central component called the CIM Object Manager (CIMOM), which gathers and manipulates information about system resources represented as CIM classes and instances.

WMI uses components called *providers* to gather information and make it available to management applications. The applications request information from CIMOM, which retrieves static data from its repository or dynamic data from a provider.

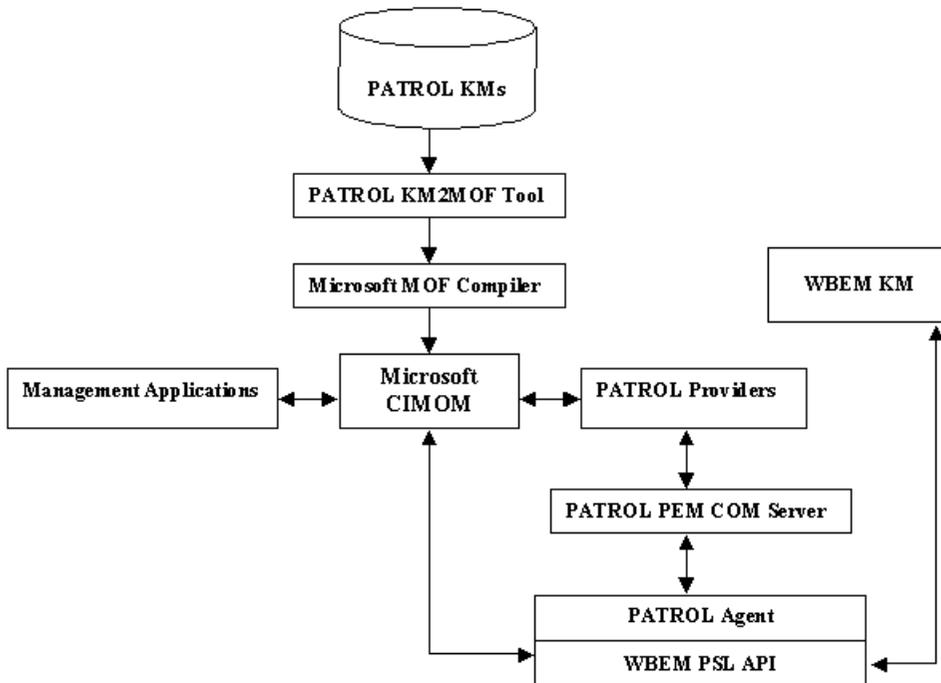
The PATROL Adapter for WBEM product supplies

- a PSL interface to allow PATROL Agents to access WMI data
- WMI providers to allow WMI clients access to PATROL data
- a Knowledge Module (KM) for the PATROL Adapter for WBEM, which monitors WMI activity

Figure 1-1 on page 1-3 illustrates the integration of the PATROL Adapter for WBEM product and CIMOM.

For specific examples of using the PATROL Adapter for WBEM, refer to Appendix A, “Example Scenarios.”

Figure 1-1 CIMOM and the PATROL Adapter for WBEM



CIM data is stored in files using the Managed Object Format (MOF). The PATROL KM2MOF Tool reads PATROL KMs or KMLs and generates a corresponding MOF file. The contents of this file can be compiled into the CIM repository using the Microsoft WMI MOF Compiler. For more information about using these tools, see Chapter 4, “PATROL KM2MOF Tool.”

Providers act as intermediaries between CIMOM and managed objects. When CIMOM receives a request from a management application for data that is not in its repository, it forwards the request to a provider. The PATROL[®] Event Manager (PEM) COM Server allows clients, like the PATROL Providers, to subscribe to PATROL events and retrieve PATROL data. For more information on PATROL Providers, see Chapter 3, “PATROL Providers.” The PATROL Adapter for WBEM KM uses the WBEM PSL API to monitor WMI activity. See Chapter 5, “PATROL Knowledge Module for WBEM” for more information about this KM.

System Requirements

To use the PATROL Adapter for WBEM, you must have the items listed in Table 1-1 installed on your system.

Table 1-1 System Requirements

Component	Version
Windows 2000 or Windows NT	Windows NT version 4.0 for Intel x86 with Service Pack 4 or greater
PATROL for Windows 2000 or PATROL for Windows NT	3.2 or greater
Microsoft WMI core components	build 1085 ¹

¹ This build is included in Windows 2000. For Windows NT, you can install this build with the PATROL Adapter for WBEM installation.

Warning

If you intend to use the PATROL Adapter for WBEM with Microsoft Systems Management Server (SMS), use SMS version 2.0 with Service Pack 2 or later. Earlier versions are not compatible with build 1085 of the WMI core components.

Components

The key components listed in Table 1-2 are included with the PATROL Adapter for WBEM.

Table 1-2 PATROL Adapter for WBEM Components (Part 1 of 2)

Component	Description	For More Information
models or schemas	.mof files imported into CIMOM	Chapter 2, "PATROL Models for CIM"
providers	registered PATROL Instance Provider, Event Provider and Method Provider	Chapter 3, "PATROL Providers"

Table 1-2 PATROL Adapter for WBEM Components (Part 2 of 2)

Component	Description	For More Information
KM2MOF Tool	tool to compile .km files into .mof format	Chapter 4, "PATROL KM2MOF Tool"
PATROL KM for WBEM	KM used in PATROL Console to retrieve data from CIMOM and to configure the PATROL providers	Chapter 5, "PATROL Knowledge Module for WBEM"
WMI Core components	WMI components used by the PATROL Adapter for WBEM They are included as an option when installing the PATROL Agent and other PATROL applications.	http://msdn.microsoft.com
PSL API	PSL library files	<i>PATROL Adapter for WBEM Reference Manual</i>
examples	WBEM-enabled PATROL KM and PATROL-enabled WBEM application examples that retrieve PATROL data	Appendix A, "Example Scenarios"
documentation	user guide, reference manual, and release notes for the PATROL Adapter for WBEM in Adobe Acrobat reader format	http://www.bmc.com

Installation Information

The PATROL Adapter for WBEM is installed as a component of the PATROL Agent and various other PATROL products. For information on the installation of PATROL products, refer to the *PATROL for Windows 2000 Installation Guide* or the installation guide for the component that you are using.

Files Installed

This table lists the files that are installed with the PATROL Adapter for WBEM.

Table 1-3 Installed Files (Part 1 of 5)

File Name	Description
<Target Directory>\lib\wbem\tools	
pwinstpr.dll	PATROL Instance and Method Providers
evtprov.dll	PATROL Event Provider
pemsvr.exe	PEM COM server used by providers
msgdll.dll	string table used by providers
km2mof.exe	PATROL KM2MOF Tool
km2mof.res	string table used by KM2MOF Tool
pkmparser.dll	COM server used by KM2MOF Tool
pwbem.xpc	PATROL PSL API server
mfc42.dll	Microsoft Visual C++ distributable dll
msvbvm50.dll	Microsoft Visual Basic distributable dll
patmof.dll	helper routines used during the install
rundll32.exe	utility used to invoke routines within a DLL
wmi_util.exe	utility used during install/uninstall
wbemadapter_migrate.txt	instructions used by wmi_util.exe
wmicheck.exe	determines whether to install WMI core components (only used when these components have been selected during the installation)
lmsrt.dll	Microsoft Visual C++ local runtime dll
lmsirt.dll	Microsoft Visual C++ local runtime dll
lmspnt.dll	Microsoft Visual C++ local runtime dll

Table 1-3 Installed Files (Part 2 of 5)

File Name	Description
<Target Directory>\lib\wbem\schemas	
patagent.mof	PATROL-generic schema file to import classes and instances into CIMOM
patcimv2.mof	CIMV2 schema file to import classes into root\PATROL namespace in CIMOM
patevent.mof	Event Provider schema file for provider registration and event class
patkm.mof	KM-specific schema file to import classes and instances into CIMOM
patremot.mof	defines PATROL namespace
patsetup.mof	installation file for creation of PATROL_Agent and PATROL_AgentDefaults instances
<Target Directory>\lib\wbem\doc	
User_Guide.pdf	<i>PATROL Adapter for WBEM User Guide</i>
PSL_API_Reference.pdf	<i>PATROL Adapter for WBEM Reference Manual</i>
Release_Notes.pdf	<i>PATROL Adapter for WBEM Release Notes</i>
<Target Directory>\lib\wbem	
wbemadapter.adm	system policy editor template
readme.txt	readme file
<SystemRoot>	
pwa_uninst.dll	uninstall extension dll
<Target Directory>\lib\knowledge	
wbem.kml	PATROL Adapter for WBEM KM list file
wbem.km	PATROL Adapter for WBEM KM file
wbem_cimom.km	PATROL Adapter for WBEM KM file
wbem_namespace.km	PATROL Adapter for WBEM KM file

Table 1-3 Installed Files (Part 3 of 5)

File Name	Description
wbem_process.km	PATROL Adapter for WBEM KM file
wbem_provider.km	PATROL Adapter for WBEM KM file
<Target Directory>\lib\msgs	
wbem.msg	PATROL Adapter for WBEM message file
util.msg	PATROL Adapter for WBEM message file
<Target Directory>\lib\psl	
pwbemlib.lib	PATROL Adapter for WBEM PSL library file
utlutil.lib	PATROL Adapter for WBEM PSL library file
utmsgsl.lib	PATROL Adapter for WBEM PSL library file
utstdmsgl.lib	PATROL Adapter for WBEM PSL library file
utstrl.lib	PATROL Adapter for WBEM PSL library file
wbemutil.lib	PATROL Adapter for WBEM PSL library file
wbemcmdagentadd.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdefaults.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdelete.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentmodify.psl	PATROL Adapter for WBEM PSL file
wbemcmdcimomstart.psl	PATROL Adapter for WBEM PSL file
wbemcmdcimomstop.psl	PATROL Adapter for WBEM PSL file
wbemcmdconfig.psl	PATROL Adapter for WBEM PSL file
wbemcmdkmdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdloadkm.psl	PATROL Adapter for WBEM PSL file
wbemcmdloadmof.psl	PATROL Adapter for WBEM PSL file

Table 1-3 Installed Files (Part 4 of 5)

File Name	Description
wbemcmdprovdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdquery.psl	PATROL Adapter for WBEM PSL file
wbemcmdunloadkm.psl	PATROL Adapter for WBEM PSL file
wbemdiscovery.psl	PATROL Adapter for WBEM PSL file
wbemprmpoccoll.psl	PATROL Adapter for WBEM PSL file
WBEMCmdExport.psl	exports a MOF file containing the list of PATROL_Agent instances
WBEMCmdSysReport.psl	generates a computer system report
<Target Directory>\lib\images	
addrspace_ok.bmp	image file
database_ok.bmp	image file
database_warn.bmp	image file
dbms_ok.bmp	image file
filesystem_ok.bmp	image file
filesystem_warn.bmp	image file
process_ok.bmp	image file
process_warn.bmp	image file
<Target Directory>\lib\wbem\examples\msvc\events	
ReadMe.txt	describes the MSVC project This example demonstrates a WMI client retrieving PATROL events using the PATROL Event model.
events.cpp	MSVC project file.
events.dsp	MSVC project file.
events.dsw	MSVC project file.
sink.cpp	MSVC project file.
sink.h	MSVC project file.
<Target Directory>\lib\wbem\examples\msvc\km_model	

Table 1-3 Installed Files (Part 5 of 5)

File Name	Description
ReadMe.txt	describes the msvc project This example retrieves data from PATROL using the PATROL Knowledge Module model.
NT_LOGICAL_DISKS.mof	MSVC project file.
km_model.cpp	MSVC project file.
km_model.dsp	MSVC project file.
km_model.dsw	MSVC project file.
<Target Directory>\lib\wbem\examples\msvc\agent_model	
ReadMe.txt	describes the msvc project This example retrieves data from PATROL using the PATROL Agent model.
agent_model.cpp	MSVC project file.
agent_model.dsp	MSVC project file.
agent_model.dsw	MSVC project file.

Classes Installed

The classes added to the local **root/CIMV2** CIM namespace during the installation of the PATROL Adapter for WBEM are listed below.

Table 1-4 Installed Classes

Class	Sub-Class
PATROL_ManagedObject	PATROL_Agent PATROL_MgmtApp PATROL_Parameter PATROL_KM_Parameter PATROL_AppInstance PATROL_KM_AppInst
PATROL_Dependency	PATROL_AgentManages PATROL_AppHasInstances PATROL_AppInstHasParameters PATROL_AppInstContains PATROL_KM_PackageHasMgmtApp PATROL_KM_MgmtAppHasParameter PATROL_KM_MgmtAppHasLocal PATROL_KM_AgentHasAppInstances
PATROL_Error	none
PATROL_Event	none
PATROL_KM_Package	none
PATROL_KM_MgmtApp	PATROL_KM_MgmtAppLocal

The local **root/PATROL** CIM namespace and all classes in it are also added during the installation of the PATROL Adapter for WBEM.

The classes are described in detail throughout this book. The **.mof** files used to import the classes into CIMOM are described in “MOF Files” on page 2-22. When uninstalling the PATROL Adapter for WBEM, all unused classes are removed as long as they are not used by another program such as PATROL Web Viewer.

Installation of the PATROL_Agent Instance

The adapter installation program may not be able to retrieve account information (such as user name and password) for a local PATROL Agent, which usually happens when an agent is not installed. When it cannot get the account information, the installation program creates an instance of the PATROL_Agent class in the CIMV2 namespace containing only default values, but it does not create a PATROL_Agent instance in the PATROL namespace.

If you have no intention of installing a PATROL Agent on the local system, the situation described in the previous paragraph is perfectly acceptable. On the other hand, installing a local PATROL Agent requires that you add the PATROL_Agent instance to the PATROL namespace and edit the default information in the PATROL_Agent instance in the CIMV2 namespace. Use Table 1-5 to find the right tool to edit the instances.

Note

Editing the instances is not necessary if the installation program for the PATROL Adapter for WBEM has no trouble retrieving the account information from a locally installed PATROL Agent.

Table 1-5 Namespace Editing Tools

Namespace	Editing tools
CIMV2	WBEM KM or WMI CIM Studio
PATROL	WBEM KM, PATROL [®] Web Viewer, or PATROL [®] QuickView

Installation of the WBEM KM on a Unix Computer

Even though PATROL Adapter for WBEM is a product for Windows 2000 (or Windows NT,) you can install the Knowledge Module on a Unix computer. Install to the Unix platform only if you want to use a PATROL Console for Unix to monitor WMI activity. Since WMI is a Windows-only program, a PATROL Console for Unix can only monitor this activity from a remote computer.

Installing the KM to a Unix computer means copying the KM files listed in Table 1-6 to a Unix computer. You can perform this copy by finding a Windows computer that has the PATROL KM for WBEM installed and copying these files to the Unix computer. Each file needs to be copied to the same directory on the Unix computer as the directory where it resided on the Windows computer. For example, you would copy the **wbem.kml** file from the **%PATROL_HOME%\lib\knowledge** directory on the Windows computer to the **\$PATROL_HOME/lib/knowledge** directory on the Unix computer. After copying every file in Table 1-6, load the KM into the PATROL Console for Unix.

Note

These files are also listed in Table 1-3, “Installed Files,” on page 1-6.

Table 1-6 Installed KM Files (Part 1 of 3)

File Name	Description
<Target Directory>\lib\knowledge	
wbem.kml	PATROL Adapter for WBEM KM list file
wbem.km	PATROL Adapter for WBEM KM file
wbem_cimom.km	PATROL Adapter for WBEM KM file
wbem_namespace.km	PATROL Adapter for WBEM KM file
wbem_process.km	PATROL Adapter for WBEM KM file
wbem_provider.km	PATROL Adapter for WBEM KM file

Table 1-6 Installed KM Files (Part 2 of 3)

File Name	Description
<Target Directory>\lib\msgs	
wbem.msg	PATROL Adapter for WBEM message file
util.msg	PATROL Adapter for WBEM message file
<Target Directory>\lib\psl	
pwbemlib.lib	PATROL Adapter for WBEM PSL library file
utlutil.lib	PATROL Adapter for WBEM PSL library file
utmsgsl.lib	PATROL Adapter for WBEM PSL library file
utstdmsgsl.lib	PATROL Adapter for WBEM PSL library file
utstrlib.lib	PATROL Adapter for WBEM PSL library file
wbemutil.lib	PATROL Adapter for WBEM PSL library file
wbemcmdagentadd.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdefaults.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdelete.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdagentmodify.psl	PATROL Adapter for WBEM PSL file
wbemcmdcimomstart.psl	PATROL Adapter for WBEM PSL file
wbemcmdcimomstop.psl	PATROL Adapter for WBEM PSL file
wbemcmdconfig.psl	PATROL Adapter for WBEM PSL file
wbemcmdkmdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdloadkm.psl	PATROL Adapter for WBEM PSL file
wbemcmdloadmof.psl	PATROL Adapter for WBEM PSL file
wbemcmdprovdetails.psl	PATROL Adapter for WBEM PSL file
wbemcmdquery.psl	PATROL Adapter for WBEM PSL file
wbemcmdunloadkm.psl	PATROL Adapter for WBEM PSL file
wbemdiscovery.psl	PATROL Adapter for WBEM PSL file
wbemprmproccoll.psl	PATROL Adapter for WBEM PSL file
WBEMCmdExport.psl	exports a MOF file containing the list of PATROL_Agent instances
WBEMCmdSysReport.psl	generates a computer system report

Table 1-6 Installed KM Files (Part 3 of 3)

File Name	Description
<Target Directory>\lib\images	
addrspace_ok.bmp	image file
database_ok.bmp	image file
database_warn.bmp	image file
dbms_ok.bmp	image file
filesystem_ok.bmp	image file
filesystem_warn.bmp	image file
process_ok.bmp	image file
process_warn.bmp	image file

PATROL Models for CIM

This chapter describes the models, or schemas, that the PATROL Adapter for WBEM product provides for CIM.

The following topics are discussed:

Overview	2-2
UML Model Diagrams	2-2
CIM Namespaces	2-3
PATROL Agent Model	2-5
PATROL Agent Classes	2-7
PATROL Agent Associations	2-7
Methods	2-8
PATROL Knowledge Module Model	2-16
PATROL KM Classes	2-19
PATROL KM Associations	2-20
PATROL Event Model	2-21
MOF Files	2-22

Overview

The models, or schemas, provided by the PATROL Adapter for WBEM are implemented as **.mof** (Managed Object Format) files that can be loaded into the CIM Object Manager (CIMOM). The **.mof** files contain the PATROL model class definitions.

The following types of models are presented in this chapter:

- PATROL Agent model—provides only runtime information
- PATROL KM model—provides runtime and PATROL KM definition information
- PATROL Event model—maps the data contained in PATROL events into a single CIM class

The PATROL KM models can be used only in combination with the PATROL KM2MOF Tool and provide application-specific information. For specific information on how the PATROL KM2MOF Tool is used, see Chapter 4, “PATROL KM2MOF Tool.”

UML Model Diagrams

As in Unified Modeling Language (UML) notation, rectangles represent classes, lines represent associations, and arrows represent generalization or specialization. Classes include name, relevant properties, and methods. In contrast to UML, abbreviated keywords are added after class and property names. Table 2-1 describes the elements of the diagrams.

Table 2-1 Diagram Elements (Part 1 of 2)

Element	Description
	inheritance hierarchy
	association between two classes

Table 2-1 Diagram Elements (Part 2 of 2)

Element	Description
.....	a link from the association line to its definition (box with association properties) When an association definition is not in the diagram, the name of the association is placed next to the association line.
0, 1, 2...	on a line, the number of objects in the class (0* means "0 or more", etc.).
a	abstract
d	dynamic (instances generated by PATROL Providers)
k	key
o	override
p	propagate
r	reference
s	static

CIM Namespaces

Multiple PATROL Agents are represented in CIMOM, which acts as a central repository. Two namespaces are loaded with the PATROL models. They are defined in Table 2-2.

Table 2-2 CIM Namespaces

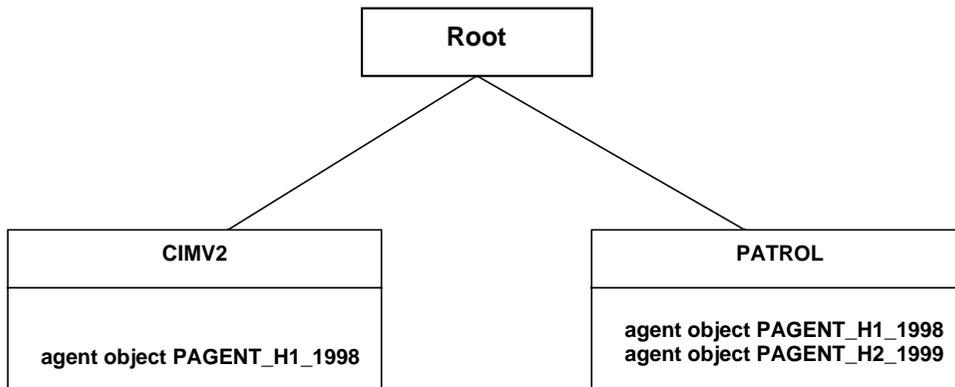
Namespace	Contains
\\.\root\PATROL	all PATROL Agents
\\.\root\CIMV2	local PATROL Agent only

The CIMV2 namespace is used by Microsoft to store data for the local machine. Due to WMI query scope restrictions, the data for the local PATROL Agent is also stored here.

Navigation and querying are available across all PATROL Agents because the PATROL namespace provides a view that includes both local and remote PATROL Agents.

Figure 2-1 illustrates CIMOM and a local PATROL Agent on host H1, port 1998. A remote PATROL Agent is on host H2, port 1999. The PATROL Agent on host H1 is represented in the `\\.\root\CIMV2` namespace. Both PATROL Agents are represented in the `\\.\root\PATROL` namespace. In both namespaces, the PATROL Agent objects are associated with the corresponding application, application instance, and parameter objects. Figure 2-1 also illustrates the namespace structure and top-level PATROL Agent objects that are created in CIMOM.

Figure 2-1 PATROL Models Namespace Structure



PATROL Agent Model

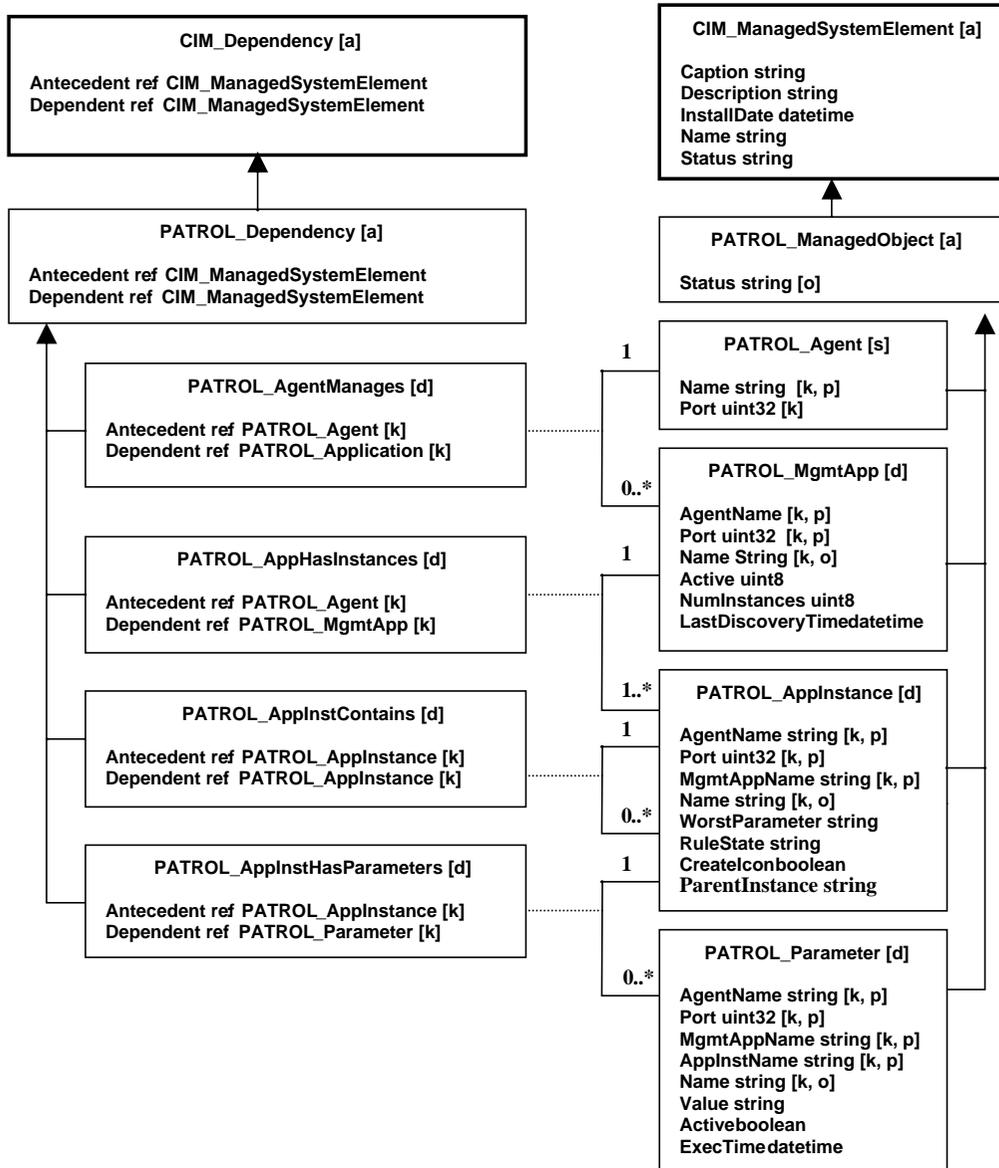
The PATROL Agent model represents the information that is available from a running PATROL Agent. A PATROL Agent manages a set of PATROL Knowledge Modules (KMs) and discovers instances of the applications being monitored. Each instance has a set of parameters that provide the monitored information.

The PATROL Agent model represents monitored information through a set of classes and associations that are derived from the CIM model. The information provided is dynamic and is related to the current value of the parameter. The PATROL Agent model is shown in Figure 2-2.

Note

See “UML Model Diagrams” on page 2-2 for help interpreting Figure 2-2.

Figure 2-2 PATROL Agent Model Diagram



Note

Only representative properties are shown.

PATROL Agent Classes

Table 2-3 defines the classes used in the PATROL Agent model.

Table 2-3 PATROL Agent Classes

Class Name	Instance Description
PATROL_Agent	fundamental software component of the PATROL product Contains parameters and methods needed to connect to and retrieve information from a PATROL Agent. Provides system management services. Manages the other objects of the PATROL Agent model.
PATROL_MgmtApp	PATROL management application (PATROL KM)
PATROL_AppInstance	instances of applications discovered and monitored by the PATROL Agents
PATROL_Parameter	monitored application instance parameter
PATROL_ManagedObject	wrapper for the CIM_ManagedSystemElement object Represents commonality of all PATROL-managed elements.

PATROL Agent Associations

Table 2-4 defines the associations used in the PATROL Agent model.

Table 2-4 PATROL Agent Associations (Part 1 of 2)

Association Name	Description
PATROL_AgentManages	enumeration of a PATROL Agent's loaded KMs Each instance links a PATROL Agent to the KMs that it manages.
PATROL_AppHasInstances	enumeration of KM instances
PATROL_AppInstHasParameters	enumeration of KM instance parameters

Table 2-4 PATROL Agent Associations (Part 2 of 2)

Association Name	Description
PATROL_ApplInstContains	container for other KM instances to form the containment hierarchy that is used to display objects in the PATROL Console
PATROL_Dependency	wrapper for the CIM_Dependency object All PATROL associations are derived from CIM_Dependency.

Methods

The PATROL Method Provider supports the methods described in this section. All of the methods are static. The methods that require a valid instance of the PATROL_Agent class to function are indicated with footnotes.

PATROL_Agent::Discover

The PATROL_Agent::Discover method discovers PATROL Agents and contains the parameters in Table 2-5.

Table 2-5 PATROL_Agent::Discover Parameters (Part 1 of 2)

Parameter	Description
string Address	host name or IP address
string PortRange	range of ports to scan The format is <lowerPortRange>- <upperPortRange>, but only <lowerPortRange> is required.
uint32 Timeout	maximum number of seconds between PATROL Agent discoveries All agents discovered to that point are returned. If this parameter is too low, some agents may be missed. If it is too high, the call may take longer than necessary.

Table 2-5 PATROL_Agent::Discover Parameters (Part 2 of 2)

Parameter	Description
boolean ScanSubnet	boolean parameter This parameter can cause all IP addresses within the 'C'-class network IP address specified in the Address field to be scanned for PATROL Agents within the specified PortRange. The address must be a valid IP Address, not a host name.
boolean ResolveHostnames	boolean parameter A parameter that discovers PATROL Agent IP addresses to be resolved to host names.
string Delimiters	string of two characters The first character delimits fields and the second character delimits PATROL Agents in the returned string.

This method returns a list of PATROL Agents using the following syntax:

<name/IP address><fd><port><fd><status><ad>

where <fd> = field delimiter and <ad> = agent delimiter

PATROL_Agent::EncryptPassword

The PATROL_Agent::EncryptPassword method encrypts a clear-text password. The result can be used in the PATROL_Agent Password field.

Table 2-6 PATROL_Agent::EncryptPassword Parameters

Parameter	Description
string Password	string to be encrypted
string Result	encrypted string

PATROL_Agent::GetVariable

The PATROL_Agent::GetVariable method retrieves PATROL Agent variables.

Table 2-7 PATROL_Agent::GetVariable Parameters

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Variable	full path of the PATROL Agent variable (i.e. /status)
string Result	value of the variable

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Agent::UpdateProvider

The PATROL_Agent::UpdateProvider method notifies the PATROL Providers of a change to a PATROL_Agent instance.

Table 2-8 PATROL_Agent::UpdateProvider Parameters

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Result	empty on failure

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Agent::Disconnect

The PATROL_Agent::Disconnect method terminates a PATROL Agent connection. This should only be used if you delete a PATROL_Agent instance and no longer want to receive data from that PATROL Agent.

Table 2-9 PATROL_Agent::Disconnect Parameters

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Result	empty on failure

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Agent::AuthenticateOnConnect

This method determines whether authentication is based on the user account used to access a PATROL agent.

Table 2-10 PATROL_Agent::AuthenticateOnConnect Parameters

Parameter	Description
boolean result ¹	true or false result The result is true if the Single Sign-On security plug-in is registered. This result also means that the PATROL Agent authenticated the account used to launch the PEM Server. The user name and password settings in the PATROL_Agent instances are not used. The result is false if the user name and password settings in the PATROL_Agent instances must be used (default case).

¹The result is determined by the registered PATROL security plug-ins. See "PATROL Single Sign-On" on page C-3.

PATROL_Event::GetEventHistory

The PATROL_Event::GetEventHistory method retrieves events sent previously by a specified PATROL Agent.

Table 2-11 PATROL_Event::GetEventHistory Parameters (Part 1 of 2)

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
uint32 Timeout	maximum number of seconds between event history messages sent by the PATROL agent When this interval is exceeded, all events discovered to that point are returned. If this interval is too low, some events may be missed. If it is too high, the call may take longer than necessary.
uint32 MaxReplies	maximum number of events to be returned
string StartTimeFilter	earliest event timestamp All of the standard PSL formats are acceptable (for example, Sun Jan 01 01:00:00 1990). Dates outside the range 1902-2037 are invalid.
string EndTimeFilter	latest event timestamp Dates outside the range 1902-2037 are invalid.
string StatusFilter	string of one or more status tags that are separated by commas O - open A - acknowledged E - escalated C - closed D - deleted (for example, O,C matches open and closed events)
string TypeFilter	string of one or more type tags that are separated by commas I - information S - state change E - error W - warning A - alarm R - response (for example, W,A matches warning and alarm events)

Table 2-11 PATROL_Event::GetEventHistory Parameters (Part 2 of 2)

Parameter	Description
string NSeverityFilter	event severity from 1 to 5, with 5 most severe An empty string matches all events.
string NodeFilter	event host name filter An empty string matches all host names.
string OriginFilter	event origin filter An empty string matches all origins.
string PatternFilter	event description filter An empty string matches all descriptions.
string EidRange	event ID range filter x = Event ID x x/y = Event IDs between x and y -/y = Event IDs less than or equal to y x/- = Event IDs greater than or equal to x -/- = All events
string EvClass	event class filter An empty string matches all classes.
string Delimiters	string of two characters The first delimits fields and the second delimits events in the returned string.
string Result	list of events returned by the PATROL Agent in the following syntax: <time><id><args><class><desc><diary><catalog><expectancy><handler><severity><origin><owner><source><status><type> Each field is separated by the field delimiter and each event by the event delimiter.

¹ A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Parameter::GetParamAlarmRanges

The PATROL_Parameter::GetParamAlarmRanges method returns parameter alarm ranges and is equivalent to the PSL get_ranges function.

Table 2-12 PATROL_Parameter::GetParamAlarmRanges Parameters

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Parameter	parameter (/NT_LOGICAL_DISKS/C:/LdIdFreeSpacePercent)
string Result	annotations in the syntax returned by the PSL get_ranges function For more information, see the PATROL <i>Script Language Reference Manual</i> .

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Parameter::GetParamAnnotation

The PATROL_Parameter::GetParamAnnotation method returns parameter annotations and is equivalent to the PSL function annotate_get.

Table 2-13 PATROL_Parameter::GetParamAnnotation Parameters (Part 1 of 2)

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Parameter	parameter (/NT_LOGICAL_DISKS/C:/LdIdFreeSpacePercent)
string TimeStamp	timestamp of the parameter value for which the annotation should be returned The default is the most recent data point.

Table 2-13 PATROL_Parameter::GetParamAnnotation Parameters (Part 2 of 2)

Parameter	Description
string Result	annotations in the following syntax: <code><ann1>\n<ann2>\n... \n<annN></code> where <code><ann1></code> = the first data argument specified in the <code>annotate()</code> call

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL_Parameter::GetParamHistory

The PATROL_Parameter::GetParamHistory method retrieves previous values of a given parameter from a PATROL Agent and is equivalent to the PSL history function.

Table 2-14 PATROL_Parameter::GetParamHistory Parameters

Parameter	Description
string AgentName	name of the PATROL Agent ¹
uint32 Port	PATROL Agent port ¹
string Parameter	parameter (<code>/NT_LOGICAL_DISKS/C:/LdIdFreeSpacePercent</code>)
string Format	up to 3 characters, specifying the format of Result: n - number of values returned t - time returned for each entry v - value returned for each entry
uint32 MaxReplies	maximum number of entries to be returned
string Result	up to MaxReplies entries in the following syntax: <code><n>\n<v> <t>\n... \n<v> <t>\n</code> in which <code><n></code> = number of available history data points, <code><v></code> = value, <code><t></code> = time

¹A PATROL_Agent instance must correspond to the AgentName and Port. You can use the WBEM KM to add this information.

PATROL Knowledge Module Model

The PATROL KM model represents the information and definitions from a PATROL KM and its corresponding application instances created while running a PATROL Agent. Each PATROL KM is defined by a set of properties, parameters, and menu commands. Each PATROL KM shows a set of application instances to monitor and defines the corresponding parameters.

The PATROL KM model represents this structure by using a set of classes and associations that are derived from the CIM model. The information provided for the PATROL KM definition is static. The information about the current value of the parameter is dynamic. To view application-specific data, you will need a PATROL KM model for each KM that you are monitoring. Figure 2-3 is a diagram of the PATROL KM model.

Note

See “UML Model Diagrams” on page 2-2 for help interpreting Figure 2-3.

Figure 2-3 PATROL Knowledge Module Model Diagram

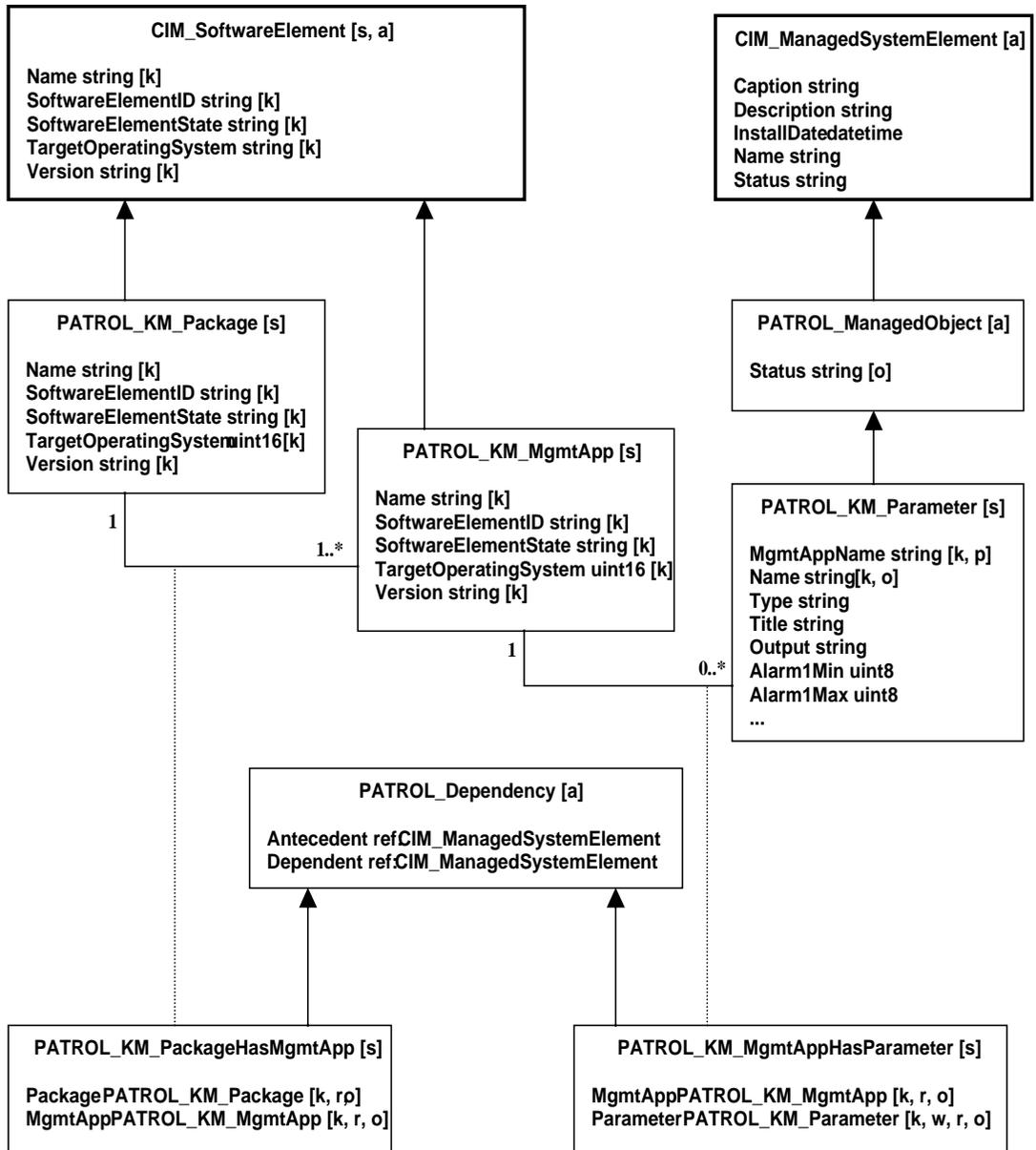
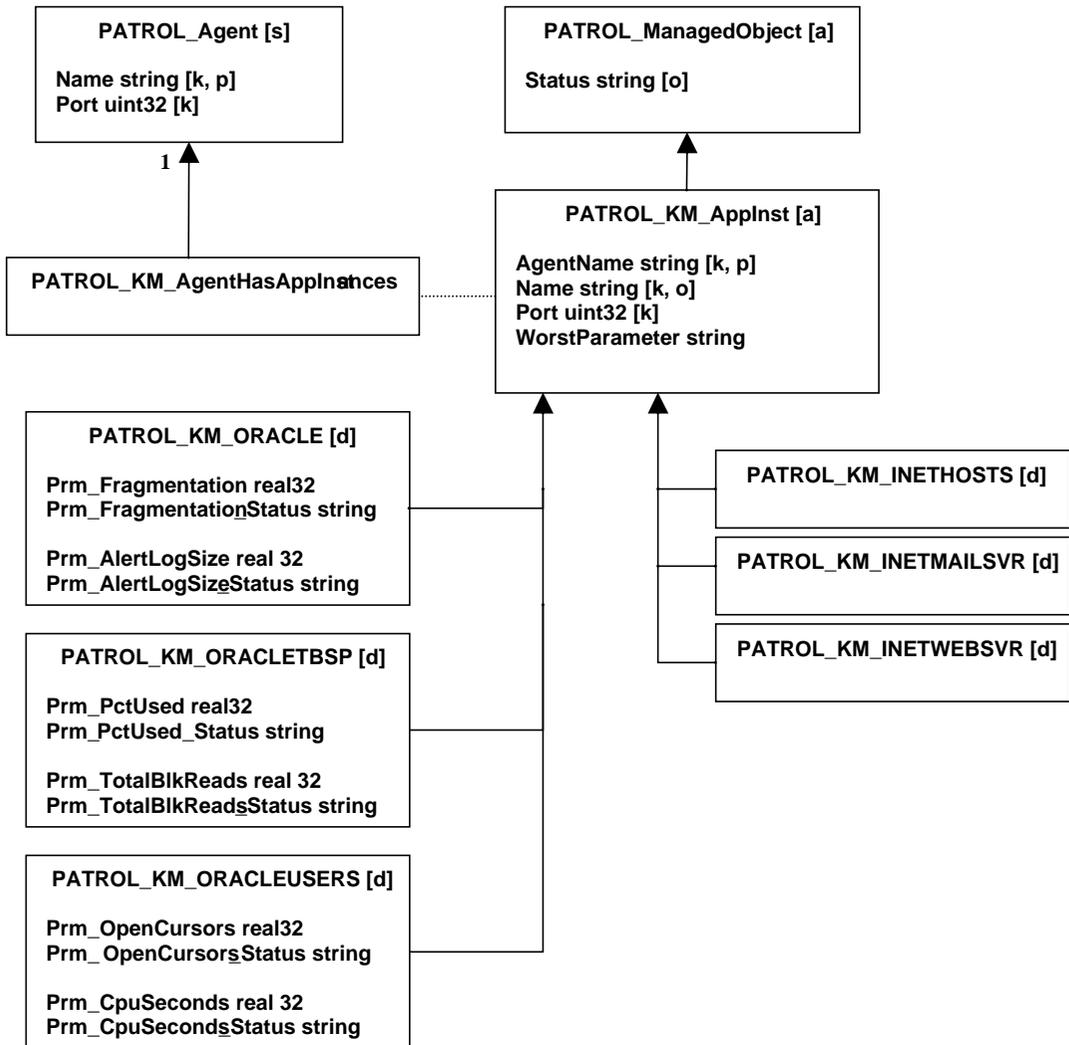


Figure 2-4 Example PATROL KM Model Using the Oracle KM



PATROL KM Classes

The PATROL_KM objects define the corresponding PATROL objects. They are stable over time and change only when their PATROL KM components change. Table 2-15 defines the classes used in the PATROL KM model.

Table 2-15 PATROL KM Model Classes

Class Name	Description
PATROL_KM_Package	<p>set of PATROL .km files that make up a PATROL KM</p> <p>This class is derived from CIM_SoftwareElement, and it includes one or more PATROL_KM_MgmtApp components.</p>
PATROL_KM_MgmtApp	<p>management applications that correspond to .km files</p> <p>This class specifies KM properties (current state excluded), parameters, commands, etc. Each management application includes a set of parameter objects, PATROL_KM_Parameter, and a set of command objects, PATROL_KM_Command. When a KM defines applications to be monitored, it creates objects that correspond to instances of these applications. These application-specific objects provide current values and states of the parameters, and through these objects, commands are implemented for the management application.</p>
PATROL_KM_Parameter	parameter properties, excluding current state, in .km files.
PATROL_KM_AppInst	<p>instance of an application monitored by a PATROL KM</p> <p>This class includes properties common to instances and current state.</p>
PATROL_KM_<km name>	<p>specific instance of a monitored application</p> <p>This class is created by the KM2MOF Tool. It inherits from PATROL_KM_AppInst. An instance of PATROL_KM_<km name> exists for each monitored instance of the application monitored by <km name>.</p>

PATROL KM Associations

Table 2-16 defines the associations used in the PATROL KM model.

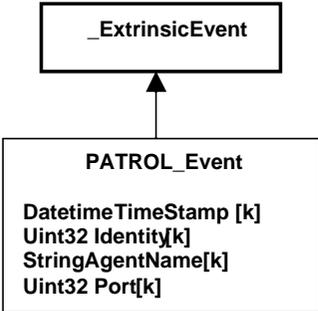
Table 2-16 PATROL KM Model Associations

Association Name	Description
PATROL_KM_PackageHasMgmtApp	top-level association that defines the PATROL KM components Each component is a management application.
PATROL_KM_MgmtAppHasParameter	set of parameters for each management application
PATROL_KM_AgentHasAppInstances	provides scoping for the application instances discovered by a PATROL Agent Monitors multiple instances of applications.

PATROL Event Model

The PATROL Event model maps the data contained in PATROL events into a single CIM class, PATROL_Event, which is illustrated in Figure 2-5. This class is derived from the __ExtrinsicEvent class defined by Microsoft.

Figure 2-5 PATROL Event Model Diagram



Note

To retrieve previous events, see “PATROL_Event::GetEventHistory” on page 2-12.

MOF Files

There are several **.mof** files included in the PATROL Adapter for WBEM product. These files are used to import the classes that are needed for the models described in this chapter into CIMOM.

You can access these files in the **%PATROL_HOME%\lib\wbem\schemas** directory installed by the PATROL Adapter for WBEM.

Table 2-17 MOF Files

MOF File	Contents
patagent.mof	PATROL generic schema class and instance registration
patcimv2.mof	CIMV2 class
patevent.mof	PATROL Event Provider registration and event class
patkm.mof	PATROL KM-specific class
patremot.mof	PATROL namespace
patsetup.mof	PATROL_Agent and PATROL_AgentDefaults instances installation file

PATROL Providers

This chapter provides you with an overview of the PATROL[®] Providers for WBEM.

The following topics are discussed:

Overview	3-2
PATROL Agent Connections	3-3
Error Logging	3-3
PATROL Instance Provider	3-4
Improving Performance	3-4
PATROL Event Provider	3-5
PATROL Method Provider	3-6

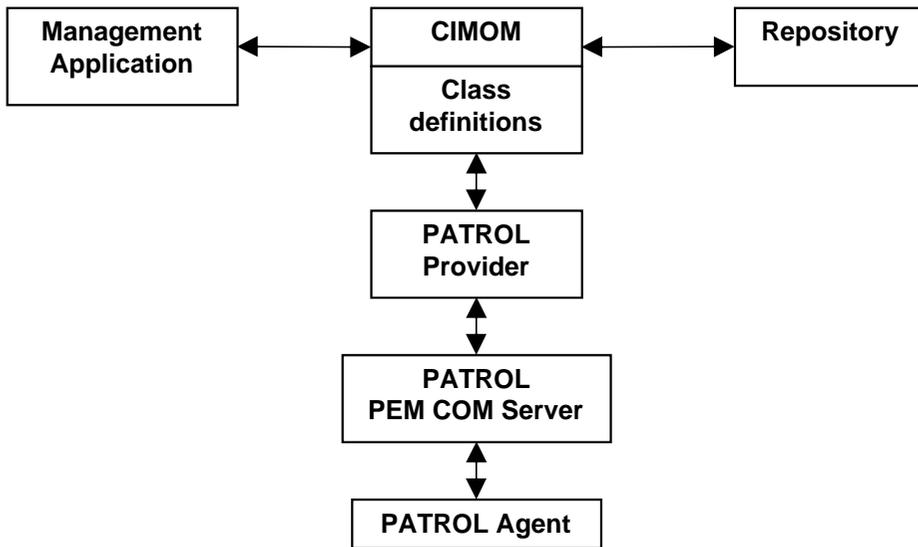
Overview

The PATROL Providers are COM servers that provide PATROL data to CIMOM on demand. When CIMOM receives a request from a management application for data that is not in its repository, it forwards the request to a provider, which returns the data in the form of CIM objects.

The types of PATROL providers are as Follows:

- PATROL Instance Provider
- PATROL Event Provider
- PATROL Method Provider

Figure 3-1 PATROL Provider Interaction Diagram



PATROL Agent Connections

PATROL Agent connections are maintained by the PEM COM server. Providers connect to a PATROL Agent on demand and, once established, the connection is shared by all providers. The session layer below the PEM periodically checks connectivity to the PATROL Agent and could cause the connection to fail. The time frame used to check connectivity is determined by the heartbeat setting for each PATROL Agent instance.

The PATROL Agent connection settings are retrieved from the corresponding PATROL_Agent instance.

Within the PATROL namespace, the PATROL_AgentDefaults instance supplies default values for attributes which are not specified within specific PATROL_Agent instances. Changes to the defaults are used immediately, but do not affect existing connections.

The providers do not monitor changes to the PATROL_Agent or PATROL_AgentDefaults instances. If a client detects a change to an instance, it can

- ignore the change
- update an existing connection using the PATROL_Agent::UpdateProvider() method
- force a disconnect using PATROL_Agent::Disconnect()

The next request reconnects to the PATROL Agent using the updated values. This may disrupt requests made by other applications.

For more information about using the methods, see “Methods” on page 2-8.

Error Logging

All errors and other messages detected by the PATROL Providers are sent to the Windows Application Event Log. To access this log, select **Start => Programs => Administrative Tools => Event Viewer => Log => Application**.

Output to the event log is controlled by the Logging environment variables shown in Appendix D, “Registry Settings.”

PATROL Instance Provider

The PATROL Instance Provider delivers instances of PATROL classes. It creates each instance by retrieving information from PATROL Agents with the PEM COM server and then sending the instance to CIMOM. CIMOM forwards instances to the management application.

The WMI SDK error routines can be used to retrieve detailed information in the event of an error. Requests that retrieve data from multiple PATROL Agents are split into individual requests by PATROL Agent. The request succeeds unless all individual requests fail. If any of these requests fail, error information can be retrieved as documented in the WMI SDK.

Improving Performance

When the Instance Provider receives a request for PATROL data from CIMOM, it enumerates all instances of PATROL_Agent within the namespace, forwards the request to each instance, then waits until all results return. CIMOM then post-filters the instances. To improve performance, you can reduce the number of PATROL Agents contacted by placing the agent list in the context of the WMI request. The format is as follows:

Class	Value
AgentName	<agent1 name>:<agent1 port>,...<agentN name>:<agentN port>

This information is used only for dynamically provided data in the **root/patrol** namespace.

To reduce the number of instances retrieved in a WBEM Query Language (WQL) select request of PATROL_AppInstance or PATROL_Parameter instances, use the following information in the context of the query request:

Class	Context Value (string)	Context Property
PATROL_AppInstance	MgmtAppName	the name of the application
PATROL_Parameter	AppInstName	the name of the application instance

Warning

Use of the context to improve performance is temporary. Queries should specify all required information to ensure that results are returned reliably in the future. For example:

```
'select * from PATROL_Parameter where  
AgentName="<name>" and Port=<port> and  
MgmtAppName="<app>" and AppInstName="<inst>"'
```

PATROL Event Provider

The PATROL Event Provider registers with PATROL to receive notification of PATROL events as they occur. When a CIMOM event consumer registers an interest in instances of the PATROL_Event class, CIMOM passes the subscription request to the Event Provider, which subscribes to the corresponding PATROL Agent events.

The PATROL Event Provider polls the PEM COM server to ensure that it is delivering events. To control this poll rate, refer to Appendix D, “Registry Settings.”

The Event Provider generates an event when detecting certain conditions. The ID for this event is zero. The type of event is contained in the argument field. The following events are generated by the PATROL Event Provider.

Table 3-1 Generated Events

Argument	Description
0	Event delivery from a PATROL Agent has been stopped. The connection has been closed by the PEM COM server. The agent name field is filled.
1	The PEM COM server has gone down and an attempt has been made to re-establish contact. The event description contains either succeeded or failed .
2	An attempt to subscribe to events from a PATROL Agent has failed. The agent name field is filled.

If the subscription request specifies a set of PATROL Agents, only events from these agents return. Otherwise, all instances of PATROL_Agent enumerate, and all events return.

PATROL Events are represented within CIMOM as instances of the PATROL_Event class. For more information, see “PATROL Event Model” on page 2-21.

PATROL Method Provider

The PATROL Method Provider handles calls to the WMI methods. It translates each call into a command that is executed by one of the PATROL Adapter for WBEM components or PATROL Agents. If a method specifies a specific PATROL Agent, an attempt is made to establish a connection to that PATROL Agent by using the given agent name and port number.

The PATROL Method Provider supports the methods described in Chapter 2, “Methods.”

PATROL KM2MOF Tool

This chapter provides you with an overview of the PATROL[®] KM2MOF Tool and describes how to use it.

The following topics are discussed:

Overview	4-2
Using the PATROL KM2MOF Tool	4-4
Command Line Options	4-4
Compiling a .km File	4-5
Compiling a .kml File	4-6
Loading .mof Files into CIMOM	4-7
PATROL KM Properties Mapped to MOF Files	4-7
Messages and Errors	4-10

Overview

The PATROL KM2MOF Tool is a compiler that reads a PATROL KM and generates the corresponding CIM model for WBEM. For more information on the PATROL KM model, see “PATROL Knowledge Module Model” on page 2-16.

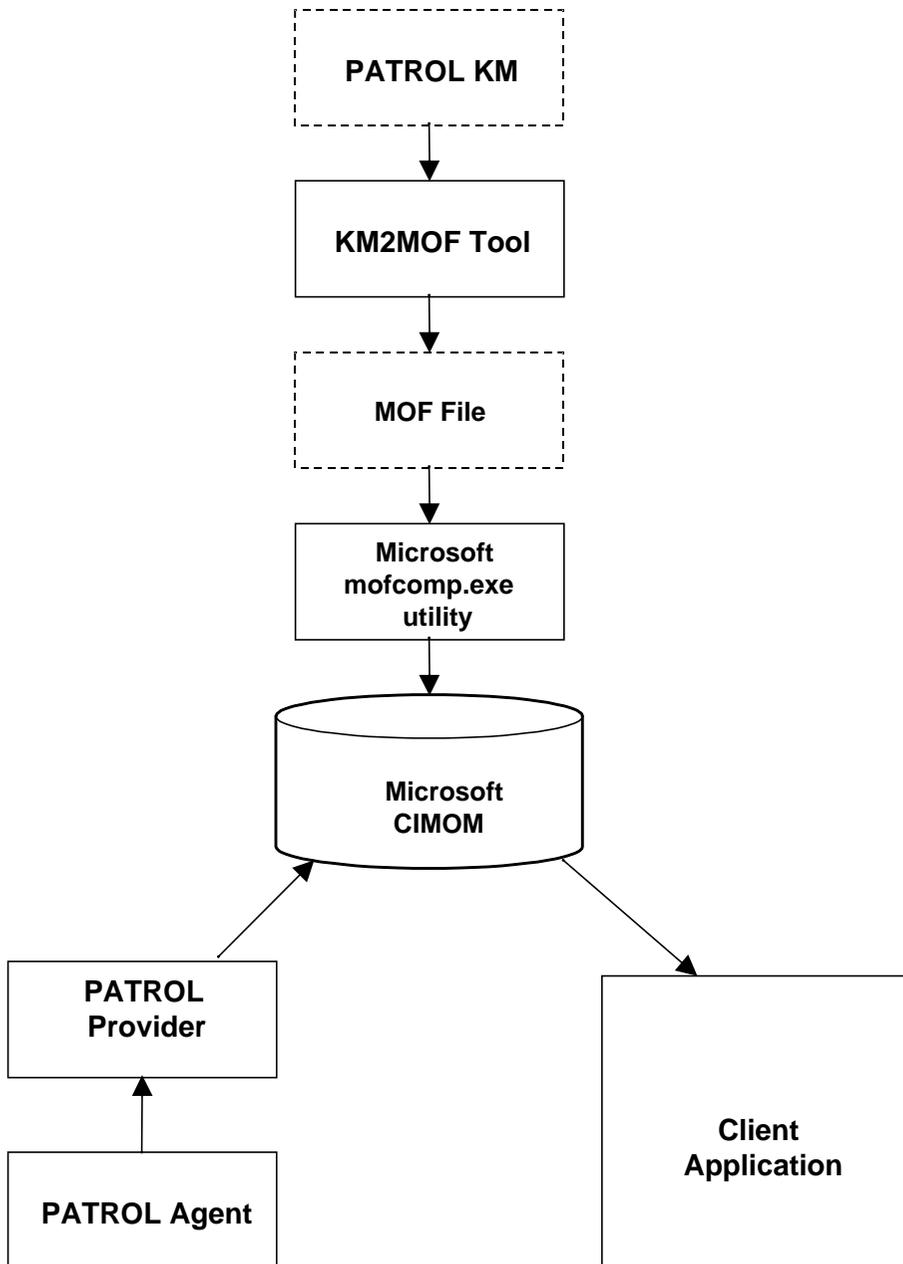
The PATROL KM2MOF Tool converts **.km** and **.kml** files into **.mof** files that can be loaded into CIMOM; applications that use the CIM standard can then interpret a PATROL KM.

Note

The PATROL KM2MOF Tool does not support PATROL KM computer classes.

The flowchart shown in Figure 4-1 shows the progression of data from a PATROL KM file through the KM2MOF Tool to CIMOM.

Figure 4-1 PATROL KM2MOF Tool Flowchart



Using the PATROL KM2MOF Tool

The PATROL KM2MOF Tool is a Windows command line tool with a minimal user interface. The following bullets give a general idea of how to use the tool. See “Compiling a .km File” on page 4-5 for specific instructions on how to use the PATROL KM2MOF Tool.

- Invoke the PATROL KM2MOF Tool from the operating system prompt.
- Specify the PATROL KM that you want to compile.

The specified **.km** or **.kml** file is read and the CIM PATROL KM model objects are output to the **.mof** file in the MOF (Managed Object Format) language.

A dialog box displays and reports either successful completion or failure. For more information, see “Messages and Errors” on page 4-10.

If successful, the PATROL KM2MOF Tool terminates with a dialog box showing the generated **.mof** filename.

- Load the generated **.mof** file into CIMOM by using the **mofcomp.exe** utility supplied with the Microsoft WMI core components. For specific instructions on this process, see “Loading .mof Files into CIMOM” on page 4-7.

Command Line Options

The two options available when using the PATROL KM2MOF Tool are **-o** and **-l**. They can be used in any order on the KM2MOF command line. For example:

```
km2mof [-o output file] [-l log file] <.km or .kml file>
```

Table 4-1 describes the command line options for the PATROL KM2MOF Tool.

Table 4-1 Command Line Options

Option	Description	Example syntax
-o	This option specifies an output file name.	<code>km2mof -o first.mof inet_host.km</code>
-l	This option suppresses dialog boxes before terminating the KM2MOF Tool and requires the log file name for messages. The log file shows the full list of processed files. This option is useful when you compile .kml files, and it appends information to existing log files rather than overwriting them.	<code>km2mof -l inet.log inet.kml</code>

Compiling a .km File

Summary: To compile the **inet_ftp_server.km** file into the corresponding **inet_ftp_server.mof** file.

To Compile a .km File

Step 1 To invoke the PATROL KM2MOF Tool, type the following entry at the command prompt and press **Enter**:

```
km2mof inet_ftp_server.km
```

Step 2 To direct the **.km** file to a path, specify the path as follows:

```
km2mof d:\patrol3-2\lib\knowledge\inet_ftp_server.km
```

The PATROL KM2MOF Tool creates the **inet_ftp_server.mof** file.

A dialog box displays and reports either successful completion or failure. For more information, see “Messages and Errors” on page 4-10.

If you specified a path in Step 2, the PATROL KM2MOF Tool creates the **inet_ftp_server.mof** file in the appropriate directory.

Compiling a .kml File

In many cases, PATROL KMs are composed of more than one **.km** file. A **.kml** file references all of the **.km** files associated with a specific PATROL KM. The PATROL KM2MOF Tool processes the **.kml** file and all the referenced **.km** files to generate the CIM model objects for the entire PATROL KM.

- » To compile the **inet.kml** file into the corresponding **inet.mof** file, type the following entry at the command line prompt and press **Enter**:

```
km2mof inet.kml
```

The PATROL KM2MOF Tool creates the **inet.mof** file. This file holds the CIM model objects for the entire PATROL KM.

Each **.km** file referenced in **inet.kml** is processed sequentially and compiled into the corresponding model.

Note

All files referenced by the **inet.kml** file must reside in the same directory.

A dialog box appears and reports either successful completion or failure. For more information, see “Messages and Errors” on page 4-10.

If successful, the PATROL KM2MOF Tool terminates with a dialog box which shows the generated **.mof** file name and the list of processed **.km** file names. When the number of processed files is large, multiple messages appear.

Loading .mof Files into CIMOM

You can load the **.mof** files generated by the PATROL KM2MOF Tool into CIMOM using the **mofcomp.exe** utility supplied with the Microsoft WMI core components. The following step describes how to load **.mof** files into CIMOM.

- To load the PATROL KM2MOF Tool generated files into the selected namespace (for example, **cimv2**), type the following entry at the command line prompt and press **Enter**:

```
mofcomp -N:root\cimv2 inet.mof
```

PATROL KM Properties Mapped to MOF Files

Table 4-2 lists PATROL KM properties and shows which files are mapped into **.mof** properties and objects.

Table 4-2 PATROL KM Properties Mapped to MOF (Part 1 of 3)

Data Type	Name	MOF Mapping
Applications	APPLICATIONS	Yes
Application	NT_LOGICAL_DISKS	Yes
StringAttribute	NAME	Yes
BoolAttribute	ACTIVE	Yes
BoolAttribute	SECURITY	Yes
BoolAttribute	PROPAGATE_STATE	Yes
BoolAttribute	CREATE_ICON	Yes
BoolAttribute	SUSPEND_GLOBAL_PARAMS	Yes
BoolAttribute	SHOWINST	Yes
IntAttribute	DISCOVERY_TIME	Yes
ConditionAttribute	DISCOVERY	Yes
StringAttribute	HELP_FILE	No
IntAttribute	HELP_CONTEXT_ID	No

Table 4-2 PATROL KM Properties Mapped to MOF (Part 2 of 3)

Data Type	Name	MOF Mapping
StringAttribute	OK_PICTURE	No
StringAttribute	WRONG_PICTURE	No
Commands	COMMANDS	No
InfoBoxes	INFO_BOX	No
Parameters	PARAMETERS	Yes
Parameter	LDIdDiskQueueLength	Yes
StringAttribute	NAME	Yes
ConditionAttribute	PARAM_TYPE	Yes
BoolAttribute	ACTIVE	Yes
BoolAttribute	MONITOR	Yes
BoolAttribute	CHECK	Yes
StringAttribute	HELP_FILE	No
IntAttribute	HELP_CONTEXT_ID	No
BaseCommands	BASE_COMMAND	Yes
BaseCommands	NT	Yes
StringAttribute	COMPUTER_TYPE	Yes
StringAttribute	COMMAND_TYPE	Yes
SerialNumberStringAttribute	COMMAND_TEXT	No
StringAttribute	TITLE	Yes
StringAttribute	HISTORY_TIME	Yes
IntAttribute	HISTORY_SPAN	Yes
BoolAttribute	HISTORY_LEVEL	Yes
StringAttribute	FORMAT	Yes
ConditionAttribute	OUTPUT	Yes
BoolAttribute	AUTO_RESCALE	Yes
IntAttribute	Y_AXIS_MIN	Yes
IntAttribute	Y_AXIS_MAX	Yes
Ranges	RANGES	Yes
Range	BORDER	Yes

Table 4-2 PATROL KM Properties Mapped to MOF (Part 3 of 3)

Data Type	Name	MOF Mapping
StringAttribute	NAME	Yes
BoolAttribute	ACTIVE	Yes
IntAttribute	MINIMUM	Yes
IntAttribute	MAXIMUM	Yes
ConditionAttribute	STATE	Yes
ConditionAttribute	ALARM_WHEN	Yes
IntAttribute	ALARM_WHEN_N	Yes
Range	ALARM1	Yes
StringAttribute	NAME	Yes
BoolAttribute	ACTIVE	Yes
IntAttribute	MINIMUM	Yes
IntAttribute	MAXIMUM	Yes
ConditionAttribute	STATE	Yes
ConditionAttribute	ALARM_WHEN	Yes
IntAttribute	ALARM_WHEN_N	Yes
Range	ALARM2	Yes
StringAttribute	NAME	Yes
BoolAttribute	ACTIVE	Yes
IntAttribute	MINIMUM	Yes
IntAttribute	MAXIMUM	Yes
ConditionAttribute	STATE	Yes
ConditionAttribute	ALARM_WHEN	Yes
IntAttribute	ALARM_WHEN_N	Yes
Application	NT_LOGICAL_DISKS	Yes
StringAttribute	NAME	Yes
Commands	COMMANDS	No
Parameters	PARAMETERS	No

Messages and Errors

Table 4-3 describes messages and errors that may occur while you use the PATROL KM2MOF Tool.

Table 4-3 PATROL KM2MOF Tool Messages

Message or Error	Description
Usage: km2mof <.km or .kml file> An absolute path can be used for a .km or .kml file. When a .kml file is provided the referenced .km files must reside in the same directory as the .kml file.	No arguments have been provided to the KM2MOF Tool.
km2mof successfully generated the MOF file <.mof file> from the following files <list of .km files>	The KM2MOF Tool terminates successfully.
km2mof <file> Error from the KM parser component (PkmParser1 , <error code>)failed to open km file	Error messages returned from the PkmParser1 COM server (PkmParser1.dll).
km2mof <file> Error from the KM parser component (PkmParser1 , <error code>)<file> (<line number>) syntax error	
km2mof <file> Cannot open file for write: <file>	The KM2MOF Tool is unable to open the destination .mof file.
Warning: File <file> specifies COMPUTERS classes. COMPUTERS class models are not supported.	Result of attempting to compile a .km file for a computer class, which is not supported.
km2mof <file> Localized Parameter <parameter> from localized application <application.host.instance> has a value type that is incompatible with the global application <application>	The localized parameter has a data type incompatible with the corresponding global parameter data type. Example: a gauge parameter localized into a text parameter.
km2mof <file> Error from <source> (code: <code>) <error description>	Microsoft Visual Basic errors such as out of memory, array bounds violations, etc.

PATROL Knowledge Module for WBEM

This chapter provides you with a brief overview of the PATROL[®] Knowledge Module for WBEM.

The following topics are discussed:

Overview	5-3
Loading the PATROL Adapter for WBEM KM	5-4
PATROL Adapter for WBEM KM Application Classes	5-5
PATROL WBEM KM Hierarchy	5-6
About the PATROL Adapter for WBEM Application Classes	5-7
WBEM Application Class	5-8
Parameters	5-8
Menu Commands	5-8
WBEM_CIMOM Application Class	5-9
Parameters	5-9
Menu Commands	5-9
WBEM_NAMESPACE Application Class	5-11
Parameters	5-11
Menu Commands	5-11
WBEM_PROVIDER Application Class	5-12
Parameters	5-13
Menu Commands	5-13
WBEM_PROCESS Application Class	5-14
Parameters	5-14
ProcessColl Parameter	5-15
PageFaultsPerSec Parameter	5-16

PageFileBytes Parameter	5-16
PrivTimePercent Parameter	5-17
ProcessorTimePercent Parameter	5-18
ThreadCount Parameter	5-18
UserTimePercent Parameter	5-19
WorkingSet Parameter	5-20
Menu Commands	5-21
Error Messages	5-21

Overview

This chapter describes the PATROL Adapter for WBEM KM. This KM provides discovery and monitoring functionality for WBEM installations on Windows 2000 (or NT) systems.

The PATROL Adapter for WBEM KM allows you to:

- monitor CIMOM and the PATROL Providers
- configure the PATROL Providers for WBEM and export their information to other applications
- generate and load PATROL models into CIMOM
- browse the CIMOM namespaces

Loading the PATROL Adapter for WBEM KM

Summary: To monitor and manage a KM, you must load the files for that KM into the PATROL Console.

Before You Begin

The PATROL Agent must have already been started.

To Load the PATROL Adapter for WBEM

Step 1 Choose **File =>Load KM...** from the PATROL Console menu bar.

A list of available Knowledge Modules for your site appears.

Step 2 Click the **wbem.kml** file from the list.

The **wbem.kml** file is selected.

Step 3 Click **OK**.

When the KM is loaded, PATROL starts the discovery process. Discovery errors display in the system output window. When complete, a dialog box appears and the KM functionality is available.

PATROL Adapter for WBEM KM Application Classes

This section summarizes the application classes for the PATROL Adapter for WBEM KM. A PATROL KM application class provides menu commands, parameters, and InfoBoxes to effectively monitor an application. For more information about how to use and navigate in a PATROL Knowledge Module, refer to the *PATROL for Windows 2000 User Guide* (or whatever user guide is appropriate for your platform).

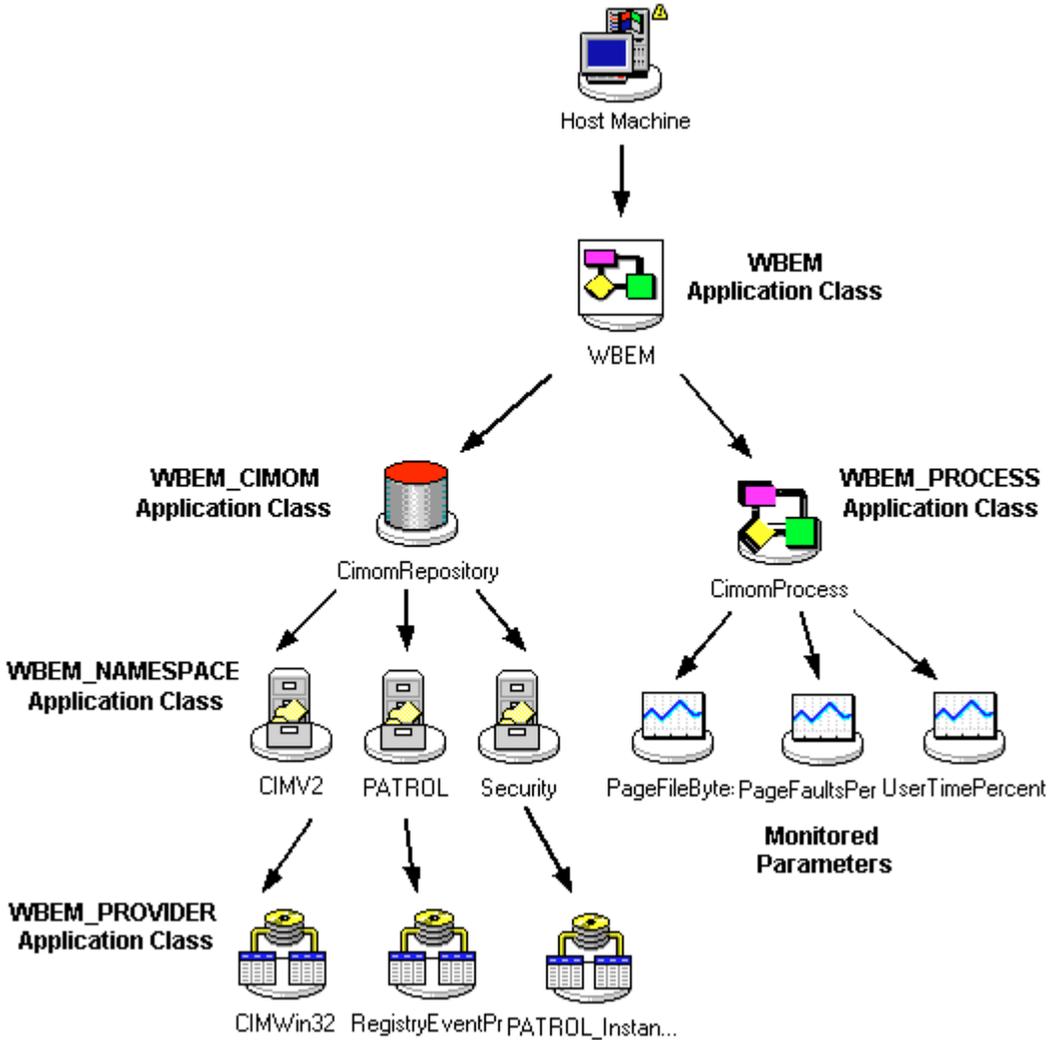
In the PATROL Adapter for WBEM KM, a hierarchy of KM application classes is used to represent and monitor various WBEM objects, including:

- CIMOM Repository
- PATROL Providers
- CIMOM Namespaces
- CIMOM Process

PATROL WBEM KM Hierarchy

The hierarchy of the PATROL Adapter for WBEM KM application objects is outlined in Figure 5-1. Example icons are taken from each application class.

Figure 5-1 WBEM KM Application Class Hierarchy



About the PATROL Adapter for WBEM Application Classes

Table 5-1 contains a summary of the information for each application class in the PATROL Adapter for WBEM KM. Details about each application class can be found in the sections listed under “For More Information”.

Table 5-1 PATROL Adapter for WBEM Application Classes

Icon	Application Class	Icon Name	Definition	For More Information
	WBEM	WBEM	top level WBEM KM icon (contains all other icons)	page 5-8
	WBEM_CIMOM	CIMOMRepository	CIMOM repository icon; represents the repository and contains main CIMOM objects	page 5-8
	WBEM_PROCESS	CIMOMProcess	represents and monitors the CIMOM process	page 5-14
	WBEM_NAMESPACE	<namespace name>	represents a namespace within CIMOM	page 5-11
	WBEM_PROVIDER	<provider name>	represents a provider with a namespace	page 5-12

WBEM Application Class

The WBEM application class performs the PATROL Adapter for WBEM KM discovery process each hour. It coordinates the creation, update and deletion of all the application objects and instances used by the KM and acts as their top level container.

Parameters

The WBEM application class does not provide parameters.

Menu Commands

To access the WBEM application class menu, perform the following action:

- » In the **PATROL Console for Windows**, right-click the WBEM icon and choose KM Commands.

The WBEM application class menu has the following menu commands.

Table 5-2 WBEM Application Class Menu Summary

Menu Command	Action
WBEM KM Configuration: Start CIMOM Service Automatically	CIMOM service is automatically started by the WBEM KM discovery process. The default is TRUE.
WBEM KM Configuration: Start CIMOM Process Monitoring Automatically	CIMOM process monitoring is started automatically. The default is FALSE.
Computer System Report	Retrieves the description of the computer system on which the PATROL Agent is running.

WBEM_CIMOM Application Class

The WBEM_CIMOM application class represents the CIMOM repository. It provides a hierarchical view of some of the important objects stored in the repository such as namespaces, providers and PATROL model objects. It is the container of the WBEM_NAMESPACE application objects.

The WBEM_CIMOM application class is created when a WBEM installation is discovered. It is initially in OFFLINE state. When a connection to the CIMOM repository is established, it changes state to OK.

Parameters

The WBEM_CIMOM application class does not provide parameters.

Menu Commands

To access the WBEM_CIMOM application class menu, perform the following action:

» In the **PATROL Console for Windows**, right-click the **CimomRepository** icon and choose **KM Commands**.

The WBEM_CIMOM application provides CIMOM-specific menu commands.

Table 5-3 WBEM_CIMOM Application Class Menu Summary (Part 1 of 2)

Menu Command	Action
PATROL Provider Configuration => Remote Agents Defaults	Modifies the remote Agent defaults object in the CIMOM PATROL namespace. The PATROL Providers use these settings as defaults when connecting to a PATROL Agent. Refer to Chapter 3, "PATROL Providers" for more information.
PATROL Provider Configuration => Add Remote Agents	Adds remote Agent objects in the CIMOM PATROL namespace. The PATROL Providers use these objects to determine the PATROL Agents that are accessible through CIMOM. The remote Agent object provides connection information for a specific Agent to the PATROL Providers.
PATROL Provider Configuration => Delete Remote Agents	Deletes remote Agent objects from the CIMOM PATROL namespace. Once the object is deleted the related Agent is not accessible through CIMOM. To make this Agent accessible again, use the Add Remote Agents menu command. Refer to Chapter 3, "PATROL Providers" for more information.
PATROL Provider Configuration => Modify Local Agent	Modifies the PATROL_Agent object representing the local Agent in both the CIMV2 and PATROL namespaces. The PATROL Providers use these objects when connecting to the local PATROL Agent to satisfy requests from CIMOM. Refer to Chapter 3, "PATROL Providers" for more information.
PATROL Provider Configuration => Generate Configuration MOF	Generates a MOF file that contains the PATROL Provider configuration information available from CIMOM. This information includes the local PATROL Agent object from the CIMV2 namespace, the remote Agent defaults object and the list of remote Agent objects from the PATROL namespace. Refer to Chapter 3, "PATROL Providers" for more information.

Table 5-3 WBEM_CIMOM Application Class Menu Summary (Part 2 of 2)

Menu Command	Action
CIMOM Service => Start	Starts the CIMOM using the command net start winmgmt . A message pops up to indicate success or failure of the operation.
CIMOM Service => Stop	Stops the CIMOM service using the command winmgmt/kill . A message pops up to indicate success or failure of the operation.

WBEM_NAMESPACE Application Class

The WBEM_NAMESPACE application class represents the namespace objects (__NAMESPACE) in the CIMOM repository. An instance of this application class is created for every namespace that is discovered in the CIMOM root. Instances of WBEM_NAMESPACE are containers of other application class objects such as WBEM_PROVIDER.

Parameters

The WBEM_NAMESPACE application class does not provide parameters.

Menu Commands

The WBEM_NAMESPACE application class provides namespace-specific menu commands.

To access the WBEM_NAMESPACE application class menu, perform the following action:

- » In the **PATROL Console for Windows**, right-click the icon representing an instance of WBEM_NAMESPACE, such as **CIMV2**, and choose **KM Commands**.

The WBEM_NAMESPACE application class menu has the following commands.

Table 5-4 WBEM_NAMESPACE Application Class Menu Summary

Menu Command	Action
Load MOF	Allows you to browse a list of .mof files on the PATROL Agent system, and load selected .mof files into the CIMOM running on the same system.
Load KM	Allows you to browse directories and select .km files to be compiled by the KM2MOF Tool. The resulting MOF file is loaded into CIMOM. As a result, the KM becomes available through CIMOM. The PATROL Provider uses this information to generate the application instance objects for the PATROL KM model.
Unload KM	Allows you to select the .km files currently loaded in CIMOM to unload them. All objects related to the selected kms are also unloaded. The objects that are removed appear in the output task window.
Reports => PATROL Provider Agents	Lists the PATROL Agents that are currently accessed by the PATROL Provider for a CIMOM namespace. These are the PATROL Agents that can be accessed through WBEM. Report details include Agent host name, port, protocol, timeout and username.
Reports => PATROL Provider KMs	Lists the PATROL KMs that are currently loaded into a CIMOM namespace. The definition information of these KMs and the KM-specific CIMOM objects are available through WBEM. Report details include KM name, active flag, discovery type and discovery time.
Reports => WQL Query	Executes any WQL query within a namespace. The query builder edits WQL queries. Click Cancel to terminate the query builder.

WBEM_PROVIDER Application Class

The WBEM_PROVIDER application class represents the provider objects (**__Provider**) in the CIMOM repository. An instance of this application class is created for every provider that is discovered in the corresponding parent namespace.

Parameters

The WBEM_PROVIDER application class does not provide parameters.

Menu Commands

The WBEM_PROVIDER application class supplies provider-specific menu commands.

To access the WBEM_PROVIDER application class menu, perform the following action:

- » In the **PATROL Console for Windows**, right-click an icon representing a WBEM_PROVIDER, such as **PATROL_EventProvider**, and choose **KM Commands**.

The WBEM_PROVIDER application class menu has the following commands.

Table 5-5 WBEM_PROVIDER Application Class Menu Summary

Menu Command	Action
Provider Details	Provides a report with specific provider details. Details include name, CLSID, impersonation level, for each user initialization, initialization reentry, etc.

WBEM_PROCESS Application Class

The WBEM_PROCESS application class represents the processes that are involved in running WBEM. An instance of this application class is created for every relevant process. Currently, only one instance is created to monitor the CIMOM process (**cimom.exe**).

Note

Process monitoring is a relatively costly activity that is not desirable at all times. For this reason, process monitoring, by default, is offline for instances of the WBEM_PROCESS application class. While it is offline, the associated parameters are active, but suspended. You can turn on process monitoring with the Start Monitoring command. For more information, “Menu Commands” on page 5-21.

Parameters

The WBEM_PROCESS application class has various parameters that provide statistical information. Table 5-6 provides parameter information that is used to monitor the PATROL Adapter for WBEM KM.

Table 5-6 WBEM_PROCESS Application Class Parameter Summary (Part 1 of 2)

Parameter	Description	For More Information
ProcessColl	Collects Process statistics from the 'Process' performance objects in the Windows Performance Database.	This is a collector parameter. There is no icon associated with it.
PageFaultsPerSec	Displays the number of page faults for each second by the threads executing in this process. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory.	page 5-16
PageFileBytes	Displays the current number of bytes this process has used in the paging file.	page 5-16
PrivTimePercent	Displays the percentage of elapsed time that this process has spent executing code in privileged mode.	page 5-17

Table 5-6 WBEM_PROCESS Application Class Parameter Summary (Part 2 of 2)

Parameter	Description	For More Information
ProcessorTimePercent	Displays the percentage of elapsed time the selected process used the processor to execute instructions.	page 5-18
ThreadCount	Displays the number of threads currently active in this process.	page 5-18
UserTimePercent	Displays the percentage of elapsed time that this process has spent executing code in user mode.	page 5-19
WorkingSet	Displays the current number of bytes in the working set of this process. The working set is the set of memory pages recently touched by the threads in the process.	page 5-20

ProcessColl Parameter

This parameter collects Process statistics from the 'Process' performance objects in the Windows Performance Database.

Table 5-7 ProcessColl Parameter Defaults

Parameter	ProcessColl
Command Type	PSL
Status	Active
Parameter Type	Collector
Platform	Windows
Application	WBEM_PROCESS
Icon	N/A
Units	N/A
Output Range	N/A
Poll Time	1 minute
Value Set By	N/A

PageFaultsPerSec Parameter

This parameter displays the number of page faults for each second by the threads executing in this process. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory.

Table 5-8 PageFaultsPerSec Parameter Defaults

Parameter	PageFaultsPerSec
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS
Icon	Graph
Units	Seconds
Output Range	Undefined
Poll Time	N/A
Value Set By	ProcessColl

PageFileBytes Parameter

This parameter displays the current number of bytes this process has used in the paging file.

Table 5-9 PageFileBytes Parameter Defaults (Part 1 of 2)

Parameter	PageFileBytes
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows

Table 5-9 PageFileBytes Parameter Defaults (Part 2 of 2)

Application	WBEM_PROCESS
Icon	Graph
Units	Megabytes
Output Range	Undefined
Poll Time	N/A
Value Set By	ProcessColl

PrivTimePercent Parameter

This parameter displays the percentage of elapsed time that the threads of this process have spent executing code in privileged mode.

Table 5-10 PrivTimePercent Parameter Defaults

Parameter	PrivTimePercent
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS
Icon	Graph
Units	Percentage
Output Range	Warning from 90% to 95%. Alarm from 95% to 100%
Poll Time	N/A
Value Set By	ProcessColl

ProcessorTimePercent Parameter

This parameter displays the percentage of elapsed time the selected process used the processor to execute instructions.

Table 5-11 ProcessorTimePercent Parameter Defaults

Parameter	ProcessorTimePercent
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS
Icon	Graph
Units	Percentage
Output Range	Warning from 90% to 95%. Alarm from 95% to 100%
Poll Time	N/A
Value Set By	ProcessColl

ThreadCount Parameter

This parameter displays the number of threads currently active in this process.

Table 5-12 ThreadCount Parameter Defaults (Part 1 of 2)

Parameter	ThreadCount
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS

Table 5-12 ThreadCount Parameter Defaults (Part 2 of 2)

Icon	Graph
Units	Number of threads
Output Range	Undefined
Poll Time	N/A
Value Set By	ProcessColl

UserTimePercent Parameter

This parameter displays the percentage of elapsed time that the threads of this process have spent executing code in user mode.

Table 5-13 UserTimePercent Parameter Defaults

Parameter	UserTimePercent
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS
Icon	Graph
Units	Percentage
Output Range	Warning from 90% to 95%. Alarm from 95% to 100%
Poll Time	N/A
Value Set By	ProcessColl

WorkingSet Parameter

This parameter displays the current number of bytes in the working set of this process. The working set is the set of memory pages touched recently by the threads in the process.

Table 5-14 WorkingSet Parameter Defaults

Parameter	WorkingSet
Command Type	PSL
Status	Active
Parameter Type	Consumer
Platform	Windows
Application	WBEM_PROCESS
Icon	Graph
Units	Megabytes
Output Range	Undefined
Poll Time	N/A
Value Set By	ProcessColl

Menu Commands

To access the WBEM_PROCESS application class menu, perform the following step:

- » In the **PATROL Console for Windows**, right-click the **CimomProcess** icon and choose **KM Commands**.

The **CimomProcess** menu has the following commands.

Table 5-15 WBEM_PROCESS Application Menu Summary

Menu Command	Action
Start Monitoring	Changes the CimomProcess instance state from OFFLINE to OK. This action causes the parameters to resume monitoring the corresponding WBEM process.
Stop Monitoring	Changes the CimomProcess instance state from OK to OFFLINE. This action causes the parameters to suspend monitoring the corresponding WBEM process.

Error Messages

Table 5-16 shows possible error messages when using the PATROL Adapter for WBEM KM.

Table 5-16 PATROL Adapter for WBEM KM Error Messages (Part 1 of 2)

Message	Error
WBEM Services are not found in the Windows registry.	discovery cannot locate WBEM
WBEM PSL API cannot open CIMOM session for the CIMOM server <i>server name</i> .	cannot connect to CIMOM

Table 5-16 PATROL Adapter for WBEM KM Error Messages (Part 2 of 2)

Message	Error
Windows Process performance counter not found.	cannot access process information
cimom.exe process performance information not found.	cannot access process information Ensure that cimom.exe is running.
WBEM PSL API routine (see origin) with arguments	error from PSL API server <arguments list> returned the message below: <msg>
WBEM KM Configuration error	error from PATROL Agent database pconfig returned errno [%1]

Example Scenarios

Several possible scenarios exist for using PATROL Adapter for WBEM. The examples shown here demonstrate requesting PATROL information through WBEM and developing WBEM-enabled PATROL Knowledge Modules (KMs).

The following example scenarios are demonstrated in this appendix:

PATROL KM Model Example	A-2
PATROL Events Example.	A-6
PATROL Agent Model Example	A-12
PATROL KM Development Example	A-15

PATROL KM Model Example

The following example demonstrates requesting PATROL KM-specific information through WBEM. The PATROL KM-Specific model is the easiest way to retrieve this type of information because it uses intuitive variable names. It also allows retrieval of information stored in the PATROL KM that is not available through the agent model.

This example displays information about Windows logical disks using the NT_LOGICAL_DISKS KM-specific class. The **NT_LOGICAL_DISKS.mof** is generated using the PATROL KM2MOF Tool. This example is available in the `%PATROL_HOME%\lib\wbem\examples` directory.

This KM must be loaded by the agents described in the PATROL_Agent instances within CIMOM. The KM is contained within **NT_LOAD.KML**.

To update the .mof file

If you experience problems retrieving the data, you may need to update your **.mof** file. If so, perform the following steps:

Step 1 From `%PATROL_HOME%\lib\knowledge`, type the following:

```
km2mof NT_LOGICAL_DISKS.KM
```

Step 2 Move the resulting file to the `%PATROL_HOME%\lib\knowledge` directory.

Step 3 Open a command prompt and type the following:

```
mofcomp -N:root/patrol NT_LOGICAL_DISKS.MOF
```

Example Code

```
/*
 * File: km_model.cpp
 *-----
 * Description:Retrieves PATROL application instances from CIMOM
```

```

*           using the KM-Specific model. Error checking is minimal.
*           N.B. - the current account must have CIMOM access
*                 - the PATROL NT KM must have been converted to
*                   a MOF and registered with CIMOM
*-----
* Author      :
* Creation date :
* Origin       : WBEM
*
* Modified      : $Date: 1998/09/24 23:16:36 $
* Revision      : $ID$
*
* Copyright (c) 1998 BMC Software, Inc.
*-----
*/

// restricted to Windows NT 4.x
#define _WIN32_WINNT 0x0400

#include <comdef.h>
#include <cstdlib>
#include <stdio.h>
#include <tchar.h>

#include "wbemcli.h"

#define TIMEOUT 2000

HRESULT getProp(IWbemClassObject *pIClassObject, VARIANT &v, LPWSTR
pwszParmName)
{
    VariantClear(&v);
    return pIClassObject->Get(_bstr_t(pwszParmName), 0L, &v, NULL, NULL);
}

void displayDiskInfo(IEnumWbemClassObject *poEnumClassObj)
{
    DWORD dwRetVal;
    do
    {
        // Output properties of disk
        IWbemClassObject *poDiskInst;
        poEnumClassObj->Next(TIMEOUT, 1, &poDiskInst, &dwRetVal);
        if(1 == dwRetVal)
        {
            VARIANT oAgentName, oDiskName, oFreeSpacePercent, oQueueLength,
                oFreeMegabytes, oDiskTimePercent;
            if(SUCCEEDED(getProp(poDiskInst, oAgentName, L"AgentName")) &&

```

```

        SUCCEEDED(getProp(poDiskInst, oDiskName, L"Name")) &&
        SUCCEEDED(getProp(poDiskInst, oFreeSpacePercent,
            L"LDldFreeSpacePercent")) &&
        SUCCEEDED(getProp(poDiskInst, oQueueLength,
            L"LDldDiskQueueLength")) &&
        SUCCEEDED(getProp(poDiskInst, oFreeMegabytes,
            L"LDldFreeMegabytes")) &&
        SUCCEEDED(getProp(poDiskInst, oDiskTimePercent,
            L"LDldDiskTimePercent"))
    {
        _tprintf(_T("\nAgent: %s\t\tDisk: %s\n\tFree Space Percent:
            %f\n\tDisk Queue Length: %f\n\tTime Percent: %f\n\tFree Mb:
            %f"),
            _bstr_t(V_BSTR(&oAgentName)),
            _bstr_t(V_BSTR(&oDiskName)),
            V_R4(&oFreeSpacePercent),
            V_R4(&oQueueLength),
            V_R4(&oDiskTimePercent),
            V_R4(&oFreeMegabytes));
    }
    else
    {
        _tprintf(_T("\nError - Unable to get application instance
            properties."));
    }
    VariantClear(&oFreeSpacePercent);
    VariantClear(&oQueueLength);
    VariantClear(&oFreeMegabytes);
    VariantClear(&oDiskTimePercent);

    if(poDiskInst)
    {
        poDiskInst->Release();
        poDiskInst = NULL;
    }
}
}
while(1 == dwRetVal);
}

int _tmain(int argc, LPTSTR *argv, LPTSTR *envp)
{
    CoInitializeEx(NULL, COINIT_MULTITHREADED);

    // lower the security level so CIMOM can contact interfaces in this app
    CoInitializeSecurity(NULL, -1, NULL, NULL, RPC_C_AUTHN_LEVEL_NONE,
        RPC_C_IMP_LEVEL_IMPERSONATE, NULL, EOAC_NONE, 0);

```

```

// connect to WBEM
IUnknown* pIUnknown;
CoCreateInstance(CLSID_WbemLocator, NULL, CLSCTX_ALL, __uuidof(IUnknown),
reinterpret_cast<void**>(&pIUnknown));

IWbemLocator *pIWbemLocator;
pIUnknown->QueryInterface(IID_IWbemLocator, (PVOID *)&pIWbemLocator);
pIUnknown->Release();

IWbemServices *pIWbemServices = NULL;
HRESULT hResult = pIWbemLocator->ConnectServer(
_bstr_t("root/patrol"), // Namespace
NULL, // use current Userid
NULL, // Password
_bstr_t("MS\\0x409"), // Locale
0, // flags
NULL, // Authority
NULL, // Context
&pIWbemServices);
pIWbemLocator->Release();
if(!SUCCEEDED(hResult))
{
    CoUninitialize();
    return -1; // CIMOM not running?
}

// retrieve all instances of PATROL_KM_NT_LOGICAL_DISKS where the free
// space percent is low
IEnumWbemClassObject *poEnumClassObj = NULL;
hResult = pIWbemServices->ExecQuery(
_bstr_t("WQL"),
_bstr_t("select * from PATROL_KM_NT_LOGICAL_DISKS where
LDldFreeSpacePercent_Status = \"ALARM\""),
0L,
NULL,
&poEnumClassObj);
if(SUCCEEDED(hResult))
    displayDiskInfo(poEnumClassObj);
else
    _tprintf(_T("\nUnable to retrieve class enumeration.));

if (poEnumClassObj)
{
    poEnumClassObj->Release();
    poEnumClassObj = NULL;
}
return ERROR_SUCCESS;
}

```

PATROL Events Example

This example is available in the `%PATROL_HOME%\lib\wbem\examples` directory.

The following example demonstrates receiving PATROL events through WMI. The example registers with CIMOM as a consumer of events from `PATROL_Event`. Each received event is displayed. If a host corresponding to an instance of `PATROL_Agent` within the `root/patrol` namespace cannot be contacted, an event is generated and the fields have the following values:

```
AgentName == <agent>
EventDescription == Unable to connect to agent.
Diary == <agent> User: <user name> Encrypted Password: <password> Port:
<port> Timeout: <request timeout> Protocol: <protocol>
Severity == 5
Application == WBEM EventProvider
Owner == WBEM Account
Status == 1
Type == 3
```

Event Subscriber Example Code

The following is an example of subscribing to CIMOM to receive PATROL events.

```
/*
 * File: events.cpp
 * -----
 * Description:Subscribes to PATROL events.
 * -----
 * Author      :
 * Creation date :
 * Origin      : WBEM
 *
 * Modified    : $Date: 1998/09/24 14:36:08 $
 * Revision    : $ID$
 *
 * Copyright (c) 1998 BMC Software, Inc.
 * -----
 */

// restricted to Windows NT 4.x
#define _WIN32_WINNT0x0400
```

```

#include <stdio.h>
#include <fstream.h>
#include <tchar.h>
#include <comdef.h>

#include "wbemcli.h"

#include "sink.h"

//*****
*
// Specify WQL query to indicate the set of events we want
//*****
*
int _tmain(int argc, LPTSTR *argv, LPTSTR *envp)
{
    CoInitializeEx(0, COINIT_MULTITHREADED);

    IWbemLocator *pIWbemLocator;
    CoCreateInstance(CLSID_WbemLocator, 0, CLSCTX_INPROC_SERVER,
        IID_IWbemLocator, (LPVOID *) &pIWbemLocator);

    // allow CIMOM to send events to the sink
    CoInitializeSecurity(NULL, -1, NULL, NULL, RPC_C_AUTHN_LEVEL_NONE,
        RPC_C_IMP_LEVEL_IDENTIFY, NULL, EOAC_NONE, 0);

    // Connect to CIMOM.
    IWbemServices *pIServices = 0;
    HRESULT hRes = pIWbemLocator->ConnectServer(
        _bstr_t("root/patrol"),
        NULL,
        NULL,
        0,
        0,
        0,
        0,
        &pIServices
    );
    pIWbemLocator->Release();
    if(!SUCCEEDED(hRes))
    {
        CoUninitialize();
        return -1; // CIMOM not running?
    }

    // Create a new sink then wait until the user hits ENTER to stop.
    CPATROLEventSink *poEventSink = new CPATROLEventSink;

```

```

if(SUCCEEDED(pIServices->ExecNotificationQueryAsync(
    _bstr_t("WQL"),
    _bstr_t("select * from PATROL_Event"),
    0,
    0,
    poEventSink))
{
    TCHAR buf[8];
    _getts(buf);
    pIServices->CancelAsyncCall(poEventSink);
}
else
{
    CoUninitialize();
    return -1; // Unable to execute the event query
}

poEventSink->Release();
pIServices->Release();

CoUninitialize();

return ERROR_SUCCESS;
}

```

Event Sink Example Code

The following code is an example of implementing the sink, which is used by the subscriber to retrieve event information from CIMOM.

```

/*
 * File: sink.cpp
 *-----
 * Description:Implementation of WBEM Event Sink.
 *-----
 * Author      :
 * Creation date :
 * Origin      : WBEM
 *
 * Modified    : $Date: 1998/09/24 14:36:09 $
 * Revision    : $ID$
 *
 * Copyright (c) 1998 BMC Software, Inc.
 *-----
 */

```

```

#include <stdio.h>
#include <tchar.h>
#include <comdef.h>

#include "wbemidl.h"

#include "sink.h"

STDMETHODIMP CPATROLEventSink::QueryInterface(REFIID riid, LPVOID * ppv)
{
    if (IID_IUnknown==riid || IID_IWbemObjectSink == riid)
    {
        *ppv = (IWbemEventProvider *) this;
        AddRef();
        return NOERROR;
    }
    *ppv = 0;

    return ResultFromScode(E_NOINTERFACE);
}

ULONG CPATROLEventSink::AddRef()
{
    return ++m_cRef;
}

ULONG CPATROLEventSink::Release()
{
    if (0 != --m_cRef)
        return m_cRef;

    delete this;
    return 0;
}

//*****
*
// Receives an event notification.
//*****
*

HRESULT CPATROLEventSink::Indicate(
long lObjectCount,
IWbemClassObject __RPC_FAR * __RPC_FAR *ppObjArray
)
{
    // these fields correspond to the PATROL_Event class attributes
    static oFieldEntry goFieldEntries[] =

```

```

{
    {_T("TimeStamp"), CIM_DATETIME},
    {_T("Identity"), CIM_UINT32},
    {_T("AgentName"), CIM_STRING},
    {_T("Arguments"), CIM_STRING},
    {_T("ClassName"), CIM_STRING},
    {_T("EventDescription"), CIM_STRING},
    {_T("Diary"), CIM_STRING},
    {_T("CatalogName"), CIM_STRING},
    {_T("LifeExpectancy"), CIM_STRING},
    {_T("HandlerUserID"), CIM_STRING},
    {_T("Severity"), CIM_UINT8},
    {_T("Application"), CIM_STRING},
    {_T("Owner"), CIM_STRING},
    {_T("Source"), CIM_STRING},
    {_T("Status"), CIM_UINT8},
    {_T("Type"), CIM_UINT8}
    {_T("Port"), CIM_UINT32}
};

// Get the info from the object.
for (long i = 0; i < lObjectCount; i++)
{
    IWbemClassObject *pObj = ppObjArray[i];

    // If here, we know the object is one of the kind we asked for.
    VARIANT oValue;
    for(UINT j=0;j<sizeof(goFieldEntries)/sizeof(goFieldEntries[0]);j++)
    {
        CIMTYPE lCIMFormatType;
        HRESULT hResult = pObj->Get(_bstr_t(goFieldEntries[j].pszFieldName),
            0, &oValue, &lCIMFormatType, 0);
        if(WBEM_S_NO_ERROR == hResult);
        {
            switch(lCIMFormatType)
            {
                case CIM_DATETIME:
                case CIM_STRING:
                    _tprintf(_T("%u: %s == %s\n"), j,
                        goFieldEntries[j].pszFieldName,
                        LPSTR(_bstr_t(oValue.bstrVal)));
                    break;
                case CIM_UINT8:
                    _tprintf(_T("%u: %s == %d\n"), j,
                        goFieldEntries[j].pszFieldName, oValue.bVal);
                    break;
                case CIM_UINT16:

```

```

        _tprintf(_T("%u: %s == %d\n"), j,
            goFieldEntries[j].pszFieldName, oValue.iVal);
        break;
    case CIM_UINT32:
        _tprintf(_T("%u: %s == %d\n"), j,
            goFieldEntries[j].pszFieldName, oValue.lVal);
        break;
    default:
        _tprintf(_T("\nUnknown data type"));
    }
    VariantClear(&oValue);
}
}
_tprintf(_T("***\n"));
}

return WBEM_NO_ERROR;
}

HRESULT CPATROLEventSink::SetStatus(long lFlags, HRESULT hResult, BSTR
strParam,
IWbemClassObject __RPC_FAR *pObjParam)
{
    return WBEM_NO_ERROR;
}

```

PATROL Agent Model Example

The following example demonstrates requesting PATROL application instance information through WMI. This example displays status information for each instance of PATROL_AppInstance. The PATROL Agent model is the easiest way to retrieve this type of information since it spans all of the PATROL KMs. This example is available in the `%PATROL_HOME%\lib\wbem\examples` directory.

Example Code

```
/*
 * File: agent_model.cpp
 * -----
 * Description:Retrieves PATROL application instances from CIMOM
 *              using the Agent model. Error checking is minimal.
 *              N.B. the current account must have CIMOM access
 * -----
 * Author      :
 * Creation date :
 * Origin      : WBEM
 *
 * Modified    : $Date: $
 * Revision    : $ID$
 *
 * Copyright (c) 1998 BMC Software, Inc.
 * -----
 */

// restricted to Windows NT 4.x
#define _WIN32_WINNT 0x0400

#include <comdef.h>
#include <cstdlib>
#include <stdio.h>
#include <tchar.h>

#include "wbemcli.h"

#define TIMEOUT 2000

HRESULT getProp(IWbemClassObject *pIClassObject, VARIANT &v, LPWSTR
pwszParmName)
{
```

```

VariantClear(&v);
return pIClassObject->Get(_bstr_t(pwszParmName), 0L, &v, NULL, NULL);
}

int _tmain(int argc, LPTSTR *argv, LPTSTR *envp)
{
    CoInitializeEx(NULL, COINIT_MULTITHREADED);

    // lower the security level so CIMOM can contact interfaces in this app
    CoInitializeSecurity(NULL, -1, NULL, NULL, RPC_C_AUTHN_LEVEL_NONE,
        RPC_C_IMP_LEVEL_IMPERSONATE, NULL, EOAC_NONE, 0);

    // connect to WBEM
    IUnknown* pIUnknown;
    CoCreateInstance(CLSID_WbemLocator, NULL, CLSCTX_ALL, __uuidof(IUnknown),
        reinterpret_cast<void**>(&pIUnknown));

    IWbemLocator *pIWbemLocator;
    pIUnknown->QueryInterface(IID_IWbemLocator, (PVOID *)&pIWbemLocator);
    pIUnknown->Release();

    IWbemServices *pIWbemServices = NULL;
    HRESULT hResult = pIWbemLocator->ConnectServer(
        _bstr_t("root/patrol"), // Namespace
        NULL, // use current Userid
        NULL, // Password
        _bstr_t("MS\\0x409"), // Locale
        0, // flags
        NULL, // Authority
        NULL, // Context
        &pIWbemServices);
    pIWbemLocator->Release();
    if(!SUCCEEDED(hResult))
    {
        CoUninitialize();
        return -1; // CIMOM not running?
    }

    // enumerate all of the PATROL application instances
    IEnumWbemClassObject *poEnumObject;
    HRESULT hRetVal = pIWbemServices->CreateInstanceEnum(
        _bstr_t("PATROL_AppInstance"), WBEM_FLAG_FORWARD_ONLY, NULL,
        &poEnumObject);
    if(!SUCCEEDED(hRetVal) || !poEnumObject)
    {
        CoUninitialize(); // PATROL_AppInstance class not registered?
        return -1;
    }
}

```

```

DWORD dwRetVal;
do
{
    // Output selected properties for each application instance
    IWbemClassObject *poMgmtAppInst;
    hRetVal = poEnumObject->Next(TIMEOUT, 1, &poMgmtAppInst, &dwRetVal);
    if(1 == dwRetVal)
    {
        VARIANT oMgmtApp, oWorstParm, oMgmtAppStatus;
        if(SUCCEEDED(getProp(poMgmtAppInst, oMgmtApp, L"MgmtAppName")) &&
            SUCCEEDED(getProp(poMgmtAppInst, oWorstParm, L"WorstParameter"))&&
            SUCCEEDED(getProp(poMgmtAppInst, oMgmtAppStatus, L"Status")))
        {
            _tprintf(_T("\nApplication: %s\n\tStatus: %s\n\tWorst Parameter:
                %s"), LPTSTR(_bstr_t(oMgmtApp.bstrVal)),
                LPTSTR(_bstr_t(oMgmtAppStatus.bstrVal)),
                LPTSTR(_bstr_t(oWorstParm.bstrVal)));
        }
        else
        {
            _tprintf(_T("\nError - Unable to get application instance
                properties."));
        }
        VariantClear(&oMgmtApp);
        VariantClear(&oWorstParm);
        VariantClear(&oMgmtAppStatus);

        if (poMgmtAppInst)
        {
            poMgmtAppInst->Release();
            poMgmtAppInst = NULL;
        }
    }
}
while(1 == dwRetVal);

if (poEnumObject)
{
    poEnumObject->Release();
    poEnumObject = NULL;
}

CoUninitialize();
return ERROR_SUCCESS;
}

```

PATROL KM Development Example

You can use the PATROL Adapter for WBEM to write PATROL KMs that are WBEM-enabled. These PATROL KMs use the PSL API for WBEM to access CIMOM. This is useful when a PATROL KM needs information that is already available within the CIMOM repository. It also avoids duplication of the information collection process.

The WBEM Adapter PSL API allows a PATROL KM to

- connect to CIMOM
- run queries to browse CIMOM objects
- access CIMOM objects directly
- create, delete and modify CIMOM objects
- subscribe to CIMOM events (PATROL KM acts as an event consumer)

Example WBEM-Enabled PATROL Knowledge Module

The PATROL Adapter for WBEM KM is a good example of a WBEM-enabled KM. The following report is a sample PSL program showing how to execute queries against CIMOM. This program implements the PATROL Adapter for WBEM KM Computer System Report.

The Computer System Report shows information from the Win32_ComputerSystem and Win32_Service CIMOM objects. These objects are accessed using the PSL API WQL query support. Two queries are used:

- one query retrieves system details from the Win32_ComputerSystem class
- the second query retrieves service details from the Win32_Service class

The data from the two queries is then formatted and displayed in a response() dialog.

Computer System Report

```
# File: WBEMCmdSysReport.psl
#####
# Description: query WBEM and generate the computer
#              system report
#####
# Creation date:
# Copyright (c) 1998 BMC Software, Inc.
#####

requires pwbemlib;

# open session to WBEM CIMOM
Namespace = "\\.\root\cimv2";
Session = wbem_open(Namespace);
if (Session <= 0)
{
    printf( "WBEM API ERROR <%s>\n", wbem_error_str(wbem_error()));
    exit;
}

# delimiters for wbem_query() result set
RD = "\A"; # row delimiter
CD = "\B"; # column delimiter
ED = "\C"; # array element delimiter

# query Win32_ComputerSystem object
Qry = "select "
      . " Name, PrimaryOwnerName, SystemType, "
      . " UserName, Roles, SystemStartupDelay, SystemStartupOptions "
      . "from Win32_ComputerSystem";
Buf = wbem_query( Session, Qry, RD, CD, ED);
Error = wbem_error( Session);
if (Error)
{
    printf( "WBEM API ERROR <%s>\n", wbem_error_str(wbem_error( Session)));
    wbem_close( Session);
    exit;
}

ComputerSystem = "";

# only 1 row returned by the query
Row = wbem_query_nth_row( Buf, 1);

# name
ComputerSystem = ComputerSystem .
```

```

sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 1), wbem_query_nth_col( Buf, Row, 1));

# primary owner
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 2), wbem_query_nth_col( Buf, Row, 2));

# system type
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 3), wbem_query_nth_col( Buf, Row, 3));

# username
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 4), wbem_query_nth_col( Buf, Row, 4));

# roles (array)
Roles = wbem_query_nth_col( Buf, Row, 5);
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 5), wbem_query_nth_elem( Buf, Roles, 1));
Count = 2;
while (wbem_query_nth_elem( Buf, Roles, Count) != "")
{
    ComputerSystem = ComputerSystem .
        sprintf("%-20s   %s\n", "", wbem_query_nth_elem( Buf, Roles, Count));
    Count = Count + 1;
}

# system startup delay
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 6), wbem_query_nth_col( Buf, Row, 6));

# system startup options (array)
StartupOptions = wbem_query_nth_col( Buf, Row, 7);
ComputerSystem = ComputerSystem .
sprintf("%-20s : %s\n",
        wbem_query_nth_title( Buf, 7), wbem_query_nth_elem( Buf, StartupOptions,
1));
Count = 2;
while (wbem_query_nth_elem( Buf, StartupOptions, Count) != "")
{
    ComputerSystem = ComputerSystem .
        sprintf("%-20s   %s\n", "", wbem_query_nth_elem( Buf, StartupOptions,
Count));
}

```

```

    Count = Count + 1;
}

# run a second query to retrieve services information
Qry = "select "
    . " Name, Caption, State, StartMode, PathName "
    . "from Win32_Service";
Buf = wbem_query( Session, Qry, RD, CD);
Error = wbem_error( Session);
if (Error)
{
    printf( "WBEM API ERROR <%s>\n", wbem_error_str(wbem_error( Session)));
    wbem_close( Session);
    exit;
}

ComputerSystem = ComputerSystem .
sprintf("%-20s : %d", "Services", wbem_query_row_count( Buf));
Services = "";

# services titles
Services = Services .
sprintf("%-15s %-25s %-10s %-10s %s\n", wbem_query_nth_title( Buf, 1),
    wbem_query_nth_title( Buf, 2), wbem_query_nth_title( Buf, 3),
    wbem_query_nth_title( Buf, 4), wbem_query_nth_title( Buf, 5));
Services = Services .
sprintf("%-15s %-25s %-10s %-10s %s\n",
    "-----",
    "-----",
    "-----",
    "-----",
    "-----");

# retrieve rows
Count = 1;
Row = wbem_query_nth_row( Buf, Count);
while (Row != "")
{
    Services = Services .
    sprintf("%-15s %-25s %-10s %-10s %s\n",
        substr( wbem_query_nth_col( Buf, Row, 1), 1, 15),
        substr( wbem_query_nth_col( Buf, Row, 2), 1, 25),
        wbem_query_nth_col( Buf, Row, 3),
        wbem_query_nth_col( Buf, Row, 4),
        wbem_query_nth_col( Buf, Row, 5));
    Count = Count + 1;
}

```

```
    Row = wbem_query_nth_row( Buf, Count);  
}  
  
wbem_close( Session);  
  
# display report  
response( "COMPUTER SYSTEM REPORT", "", "o=Close,c=",  
    [R_LABEL, ComputerSystem],  
    [R_LIST_SINGLE_ND, Services]);
```

Troubleshooting

This appendix describes possible problems that you may encounter while using the PATROL Adapter for WBEM. Solutions to the problems are provided:

Possible Problems and Solutions B-2

Possible Problems and Solutions

Table B-1 describes possible problems that you may encounter while using the PATROL Adapter for WBEM. Solutions to the problems are provided.

Table B-1 Possible Problems and Solutions

Problem	Solution
The mofcomp.exe program crashes during installation.	If the platform SDK is installed and its path precedes the WBEM path in the PATH environment setting, mofcomp.exe can crash due to incompatibility with the PATROL Adapter for WBEM. To resolve this problem, move the WBEM directory (typically, <windows directory>\system32\wbem) ahead of the platform SDK in the PATH.
You cannot connect to CIMOM.	Ensure that the WMI service is started and that the account used by the client is permitted to access WMI. By default, local administrators have full access to the local CIMOM as do members of the local administrators group.
When querying the PATROL models, CIMOM returns an error that it is unable to load the provider.	Ensure that the PATROL Provider (pwinstpr.dll) is installed in %PATROL_HOME%\lib\wbem\tools and that it is properly registered. (Refer to regsrv32 pwinstpr.dll). Also ensure that it is not registered twice in the CIM repository.
When querying the PATROL models, objects from a PATROL Agent monitored through CIMOM are not available.	Ensure that a PATROL_Agent object exists for that Agent in the PATROL namespace and that it has a valid user name, port, and password.
When querying the PATROL model KM-specific dynamic classes, some parameters are not available.	Remember that collector and no-output parameters are not provided in the KM-specific dynamic classes. However, their definition is available from the corresponding instances of PATROL_KM_Parameter.
The PATROL Instance Provider does not return instances for the monitored PATROL Agents.	Ensure that the corresponding PATROL_Agent objects have the correct user name, port, and password; make sure those Agents are reachable through the PATROL Command Line Interface (CLI) (patrolcli command).
You are unable to uninstall the WBEM Adapter.	Uninstallation is not possible without removing the PATROL Agent. The WBEM Adapter removes earlier versions at the start of the installation process.

Security

This appendix describes the security interface for the PATROL Adapter for WBEM and its configuration.

The following topics are discussed:

Firewalls	C-2
Passwords	C-2
COM/DCOM Configuration	C-3
PATROL Single Sign-On	C-3
Support for PATROL Single Sign-On	C-3
Changing the Account for the PEM COM Server	C-4
Terminating the PEM COM Server Process	C-6

Firewalls

By default the PATROL Adapter for WBEM uses a semi-random port to communicate with PATROL Agents with TCP or UDP. If there are one or more firewalls between the Windows 2000 (or Windows NT) machine running the PATROL Adapter for WBEM and the PATROL Agents it communicates with, you need to ensure the PATROL Adapter for WBEM uses a specific port.

The port setting is read for each new PATROL Agent connection so it is best to set it immediately after system startup before the first call is made to the PATROL Adapter for WBEM. You can set the port by using the registry settings defined in Appendix D, “Registry Settings.”

The firewalls between the server and the agents must be configured to permit traffic on this port. Otherwise attempts to connect to the PATROL Agent are unsuccessful. The Windows Event Viewer can be used to examine failed connection attempts.

Passwords

Passwords are stored in encrypted form within the CIM Repository. WMI controls access to the repository and the following users have access to WMI:

- local and remote administrators
- members of the **WBEM Users** group
- users added explicitly using the WMI user manager

The repository is stored in a proprietary binary format in the **Wbem\Repository** subdirectory under the **system** directory.

COM/DCOM Configuration

To preserve resources, the PEM COM server maintains one connection to each PATROL Agent. Clients of the server share this connection, so it may be necessary to secure the COM server, which can be done using the standard Microsoft **DCOMCNFG.exe** utility. The AppID of the COM server is **BMC PEM COM server**.

PATROL Single Sign-On

This plug-in provides account authentication between PATROL products running within the same Active Directory domain on Windows 2000.

When a PATROL product is enabled by this plug-in, it can authenticate the credentials of a Windows account instead of requiring PATROL-specific user names and passwords. As the name suggests, the Single Sign-On plug-in means that a user is prompted only once for a user name and password at the time the Windows session is started.

See the *PATROL Agent Reference Manual* for more information about the PATROL Single Sign-On plug-in.

Support for PATROL Single Sign-On

While PATROL Adapter for WBEM supports the PATROL Single Sign-On plug-in, you must satisfy two conditions to enable this support. The first condition is that the PATROL Single Sign-On plug-in is installed on both the computer running the PATROL Agent and the computer hosting the PATROL Adapter for WBEM. The PATROL installation guides can help you if you need to install this plug-in.

The other condition is that the PEM COM server should use a domain account that has access privileges to all required PATROL Agents. If the account has sufficient privileges, the PEM COM server can forward PATROL information to WMI. If the account does not have the right privileges, you can use the instructions in this section to change which account the PEM COM server uses.

Changing the Account for the PEM COM Server

Summary: This task specifies which account launches the PEM COM server. The PATROL Single Sign-On plug-in requires a domain account that has access to all required PATROL Agents.

Before You Begin

Determine that the following requirements are complete.

- Install the PATROL Single Sign-On plug-in on the computer hosting the PATROL Adapter for WBEM and all computers running the PATROL Agent for Windows 2000.
- Verify that the Windows account, that you intend to use in this task, has access to all required PATROL Agents.

To Change the Account Used by the PEM COM Server

Step 1 Open a command prompt, and enter **dcomcnfg**.

The Distributed COM Configuration Properties window opens.

Step 2 Under Applications, click **BMC PEM COM Server**.

Step 3 Click **Properties**.

The BMC PEM COM Server Properties window opens.

Step 4 Click the Identity tab.

Step 5 Click the This User radio button.

Step 6 Complete the following fields: User, Password, and Confirm Password.

Note

The information that you enter in the previous step must belong to a domain account that has access to all required PATROL Agents.

Step 7 Click **OK** to close the BMC PEM COM Server Properties window.

Step 8 Click **OK** to close the Distributed COM Configuration Properties window.

Note

If the PEM COM server process (pemsrv.exe) is running, applying the previous instructions will not result in any changes until this process is terminated. For these changes to take effect quickly, see the following task, “Terminating the PEM COM Server Process” on page C-6.

Terminating the PEM COM Server Process

Summary: If you have configured the PEM COM server using the previous instructions, you may need to terminate the PEM COM server process to make these changes take effect quickly.

Before You Begin

Determine that the following requirements are complete.

- Complete the task in “Changing the Account for the PEM COM Server” on page C-4.
- Verify that the PEM COM server process (pemsrv.exe) is running, which can be done with the Windows Task Manager.

To Terminate the PEM COM Server Process

- Step 1** Open a command prompt, and enter **net stop winmgmt**.
- Step 2** Check whether the PEM COM server process (pemsrv.exe) is running, and perform one of the actions in the following table.

Process Status	Action
pemsrv.exe is not running	enter the net start winmgmt command at a command prompt
pemsrv.exe is running	restart your computer to terminate the process

Registry Settings

This appendix describes the registry settings for the PATROL Adapter for WBEM.

The following table is presented:

Registry Settings D-2

Registry Settings for the PATROL Adapter for WBEM

The root registry key for use with the PATROL Adapter for WBEM is **HKEY_LOCAL_MACHINE/SOFTWARE/BMC Software/PATROL Adapter for WBEM/1.2.00**.

Warning

Before changing any settings, stop the Windows Management service and the **PEMSVR.exe** process.

The registry items and descriptions are shown in Table D-1.

Table D-1 Registry Settings (Part 1 of 2)

Registry Item	Description
Logging Options	
LogOutputLevel	Filters log messages. The values are: 1 - no output 2 - errors only (default) 3 - warnings and errors 4 - info, warnings, and errors 5 - everything
OutputToApplicationLog	Toggles output to text log. The values are: 1 - output to log in temporary directory 0 - prevent output to log (default)
OutputToNTEventLog	Toggles output to Windows event log. The values are: 1 - output to Windows application event log (default) 0 - prevent output to log
OutputToDebugger	Toggles output to debugger. The values are: 1 - output to debugger 0 - prevent output to debugger (default)
Event Provider Options	
PEMServerPollPeriod_Minutes	The time period, in minutes, between checks of the activity of the PEM COM server. If a check fails, an attempt is made to restart the server so that events can be received. (Default is 5 minutes.)

Table D-1 Registry Settings (Part 2 of 2)

Registry Item	Description
COM Server Options	
LocalPort	The port used by the PEM COM server to access agents. Useful for avoiding firewalls. (Default is 0.) If you are using this setting, ensure that all PATROL_Agent instances are using the UDP protocol.
MaxPingsPerSecond	The maximum number of pings issued per second by the PEM COM server during PATROL Agent discovery. (Default is 100 pings per second.)

Glossary

agent	<i>See</i> PATROL Agent.
API	<i>See</i> Application Program Interface.
Application Program Interface (API)	A set of protocols, routines and tools for building software applications.
association class	A class that describes a relationship between two classes or between instances of two classes.
CIM	<i>See</i> Common Information Model.
CIM Object Manager (CIMOM)	A component in the CIM management infrastructure that provides a collection and manipulation point for managed objects stored in the repository. It facilitates gathering and manipulating information about these managed objects.
CIM Repository	A central storage area managed by CIMOM. This repository contains static classes and instances. Dynamic information is always supplied on demand by a provider.
CIMOM	<i>See</i> CIM Object Manager.

CIMV2	The namespace created by Microsoft to contain the classes and instances that represent a Win32 environment, such as Win32_LogicalDisk and Win32_OperatingSystem. It is used to store local machine information only.
class	A definition for a particular kind of object. A class definition presents such things as variables and methods belonging to the class and its instances.
COM	<i>See</i> Component Object Model.
COM Server	A process which services requests, either local or remote, for COM interfaces.
Common Information Model (CIM)	An object-oriented model that describes how to represent real-world managed objects using the concepts of classes and instances. CIM was established by the DMTF.
Component Object Model (COM)	A specification for building binary components which support one or more well-defined interfaces for use on a single machine.
DCOM	<i>See</i> Distributed Component Object Model.
Distributed Component Object Model (DCOM)	A specification for allowing remote access to COM components.
Distributed Management Task Force (DMTF)	An industry organization that strives to lead the development, adoption and unification of management standards and initiatives for desktop, enterprise and Internet environments.
DMTF	<i>See</i> Distributed Management Task Force.
dynamic	Describes a CIM class definition or instance that is supplied by a provider at runtime. Providers fetch this information since it is likely to change often. Static information, on the other hand, is stored in the CIM repository.
heartbeat	The interval (in seconds) at which the PATROL Console checks to see if the PATROL Agent is still running. The longer the interval, the lower the network traffic.

KM	<i>See</i> PATROL Knowledge Module.
managed object	Physical or logical enterprise components that are modeled using CIM. For example, a managed object can be hardware such as a cable or system chassis, software such as a database application, or other logical entities such as computer systems, files, and devices. Management applications can access managed objects through CIM Object Manager.
Managed Object Format (MOF)	A compiled language for defining classes and instances, which can be added to the CIM repository.
management application	An application or service that processes or displays data from managed objects or change the state of managed objects.
method	A function describing the behavior of a class.
model	A collection of class definitions that describe managed objects in a particular environment, such as networking, applications, and systems. Model is synonymous to schema.
MOF	<i>See</i> Managed Object Format.
namespace	A unit for grouping classes and instances to control their scope and visibility. Namespaces are not physical locations; they are more like logical databases containing specific classes and instances.
PATROL Agent	The core component of the PATROL architecture. It can be used to monitor and manage host computers, applications, system resources, and more. For more information, see the PATROL Agent Reference Manual.
PATROL Agent model	Class definitions and associations for objects that are managed by a PATROL Agent. <i>See also</i> model.

PATROL Event Manager (PEM)

A PATROL component that you can use to view and manage events that occur on monitored system resources and that are sent by PATROL Agents. You can access the PEM from the PATROL Console or use it as a stand-alone facility. It works with the PATROL Agent and user-specified filters to provide a customized view of events.

PATROL Event model

A single CIM class, `PATROL_Event`, defines the event objects created by PATROL. `PATROL_Event` is a subclass to the `__ExtrinsicEvent` class defined by Microsoft.

PATROL Event Provider

A provider, created by BMC Software, that registers with PATROL to receive notification of PATROL events as they occur. When a CIMOM event consumer registers an interest in instances of the `PATROL_Event` class, CIMOM passes the subscription request to the Event Provider which, in turn, subscribes to the corresponding PATROL Agent events. *See also* provider.

PATROL Instance Provider

A provider, created by BMC Software, that is responsible for delivering instances of PATROL classes. It creates each instance by retrieving information from PATROL Agents via the PEM COM server and then sends the instance to CIMOM. *See also* provider.

PATROL KM

See PATROL Knowledge Module.

PATROL KM2MOF Tool

A compiler tool that converts `.km` and `.kml` files into `.mof` files that can be loaded into CIMOM. Applications that use the CIM standard can then interpret a PATROL KM.

PATROL Knowledge Module (KM)

A set of files from which a PATROL Agent receives information about resources running on a monitored computer. A KM file can contain the actual instructions for monitoring objects or simply a list of KMs to load. KMs are loaded by a PATROL Agent and a PATROL Console. For more information, see the PATROL for Windows User Guide.

PATROL Knowledge Module model

A set of classes and associations, derived from the CIM model, that represent the structure of a PATROL KM. *See also* model.

PATROL Method Provider	A provider, created by BMC Software, that handles calls to the WMI methods. It translates each call into a command that is executed by one of the PATROL Adapter for WBEM components or PATROL Agents. <i>See also</i> provider.
PATROL Namespace	A memory array that contains an internal representation of the PATROL object hierarchy. Values in the agent namespace are available to PSL scripts, eliminating the need to develop code to collect this data.
PATROL Provider	A provider, created by BMC, to allow CIMOM to access PATROL data on demand in the form of CIM objects. <i>See also</i> provider.
PEM	<i>See</i> PATROL Event Manager.
PEM COM server	A COM server which exposes interfaces used to access PATROL Event Manager functions.
property	A name/value pair that describes a unit of data for a class. Property values must have a valid Managed Object Format (MOF) data type.
provider	A COM server that communicates with managed objects to access data and event notifications from a variety of sources, such as the system registry or an SNMP device. Providers forward this information to the CIM Object Manager for integration and interpretation.
schema	<i>See</i> Model.
SDK	<i>See</i> Software Development Kit.
Software Development Kit (SDK)	A set of tools designed to help a developer create software. An SDK can include such things as libraries, header files, books, on-line help and sample programs.

static	A CIM class definition or instance that changes infrequently and is stored in the CIM repository until it is explicitly deleted. The CIMOM can provide this information without the help of a provider.
Unified Modeling Language (UML)	A notation language used to express a software system using boxes and lines to represent objects and relationships.
UML	<i>See</i> Unified Modeling Language.
WBEM	<i>See</i> Web-Based Enterprise Management.
Web-Based Enterprise Management (WBEM)	An initiative (supported by the DMTF) based on a set of management and Internet standard technologies developed to unify the management of enterprise computing environments. WBEM provides the ability for the industry to deliver a well-integrated set of standard-based management tools leveraging the emerging technologies such as CIM and XML.
Windows Management Instrumentation (WMI)	The implementation of WBEM for the Windows platform.
WMI	<i>See</i> Windows Management Instrumentation.
WMI Query Language (WQL)	A dialect of structured query language (SQL) with extensions to support WMI event notification and other WMI-specific features.
WQL	<i>See</i> WMI Query Language.

Index

A

Add Remote Agents menu command 5-10
 Address parameter 2-8
 Agent Model. *See* PATROL Agent Model 2-5
 AgentName 3-4
 AgentName parameter 2-10, 2-12, 2-14, 2-15
 application classes
 description 5-5
 WBEM 5-8
 WBEM_CIMOM 5-9
 WBEM_NAMESPACE 5-11
 WBEM_PROCESS 5-14
 WBEM_PROVIDER 5-12
 application icons 5-7
 associations
 PATROL Agent Model 2-7
 PATROL KM Model 2-20
 AuthenticateOnConnect
 PATROL_Agent 2-11

C

CIM namespace 2-3
 CIM Object Manager. *See* CIMOM

CIMOM (CIM Object Manager) 1-2
 connecting to B-2
 repository 5-9
 CIMOM Service menu command 5-11
 cimom.exe 5-14
 CIMOMProcess 5-7
 CIMOMRepository 5-7
 CIMV2 namespace 2-3
 classes
 PATROL Agent Model 2-7
 PATROL KM Model 2-19
 COM objects 3-2
 command line options, PATROL KM2MOF Tool 4-4
 compiler 1-3
 compiling a .km file 4-5
 compiling a.kml file 4-6
 computer classes 4-2
 Computer System Report menu command 5-8
 conventions, document xvi
 converting .km and .kml files 4-2
 core components 1-5

D

Delete Remote Agents menu command 5-10

Delimiters parameter 2-9, 2-13
diagram interpretation 2-2
Disconnect
 PATROL_Agent 2-11
Discover
 PATROL_Agent method 2-8
DMTF (Desktop Management Task Force)
 CIM xiii
document conventions xvi

E

EidRange parameter 2-13
EncryptPassword
 PATROL_Agent method 2-9
EndTimeFilter parameter 2-12
error
 PATROL Instance Provider B-2
 PATROL KM2MOF Tool 4-10
 query B-2
 uninstall B-2
error logging 3-3
error messages, KM 5-21
EvClass parameter 2-13
event log 3-3
Event Model. *See* PATROL Event Model
Event Provider 1-4, 3-5
evtprov.dll 1-6
example scenarios
 KM Model A-2
 PATROL Agent Model A-12
 PATROL Event Model A-6
 PATROL KM Development Model A-15
ExtrinsicEvent 2-21

F

firewalls C-2
Format parameter 2-15

G

Generate Configuration MOF menu
 command 5-10
GetEventHistory
 PATROL_Event method 2-12
GetParamAlarmRanges
 PATROL_Parameter method 2-14
GetParamAnnotation
 PATROL_Parameter method 2-14
GetParamHistory
 PATROL_Parameter method 2-15
GetVariable
 PATROL_Agent method 2-10

I

icons, description 5-7
Instance Provider 1-4, 3-4

K

KM 5-3
km and kml files 4-2
KM computer classes 4-2
km file 4-5
KM hierarchy 5-6
KM Model. *See* PATROL KM Model
KM properties mapped to MOF 4-7
KM2MOF Tool 4-2
 command line options 4-4
 description 1-5
 messages 4-10
 using 4-4
km2mof.exe 1-6
km2mof.res 1-6
kml file 4-6
Knowledge Module (KM)

- application classes 5-5
- error messages 5-21
- icons 5-7
- loading 5-4

L

- l command line option 4-5
- Load KM menu command 5-12
- Load MOF menu command 5-12
- loading
 - .mof files into CIMOM 4-7
 - PATROL Adapter for WBEM KM 5-4
- LocalPort D-3
- LogOutputLevel D-2

M

- mapping to MOF files 4-7
- MaxPingsPerSecond D-3
- MaxReplies parameter 2-12, 2-15
- menu commands
 - Add Remote Agents 5-10
 - CIMOM Service 5-11
 - Delete Remote Agents 5-10
 - Generate Configuration MOF 5-10
 - Load KM 5-12
 - Load MOF 5-12
 - Modify Local Agent 5-10
 - PATROL Provider Agents 5-12
 - PATROL Provider Configuration 5-10
 - PATROL Provider KMs 5-12
 - Provider Details 5-13
 - Remote Agents Defaults 5-10
 - Reports 5-12
 - Start 5-11
 - Start Monitoring 5-21
 - Stop 5-11
 - Stop Monitoring 5-21

- Unload KM 5-12
- WQL Query 5-12

- messages
 - PATROL KM2MOF Tool 4-10
- Method Provider 1-4, 3-6
- methods 2-8
- mfc42.dll 1-6
- Microsoft SDK MOF Compiler 1-3
- Microsoft WMI core components 1-5
- model diagram interpretation 2-2
- models 1-4
 - PATROL Agent Model 2-5
 - PATROL Event Model 2-21
 - PATROL KM Model 2-16
- Modify Local Agent menu command 5-10
- MOF compiler 1-3
- MOF file mapping 4-7
- MOF files 2-22, 4-7
- mofcomp.exe 1-3, 4-4, 4-7
- monitoring
 - CIMOM 5-3
 - CIMOM repository 5-9
 - PATROL Providers 5-3
- msgdll.dll 1-6
- msvbvm50.dll 1-6

N

- namespace 2-3, 2-4, 5-7
- NodeFilter parameter 2-13
- notation 2-2
- NSeverityFilter parameter 2-13
- NT Event Log 3-3

O

- o command line option 4-5
- Object Manager. *See* CIMOM
- OriginFilter parameter 2-13

output file name 4-5
OutputToApplicationLog D-2
OutputToDebugger D-2
OutputToNTEventLog D-2

P

PageFaultsPerSec parameter 5-14, 5-16

PageFileBytes parameter 5-14, 5-16

Parameter parameter 2-14, 2-15

parameters

PageFaultsPerSec 5-14, 5-16

PageFileBytes 5-14, 5-16

PrivTimePercent 5-14, 5-17

ProcessColl 5-14, 5-15

ProcessorTimePercent 5-15, 5-18

ThreadCount 5-15, 5-18

UserTimePercent 5-15, 5-19

WorkingSet 5-15, 5-20

Password parameter 2-9

patagent.mof 1-7, 2-22

patcimv2.mof 1-7, 2-22

patevent.mof 1-7, 2-22

patkm.mof 1-7, 2-22

patmof.dll 1-6

patremot.mof 1-7, 2-22

PATROL Adapter for WBEM KM

application classes 5-5

PATROL Adapter for WBEM Knowledge

Module 5-3

PATROL Agent Model

associations 2-7

classes 2-7

description 2-5

example A-12

PATROL Event Manager 1-3

PATROL Event Model

description 2-21

example A-6

PATROL Event Provider 3-5

PATROL Instance Provider 3-4

PATROL KM

computer classes 4-2

development example A-15

PATROL KM Model

associations 2-20

classes 2-19

description 2-16

example A-2

PATROL KM2MOF Tool 4-2, 4-4

PATROL Method Provider 3-6

PATROL Models 1-4

PATROL namespace 2-3

PATROL Provider Agents menu command
5-12

PATROL Provider Configuration menu
command 5-10

PATROL Provider KMs menu command
5-12

PATROL Providers 3-2

PATROL Single Sign-On plug-in C-3

PATROL WBEM KM 1-5

PATROL WBEM KM hierarchy 5-6

PATROL_Agent 2-7

AuthenticateOnConnect 2-11

Disconnect 2-11

UpdateProvider 2-10

PATROL_Agent method

Discover 2-8

EncryptPassword 2-9

GetVariable 2-10

PATROL_AgentManages 2-7

PATROL_AppHasInstances 2-7

PATROL_AppInstance 2-7

PATROL_AppInstContains 2-8

PATROL_AppInstHasParameters 2-7

PATROL_Dependency 1-11, 2-8

PATROL_Error 1-11

PATROL_Event 1-11, 2-21, 3-5

PATROL_Event method

- GetEventHistory 2-12
- PATROL_KM_ 2-19
 - PATROL_KM_AgentHasAppInstances 2-20
 - PATROL_KM_AppInst 2-19
 - PATROL_KM_MgmtApp 1-11, 2-19
 - PATROL_KM_MgmtAppHasParameter 2-20
 - PATROL_KM_Package 1-11, 2-19
 - PATROL_KM_PackageHasMgmtApp 2-20
 - PATROL_KM_Parameter 2-19
 - PATROL_ManagedObject 1-11, 2-7
 - PATROL_MgmtApp 2-7
 - PATROL_Parameter 2-7
 - PATROL_Parameter method
 - GetParamAlarmRanges 2-14
 - GetParamAnnotation 2-14
 - GetParamHistory 2-15
- patsetup.mof 1-7, 2-22
- PatternFilter parameter 2-13
- PEM COM Server 1-3
- PEMServerPollPeriod_Minutes D-2
- pemsvr.exe 1-6
- pkmparser.dll 1-6
- PkmParser1.dll 4-10
- Port parameter 2-10, 2-12, 2-14, 2-15
- PortRange parameter 2-8
- PrivTimePercent parameter 5-14, 5-17
- problems and solutions B-2
- ProcessColl parameter 5-14, 5-15
- ProcessorTimePercent parameter 5-15, 5-18
- Provider Details menu command 5-13
- provider name 5-7
- providers
 - definition of 3-2
 - event 3-5
 - instance 3-4
 - method 3-6
 - PATROL Providers 1-4
- PSL API 1-5
- pwa_uninst.dll 1-7

- pwbem.xpc 1-6
- pwbemlib.lib 1-8, 1-14
- pwinstpr.dll 1-6

Q

- query errors B-2

R

- readme.txt 1-7
- registry settings D-2
- release notes xvi
- Remote Agents Defaults menu command 5-10
- Reports menu command 5-12
- ResolveHostnames parameter 2-9
- Result parameter 2-9, 2-10, 2-13, 2-14, 2-15
- rundll32.exe 1-6

S

- ScanSubnet parameter 2-9
- schema files 2-22
- schemas 1-4
- Single Sign-On plug-in C-3
- Start CIMOM Process Monitoring
 - Automatically menu command 5-8
- Start CIMOM Service Automatically menu command 5-8
- Start menu command 5-11
- Start Monitoring menu command 5-21
- StartTimeFilter parameter 2-12
- StatusFilter parameter 2-12
- Stop menu command 5-11
- Stop Monitoring menu command 5-21
- suppresses dialog boxes 4-5

T

ThreadCount parameter 5-15, 5-18
Timeout parameter 2-8, 2-12
TimeStamp parameter 2-14
TypeFilter parameter 2-12

U

UML notation 2-2
Unload KM menu command 5-12
UpdateProvider
 PATROL_Agent 2-10
UserTimePercent parameter 5-15, 5-19
util.msg 1-8, 1-14
utlutil.lib 1-8, 1-14
utmsgsl.lib 1-8, 1-14
utstdmsgl.lib 1-8, 1-14
utstrl.lib 1-8, 1-14

V

Variable parameter 2-10

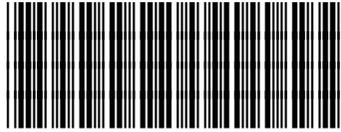
W

WBEM 5-7
WBEM (Web-Based Enterprise
 Management)
 definition 1-2
WBEM application class 5-8
 menu commands 5-8
 parameters 5-8
WBEM KM 1-5, 5-3
WBEM KM Configuration menu command
 5-8
WBEM KM hierarchy 5-6
WBEM KM icons 5-7

WBEM registry settings D-2
WBEM Services 5-21
wbem.km 1-7, 1-13
wbem.kml 1-7, 1-13
wbem.msg 1-8, 1-14
WBEM_CIMOM 5-7
WBEM_CIMOM application class
 description 5-9
 menu commands 5-9
 parameters 5-9
wbem_cimom.km 1-7, 1-13
WBEM_NAMESPACE 5-7
WBEM_NAMESPACE application class
 description 5-11
 menu commands 5-11
 parameters 5-11
wbem_namespace.km 1-7, 1-13
WBEM_PROCESS 5-7
WBEM_PROCESS application class
 description 5-14
 menu commands 5-21
 parameters 5-14
wbem_process.km 1-8, 1-13
WBEM_PROVIDER 5-7
WBEM_PROVIDER application class
 description 5-12
 menu commands 5-13
 parameters 5-13
wbem_provider.km 1-8, 1-13
wbemadapter.adm 1-7
wbemadapter_migrate.txt 1-6
wbemcmdagentadd.psl 1-8, 1-14
wbemcmdagentdefaults.psl 1-8, 1-14
wbemcmdagentdelete.psl 1-8, 1-14
wbemcmdagentdetails.psl 1-8, 1-14
wbemcmdagentmodify.psl 1-8, 1-14
wbemcmddcimomstart.psl 1-8, 1-14
wbemcmddcimomstop.psl 1-8, 1-14
wbemcmdconfig.psl 1-8, 1-14
WBEMCmdExport.psl 1-9, 1-14

wbemcmdkmdetails.psl 1-8, 1-14
wbemcmdloadkm.psl 1-8, 1-14
wbemcmdloadmof.psl 1-8, 1-14
wbemcmdprovdetails.psl 1-9, 1-14
wbemcmdquery.psl 1-9, 1-14
WBEMCmdSysReport.psl 1-9, 1-14
wbemcmdunloadkm.psl 1-9, 1-14
wbemdiscovery.psl 1-9, 1-14
wbemprmproccoll.psl 1-9, 1-14
wbemutil.lib 1-8, 1-14
Web-Based Enterprise Management. *See*
 WBEM
Windows NT Event Log 3-3
WMI core components 1-5
wmi_util.exe 1-6
wmicheck.exe 1-6
WorkingSet parameter 5-15, 5-20
WQL Query menu command 5-12

Notes



100031650