



System Administration using WMI

Robert Stuhr
May 3, 2001

Overview

- **System Administration Problem**
- **Consortium Solution, WBEM**
- **Microsoft WBEM Implementation, WMI**
- **Observations/Current Activities**
- **Cross Platform WBEM Architectures**
- **Conclusions**

**System
Administration
Problem**

System Administration Requirements

- **Typical functions**
 - application configuration (install, update, delete)
 - user admin (install, privileges, profiles, delete)
 - run time services (backups, status events, process queue manipulation)
- **Remote, as well as local, computer management**
- **Staffing considerations**
 - lower level employees
 - software tools as aid

System Administration Challenges

- **N x M Enterprise Problem**
 - N different types of Operating Systems (OS)
 - M machines of each OS, configured differently

= Logistical Headache

- **Everyone trying to address problem**
 - Commercial (Tivoli, Unicenter)
 - Government (COE)
- **Key Issue: make all computing platforms look the same from the outside**

**Consortium Solution, Web
Based Enterprise Management
(WBEM)**

WBEM Basics

- **Developed by Distributed Management Task Force (DMTF)**
- **DMTF is collaborative group of technology vendors (Microsoft, Sun, Cisco, et.al.)**
- **Consortium used Universal Modeling Language to describe generic, manageable computer components**
- **Model schema examples**
 - machine hardware characteristics
 - installed software inventories
 - current file structures
- **Common Information Model (CIM) result**

CIM Characteristics

- **Object oriented database, describing computer**
 - objects represent manageable components
 - methods to access property attributes, discerning behavior of physical object instances
- **Object repository contains**
 - static object instances
 - definitions for dynamic objects
 - object described by Managed Object Format (MOF)
- **Repository controlled by CIM Object Manager (CIMOM)**
- **DMTF envisioned vendors would subclass CIM objects, adding information relevant to particular hardware/software configurations**

WBEM Drawbacks

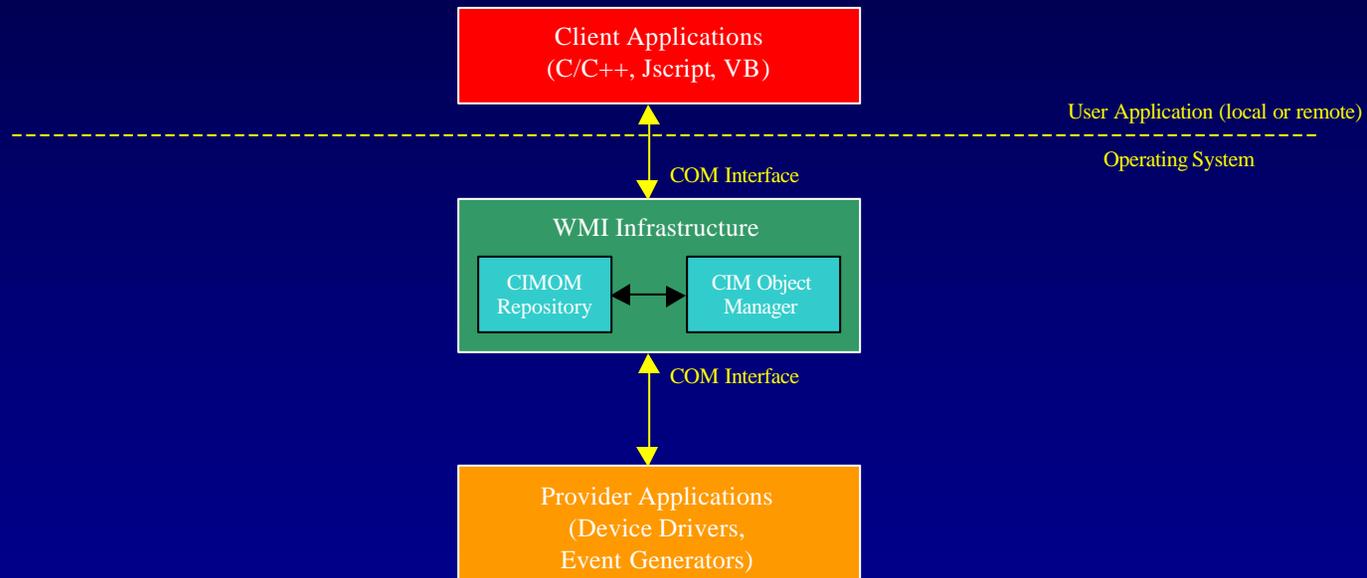
- Only CIM object framework standardized
- No standard user API defined
- Interfaces to vendor implementations differ
 - Sun: Java RMI
 - Microsoft: COM
- Other interfaces mechanisms need adapters

**Microsoft WBEM Solution,
Windows Management
Instrumentation
(WMI)**

WMI Characteristics

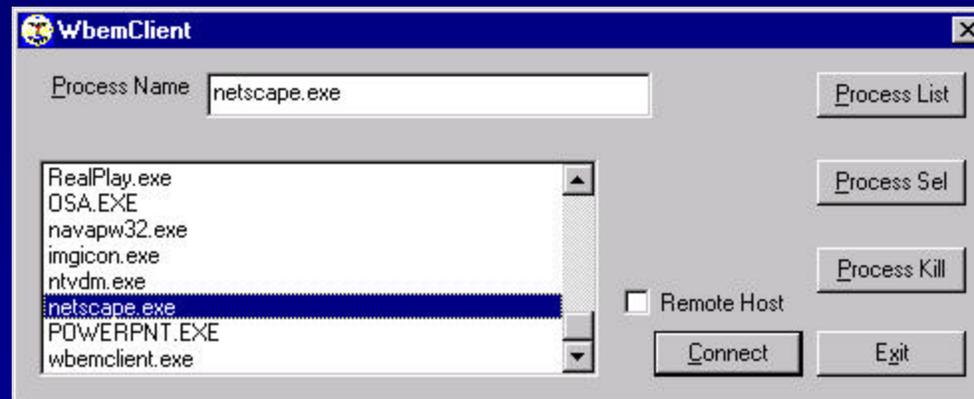
- **Availability**
 - Win2000: built in
 - WinNT: installable service
- **Users/Providers bi-directional communication via CIM**
 - consumers (users) query WMI for information
 - producers (providers) notify users, registered to event
- **Windows includes generic providers (CIMWin32, msiProv, RegProv)**
- **COM interface (other standards through adaptors)**
- **Microsoft Software Development Kit (SDK)**
 - C header files, compiled languages
 - SWbem* object tree, scripting languages

WMI Block Diagram



WMI Demonstration

- Process queue manipulation
 - connect to WMI
 - list running processes
 - kill selected process



Observations/ Current Activities

WMI Benefits for COE

- **Windows sys adm functions become part of OS**
 - eliminates sys adm segments on each machine
 - WinTel device driver vendors “encouraged” to expose objects to WMI
 - COE approved OS
- **Government gets for free what they are currently paying for**
- **Same benefits for WBEM implementation on UNIX COE platforms**

Personal Experience WMI, Good and Bad

- **Good** user client easy to generate (demo)
 - security included (password challenge)
 - “discovery” worked (queried on WBEM superclass)
 - extendable to remote machine (DCOM)
- **Bad** missing sys adm functions
 - installed apps not directly exposed (have to navigate through Registry with RegProv)
 - cannot change user password (have to use ADSI)

**Opportunity: infancy of WMI enables
Government to influence final product**

Current COE Application Profile Manager (APM) Segment

- Platform independent (Windows/UNIX)
 - GUI Java based
 - Machine Interrogator (server) Perl based
- Perl code internals are platform specific (separate code blocks per platform)
- Initially written for UNIX, then Windows version force fitted

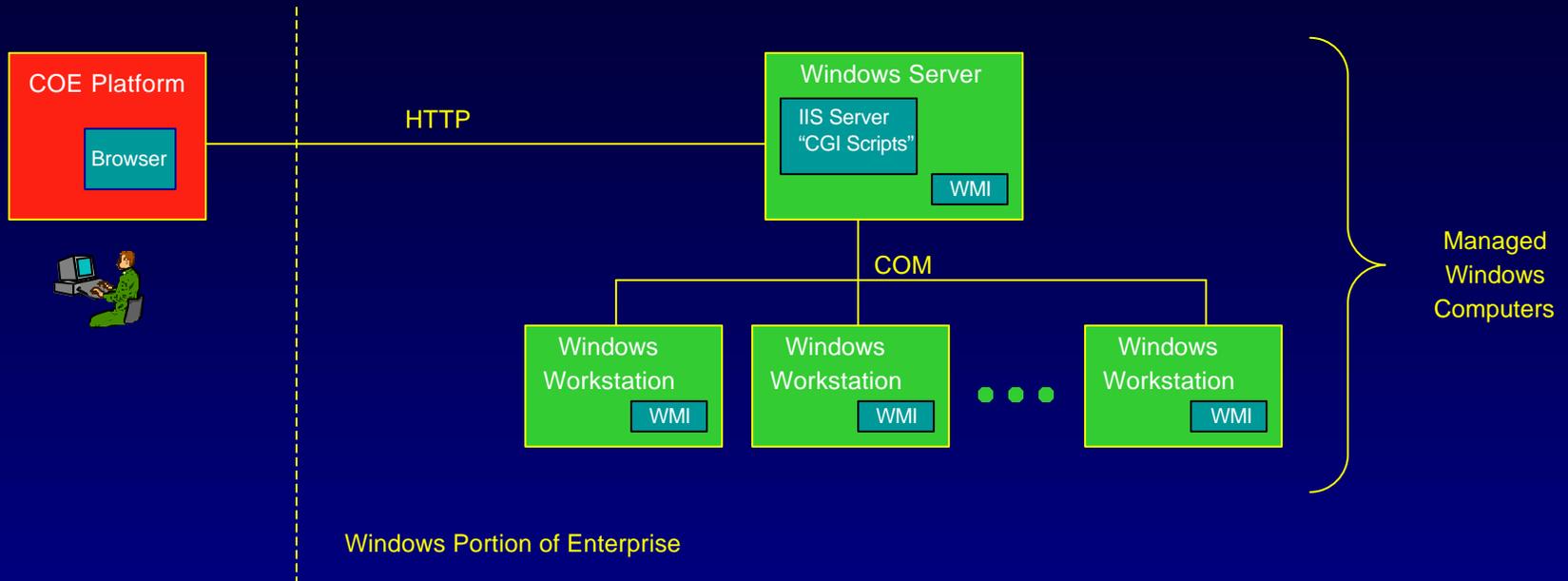
Windows segments not required to interface with APM

Pursuing WMI Prototype

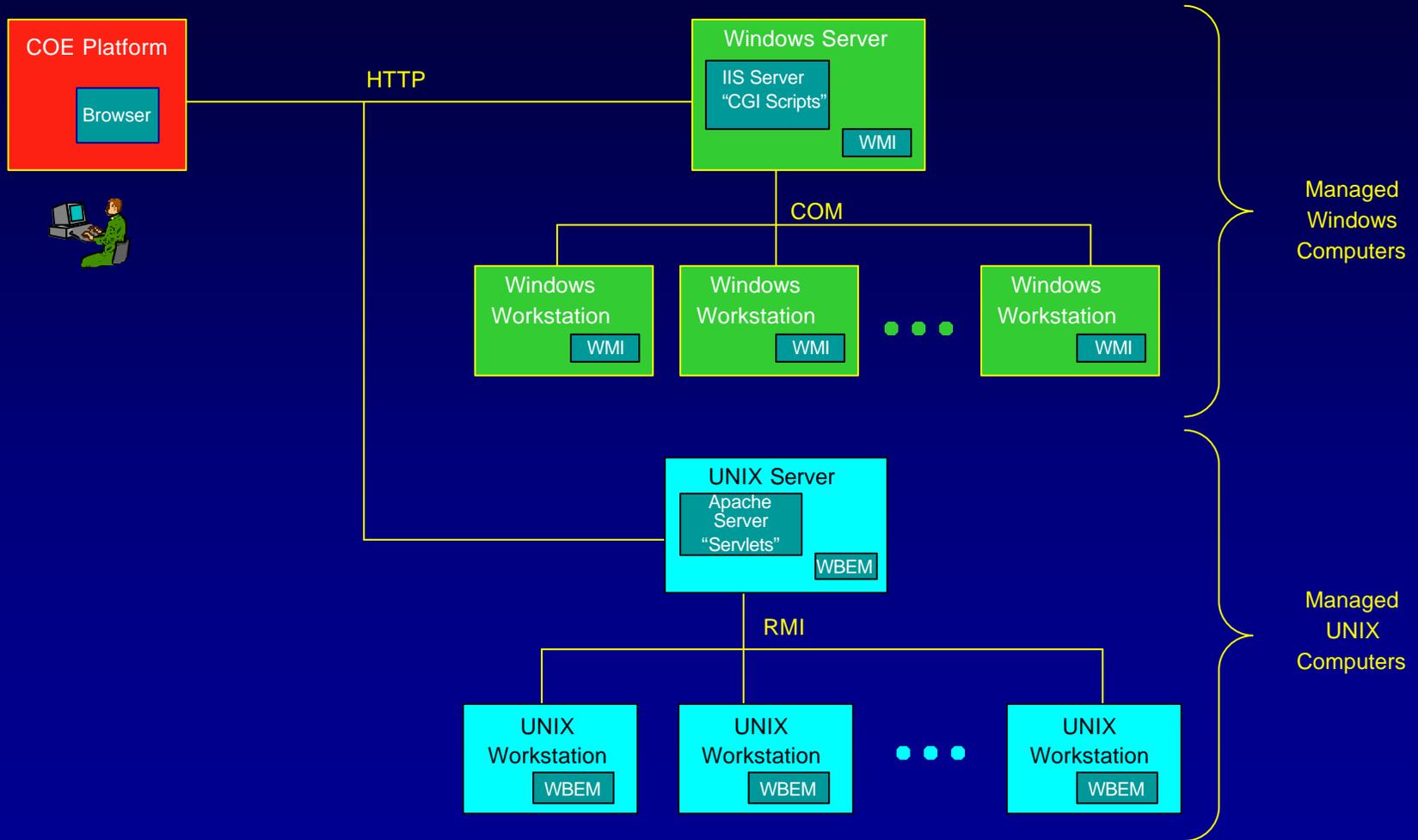
- **Work with Microsoft in implementing some APM functions**
 - list installed applications
 - list users
 - assign applications to users (profiling)
- **Consider using platform independent GUI**
 - browser (Web architecture)
 - Java (Client / Server architecture)
- **Solicit user input**
 - Underway, Microsoft POC named (Lorenzo Rizzo)**

Cross Platform WBEM Architectures

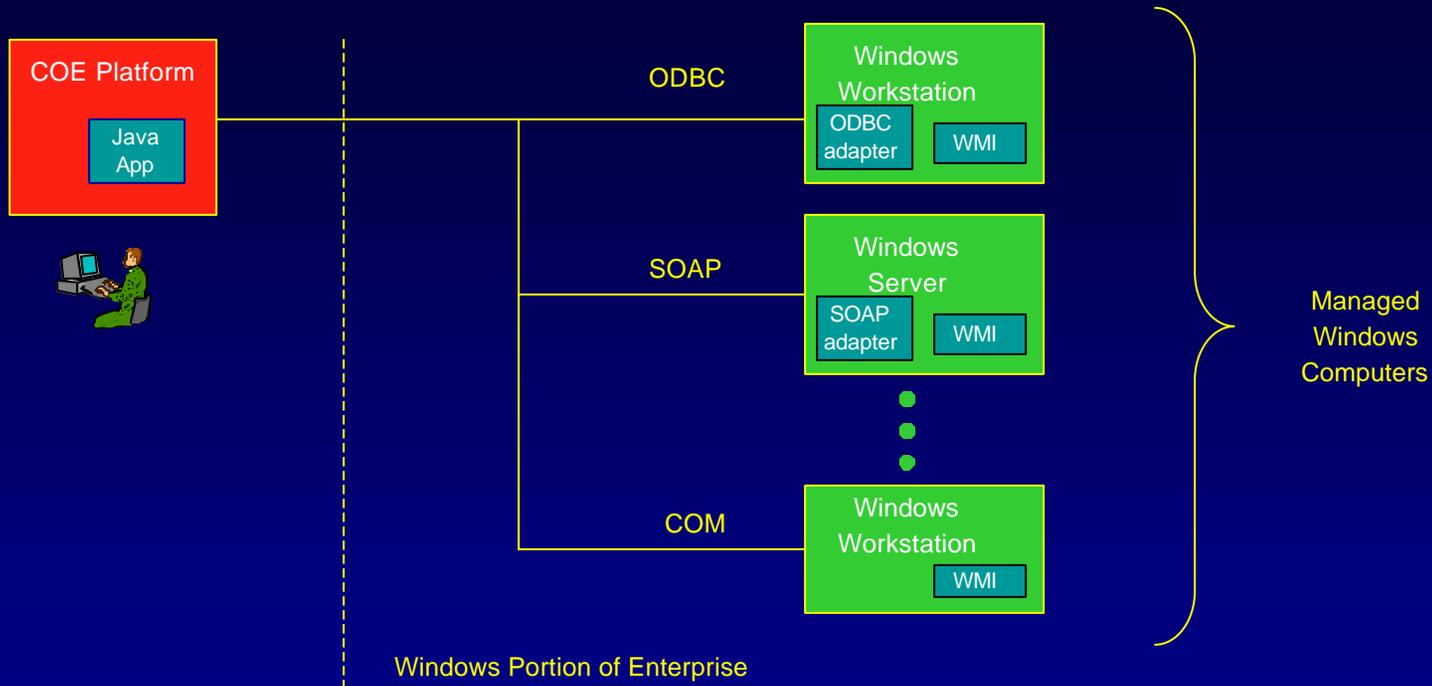
WMI Architecture: Web with Browser GUI



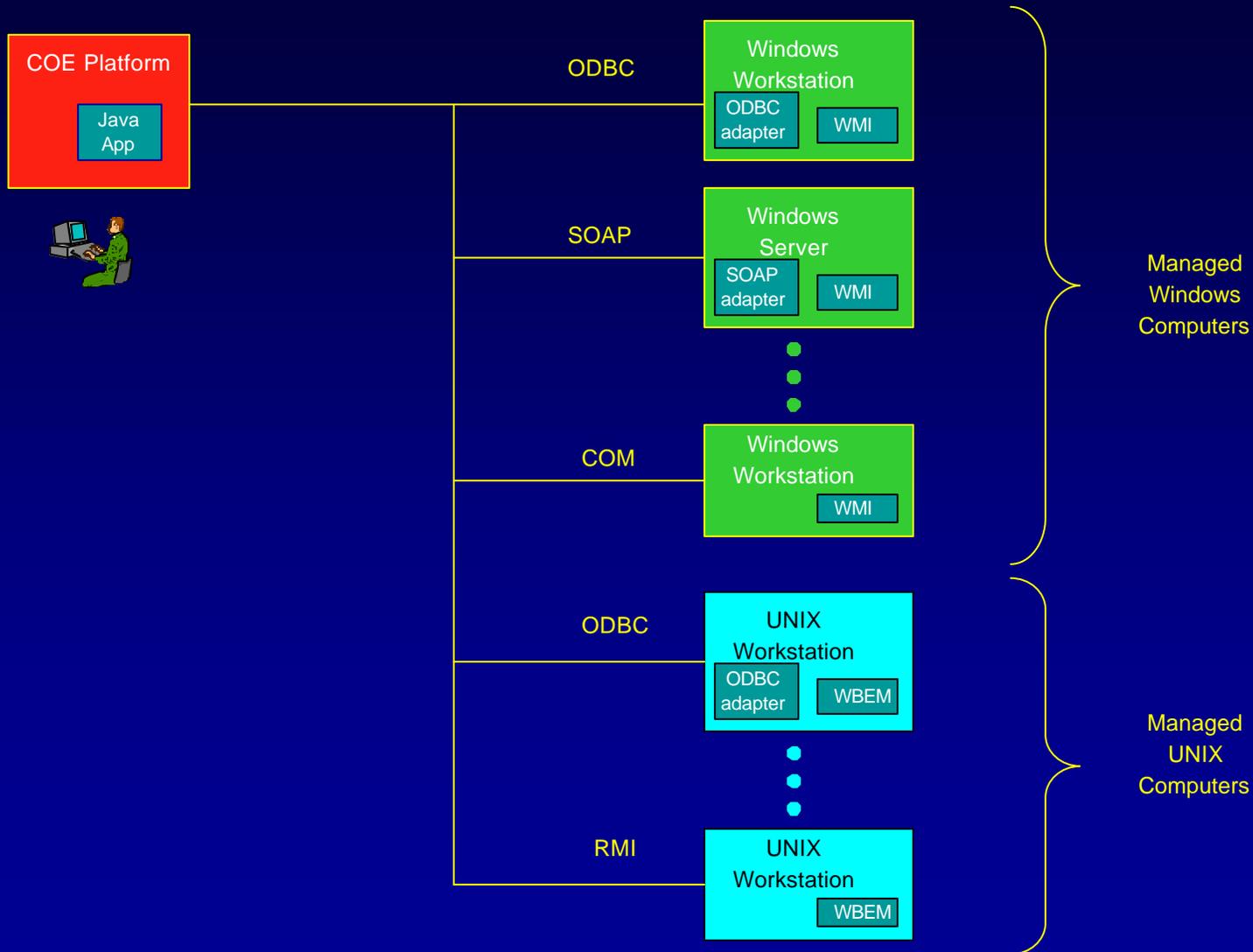
Heterogeneous Architecture: Web with Browser GUI



WMI Architecture: Client / Server with Java GUI



Heterogeneous Architecture: Client / Server with Java GUI



Conclusions

WMI System Management Summary and Next Steps

- **Native Mechanism for Windows Platform**
 - researching WMI capabilities
 - prototyping some functionality relevant to COE
 - interacting with Microsoft
- **Needed from Other NTAG Members**
 - feedback on our WMI approach
 - shouldering some of the load
- **Needed from Broader COE Community**
 - evaluate WBEM implementations on UNIX systems
 - contribute to WBEM cross platform architecture

Backups

I&RTS Recommended Changes

7.9.3.1 Client WMI Requirements

Client management applications **shall** use the Component Object Model (COM) to interface with WMI. COM is the preferred connection mechanism, as it provides direct access both remotely and locally. Vendors using scripting languages may take advantage of the special “helper” objects that WMI provides (SWbem* object tree), which isolate developers from COM interface details.

Other interfaces, such as XML and ODBC, may provide WMI connectivity in the future, when appropriate (COE approved) adapters become available

I&RTS Recommended Changes

7.9.3.2 WMI Provider Requirements

Device drivers **shall** ensure the device's resource objects are exposed to the WMI infrastructure. The intent is to maximize the ability for client management segments to monitor and/or manage devices through the standard WMI interface. Developers providing WMI event notification management services **shall** utilize the object structures already existing within WMI. Exposing mission-application segment objects to the WMI is determined by the SSA.

Segments with WMI providers **shall** use the existing CIM (with WMI subclasses) management objects, however they may create additional subclasses if required. Developers should use the Managed Object Format (MOF) language, defined by DMTF, to describe the new objects.

WMI Class Repository

- Root\CIMV2 (namespace for CIM version 2)
 - 600 classes, 3000 properties
 - Win32_* classes inherent from CIM_*
- WMI class examples, out of the box
 - Win32_SerialPort : CIM_SerialController
 - Win32_SerialPort.DeviceID="COM1"
 - Win32_UserAccount : Win32_Account : CIM_LogicalElement
(abstract class)
 - Win32_UserAccount.Name="stuhr",Domain="D560"
 - many association classes (per WBEM spec)
 - Win32_SerialPortSetting (ties Win32_SerialPort to its settings)
 - Win32_GroupUser (ties Win32_UserAccount to its group)
 - Win32_SubDirectory (connects subdirectory instances)

Win32_Directory Recursion Example (using WMI Object Browser Tool)

