Supporting Web-Based Enterprise Management:

WMI implementation case

A. Retik
WMI Group
Microsoft Corporation
One Microsoft Way
Redmond, WA
e-mail:arkadyr@microsoft.com

D. Fischer
WMI Group
Microsoft Corporation
One Microsoft Way
Redmond, WA
e-mail:dfischer@microsoft.com

S. Muzio
WMI Group
Microsoft Corporation
One Microsoft Way
Redmond, WA
e-mail: simonmu@microsoft.com
Abstract

Globalization of business activities and creation of virtual organizations have produced a need for efficient tools to support management tasks in a new environment. Web-based Enterprise Management (WBEM) was introduced by The WBEM 5 (Microsoft, Intel, BMC Software, Compaq, Cisco Systems) in June 1996 in order to facilitate the use of Internet technologies for management of distributed systems. Distributed Management Task Force (DMTF), the organization made up of the leading companies in today’s computing industry, has adopted the initiative as a set of management and Internet standard technologies aimed to unify the management of enterprise computing environments.

This paper starts from an overview on Web-Based Enterprise Management (WBEM), followed by a brief description of the standard WBEM components. It subsequently focuses on the Microsoft’s implementation of WBEM technologies—Windows Management Instrumentation (WMI), which enables enterprise IT managers to monitor, access and control all aspect of a distributed Internet system. This paper then outlines how the WMI implementation can be used to enable web-based enterprise management. Specific scenarios and solutions are presented. Finally, the conclusions are made.

Keywords: Web-Based Enterprise Management, Distributed Systems Management, Common Information Model, Windows Management Instrumentation (WMI).
Supporting Web-Based Enterprise Management:
WMI implementation case

1. INTRODUCTION

Recent advances in information and communications technologies have resulted in rapid modernization of both public and private computing infrastructure. As enterprise networks have become the backbone of many organizations, large investments are being made in order to adapt to the changing environment. However, the real cost of setting and maintaining a distributed computer network (known as a total cost of ownership) extends far beyond the initial purchase of hardware and software. It includes the deployment and configuration expense, costs associated with deploying hardware and software updates, training and retraining, day-to-day maintenance and administration, and telephone and on-site technical support. With these escalating costs in mind, many companies are working together on several initiatives designed to lower the total cost of ownership of personal computers in the enterprise.

Key among these efforts is Web-Based Enterprise Management (WBEM), an industry initiative that establishes management infrastructure standards and provides a way to combine information from various hardware and software management systems [1]. WBEM specifies standards for a unifying architecture that allows access to data from a variety of underlying technologies and platforms, and presents that data in a consistent
fashion. Management applications can then use this information to create solutions that reduce the maintenance and life cycle costs of managing an enterprise network. The WBEM standard includes the Common Information Model (CIM) schema, which is an industry standard driven by the Distributed Management Task Force (DMTF) [2].

Microsoft Windows Management Instrumentation (or WMI) is an implementation of the WBEM standard. It provides a consistent and richly descriptive model of the configuration, status, and operational aspects of the Windows operating systems. Used in conjunction with other management services provided in the system, WMI can simplify the task of developing well-integrated management applications, allowing vendors to provide customers with scaleable, effective enterprise management solutions over the Internet.

This paper proceeds as follows. It starts from an overview of WBEM and Common Information Model. It then briefly describes the WBEM standard components and the WMI architecture elaborating on how it functions with other management components. The paper subsequently focuses on the XML/HTTP extension elaborating on how it supports Web-Based Management of an enterprise and analyzing the current implementation. Finally, the conclusions are drawn.

2. WEB-BASED ENTERPRISE MANAGEMENT

This Section introduces and describes a Web-Based Enterprise Management Initiative and Common Information Model, which are the basis for WMI implementation.
2.1 Background

Increasing adaptation of Web-based management for enterprise networks has attracted a lot of attention of both academic and commercial research and development. Some research works concentrate on a new distributed paradigm, analyzing and classifying new ways of systems management [3], or formulating methodologies for the re-engineering of existing management solutions [4]. Others aim to improve performance and increase efficiency of the networked systems by proposing new management schemes and protocols [5, 6] while another group proposes a different architecture for scaleable management services in a distributed environment [7], open management [8] and active management [9] frameworks.

There are also industrial research and development projects and initiatives addressing this issue. One of the most notable among them is the Web-Based Enterprise Management (WBEM), an industry initiative to develop a standardized, non-proprietary means for accessing and sharing management information in an enterprise network across the Internet. WBEM will result in technology that enables customers to collect, associate, and aggregate management data from diverse sources, thus creating richer and more accurate views of their enterprise environments. The WBEM initiative is intended to solve the problem of collecting end-to-end management and diagnostic data in enterprise networks that may include hardware from multiple vendors, numerous protocols and operating systems, and a legion of distributed applications, as illustrated in Figure 1.
Typically, enterprise management has been tied to different protocols and interfaces for different disciplines—for example, Simple Network Management Protocol (SNMP) has been used for network management, and the Desktop Management Interface (DMI) has been used for desktop systems management. WBEM assumes that enterprise network
management requires tools that work together to provide a single, shared model for the collection of management information. WBEM provides this common model and data source, and can be extended to work with existing network components, tools, and protocols (see Figure 2).

Figure 2. WBEM standards-based initiative for enterprise network management
WBEM proposes a set of standards creating infrastructure for managing the distributed enterprise network. These management standards are:

- **CIM** - Common Information Model – defines the structure and conventions necessary to access information about the managed objects.
- **XML DTD** – supports centralization of information so that different clients and management tools can provide, retrieve, and analyze data.
- **HTTP/XML protocol mapping** – supports authorized access to managed objects from anywhere in the network.

The implementation of these standards within WMI will be described in sections 3 and 4.

**2.2 WBEM Background**

The WBEM proposal was originally envisioned in 1996 by a collection of companies headed by Microsoft, Compaq Computer, BMC Software, Cisco Systems, and Intel. The vision was to define an open environment for management, where all managing systems and application could access, control, and share management information with each other and with any managing agent on a managed device, using existing technology and standards as much as possible. In many respects, the goal reflected the technological breakthroughs of the World Wide Web, where, for the first time, devices on the Internet could act as sources and consumers of information without any knowledge of the specific environments in which each component operated. Because of this shared vision, together with the possibility of using Web-based technologies in addition to more conventional management tools to create an open management environment, the name for the initiative became Web-Based Enterprise Management (WBEM).
The founding companies, working together with the Distributed Management Task Force (or DMTF) [10], developed the prototype set of environment-independent specifications for how to describe and access any type of management instrumentation, including standards such as SNMP and DMI. The core component of this specification is a data description mechanism that is the DMTF standard known as the Common Information Model (CIM).

The CIM Specification describes the modeling language, naming, and mapping techniques used to collect and transfer information from data providers and other management models. The CIM Schema provides the actual model descriptions and information framework. It defines a set of classes with properties and associations, making it possible to organize information about the managed environment [2].

In June 1998, the Distributed Management Task Force (DMTF) announced that it was accepting a transfer of the WBEM initiative from the founding corporations. The DMTF is now the focal point for WBEM initiative efforts, providing an organizational framework for broader industry participation in the development of WBEM-compatible technologies and standards. Specific implementations of WBEM-based standards, such as the Microsoft Windows Management Instrumentation SDK, remain the responsibility of the vendors who developed them. In taking on the WBEM initiative, the DMTF agreed that it would use current WBEM technologies, such as the Microsoft implementation of CIM, as reference examples.
2.3 Common Information Model

Fundamentally, WBEM is an initiative that proposes the implementation of the Common Information Model. CIM is a standard under the WBEM umbrella rather than the other way around. CIM can be defined in our context as an object-oriented schema of managed objects. It offers a single data description mechanism for any management data. WBEM provides an information standard that defines how data is represented and a process standard that defines how components interact.

3. WMI: THE MICROSOFT WBEM IMPLEMENTATION

Since 1995, the Windows Management Instrumentation (WMI) group at Microsoft has been working on a Windows based implementation of WBEM technology. This work included the development of a WBEM Software Development Kit (SDK) and various CIM component and CIM compliant data provider technologies.

As the core of the Microsoft management infrastructure, WMI helps to reduce the maintenance and cost of managing components in a Windows 2000-based enterprise network. WMI provides:

• A rich and consistent model of Windows 98 and Windows 2000 operation, configuration, and status
• A COM API (Component Object Model Application Program Interface) that
supplies a single point of access to all management information.

- Interoperability with other Windows 2000 management services, which will facilitate vendors’ efforts to create well-integrated management applications.
- A flexible architecture that allows vendors to extend the information model to cover new devices, applications, and other enhancements by supplying schema extensions and writing code modules (WMI providers) that populate them.
- An event handling architecture that allows changes in management information to be identified, aggregated, compared to, and associated with other management information, and forwarded to local or remote management applications.
- Location transparent data access allowing applications to be developed that manage multiple systems or systems that consist of geographically dispersed components.
- A rich query language that enables detailed queries of the information model.
- A scriptable API, which enables management application developers to use simple software tools.

For example, local and remote eventing combined with a rich query language to the information model provides the means to create solutions to complex management problems. The ability to easily script these solutions in Visual Basic or using WSH facilitates the Windows NT-based management.

The following subsections describe the WMI, Microsoft’s WBEM implementation, in more detail.
3.1 WMI Architecture

Based on the CIM structure coupled with the XML DTD representation support and HTTP/XML protocol and mapping, WMI provides a unique infrastructure capable of supporting a variety of management solutions in a distributed Internet environment.

Figure 3. WMI infrastructure
A general picture of WMI infrastructure is presented in figure 3. A system service that provides all of the WMI functionality is *WinMgmt.exe*. This executable supports the CIM object repository, the CIM Object Manager, and the APIs that together deliver WMI. The other main components of WMI are described below.

3.1.1 *CIM Object Manager*

The CIM Object Manager (CIMOM) is a key component of the Microsoft implementation of WBEM technology. The CIMOM provides a collection and manipulation point for managed objects stored in the CIM repository—it facilitates gathering and manipulating information about these managed objects.

The CIM Object Manager does not access management information directly. WMI providers gather information from a resource (a managed object), and then make it available to management applications through the WMI API. The CIMOM, therefore, provides the CIM functionality in WMI, facilitating manipulation of distributed data. The essential function of the CIMOM is to provide an isolation layer between the client and the data providers, hiding from the client the details of how the information is produced. These details may be significant and come in the form of relationships between providers and differences in provider capabilities.

3.1.2 *WMI Providers*
WMI providers act as intermediaries between the CIM Object Manager and one or more managed objects. The essential function of the providers is to act as an isolation layer between the CIMOM and the environment being managed. This allows the CIMOM to be constructed in such a way that it is indifferent to the specifics of the interfaces of particular devices applications or management protocols. When the CIM Object Manager receives a request from a management application for information that is not available from the CIM repository or for notification of events that it doesn’t support, it identifies the appropriate provider using information contained in the schema and forwards the request to the provider. The provider then supplies the information or event notification requested.

WMI includes the following providers:

- Win32 Provider
- WDM Provider
- Event Log Provider
- Registry Provider
- Performance Counter Provider
- Active Directory Provider
- Windows Installer Provider
- SNMP Provider
- View Provider

Third-party vendors can use the WMI SDK to create custom providers to interact with managed objects that are specific to their own environments.
Note that the Microsoft WMI technologies do not attempt to replace existing management standards such as SNMP, DMI, or CMIP, or to preclude proprietary or platform-specific frameworks such as NDS. In fact, WMI complements these technologies by providing an integration point through which data from all such sources can be accessed. This integration point makes any management application independent of specific APIs or standards used to instrument managed entities, allowing system administrators to correlate data and events from multiple sources on either a local or enterprise basis.

3.1.3 WMI Query Language

The WMI Query Language (WQL) is a dialect of structured query language (SQL). It not only facilitates retrieval of data objects but also has extensions to support event notification and other WBEM-compatible features. When consumers register to receive event notifications, they specify a query that defines the type of event and the conditions under which it is delivered to them. It is possible to use WQL to construct specific event notification filters for components in an enterprise distributed network (see an example in Appendix I demonstrating how filtering can reduce the amount of data traffic).

3.1.4 Event Handling

Event publication, subscription and notification are key features of WMI, allowing components to detect hardware or software events and/or errors. An event can then be passed through the WMI architecture to the appropriate management component for
corrective action.

In WMI, an event is published as an instance of an event class. When an event occurs in the managed environment, an instance of the event class is created, the set of outstanding subscriptions is examined to determine if any client is interested in the event, and the event instance is then either delivered or discarded as appropriate. Event instances are never persisted and can only be accessed through the subscription mechanism. Events may be detected in one of two ways: 1) as a result of polling by the CIM Object manager or 2) as a result of detection by a provider (using any means available to the provider).

Events come in one of two types:

- intrinsic events are events that represent changes in the state of a class or instance defined in the schema;
- extrinsic events are events that do not correspond directly to some state change in a defined instance or class.

The object manager determines at subscription time which type of event is involved and handles the event detection and delivery appropriately. In the case of an event supplied by a provider, after an event occurs, the event provider notifies the CIM Object Manager, and then the CIM Object Manager delivers this notification to one or more registered recipients, known as event consumers. Event consumers can register with the CIM Object Manager to receive particular types of notifications, and event providers can register to supply particular types of notifications. To enable event consumers to operate independently from event providers, the CIM Object Manager acts as the intermediary, matching registered consumers with responsible providers and forwarding appropriate
events. Where the event is supplied through polling, the Object Manager simulates the event by periodically polling the data provider to determine if any relevant change has occurred. Apart from supplying the polling interval as a part of the subscription, the client is unaware of any difference between polled events and those supplied by a provider.

Event consumers register to receive notifications without knowing how the events and notifications are provided. To register, these consumers specify a filter. The filter is created using the WMI Query Language (WQL). It describes the conditions under which the consumer wants to receive event notification.

3.1.5 WBEM-compatible Scripting

One can use the scripting interfaces for WMI to develop script and Visual Basic applications that can interact with the CIM Object Manager. WMI provides scripting support for the following languages:

- Microsoft Visual Basic
- Visual Basic for Applications
- Visual Basic, Scripting Edition (VBScript)
- Microsoft JScript®
- Perl

Scripting languages and the ability to write scripts for batch processes, automating event handling, and so forth, have been around for many years. However, the Microsoft WBEM-compatible scripting provides the following scripting advantages:
• It uses a data-driven approach—CIM. CIM provides one model for manipulating disparate information, and the Object manager isolates applications from the complexity of various data sources. In writing scripts against WMI, the script writer is required to understand two things, the Object Manager interface and the schema. This contrasts with traditional scripting environments in which an arbitrarily large number of APIs and data structures have to be mastered with no guarantee of consistency of interpretation or presentation between one API and the next.

• It provides expansive coverage of system, network, and application information. The Microsoft implementation provides Win32, SNMP, registry, Windows Driver Model (WDM), Performance Monitor, Event Log, and ADSI providers. Other vendors, including Intel, Compaq Computer, Hewlett-Packard, and BMC Software, will be distributing providers to enable vendor-specific instrumentation, as will Microsoft Systems Management Server. Other providers from Microsoft are in development.

• Provider instrumentation is simple to extend. Tools, samples, and the extensible provider architecture are defined fully in the Microsoft WMI SDK. Moreover, there is wide industry support for provider development.

• New scripts are simple to write. The Microsoft WBEM-compatible API is simple to use, and the schema can be browse and is extensible to allow script coverage and innovation.

In the Windows 2000 timeframe, Microsoft intends to provide a comprehensive set of systems administration scripts. These scripts will provide local and remote system
administration capabilities from the command line, and will provide support for the
Windows 95, Windows 98, Windows NT 4.0, and Windows 2000 family of operating
systems. Script versions will be provided in VBScript, Perl, and JScript, and these scripts
will be easy to extend and customize for specific networks.

3.2 WMI XML/HTTP features

The following main features have been added to WMI to meet the industry standards for
information exchange (an example is also provided in Appendix I):

3.2.1 WMI XML/HTTP Server

This feature enables management applications to use HTTP protocols including the
ability to traverse firewalls. WMI XML/HTTP Server is a component that handles WMI
messages carrying XML representations of WMI objects and transported via HTTP.
XML/HTTP Server accepts in-bound messages and converts from XML to native WMI
representation before handing to WMI core server-side processes and also receives WMI
representation of WMI objects from WMI core and converts to XML before generating
out-bound messages that are transported over HTTP. XML conversions are transparent
to the WMI core. Therefore, any DMTF-conformant client can be used to access WMI
using this mechanism.

3.2.2 WMI XML/HTTP Client
WMI XML/HTTP client enables existing management applications to plug in to the WMI client infrastructure that supports HTTP protocols including the ability to traverse firewalls. The WMI XML/HTTP client exposes the current COM interface that handles CIM-encoded WMI objects allowing transparent access via either DCOM or XML/HTTP using the same WMI client API set (from C/C++, VB or script). The client connects to the server via HTTP and ships/receives payloads encoded in XML. The WMI client encapsulates out-bound CIM-to-XML and in-bound XML-to-CIM conversions so they are transparent to the management application. Another important point is that WMI APIs can then be used to access any remote DMTF-compliant server implementation (not just Microsoft, but certainly including it).

3.2.3 WMI XML Compiler

The XML Compiler parses a file containing XML statements and adds the classes and class instances defined in the file to the Common Information Model (CIM) repository.

3.2.4 WMI XML Representation of WMI Objects

The WMI client APIs expose the XML representation of any CIM object, not just those retrieved via XML/HTTP.

3.3 XML over HTTP

The WMI XML/HTTP transport provides a seamless way to communicate directly with WMI schema; it is intended to allow WMI users access to XML representations of WMI
objects via the HTTP protocol. This will allow Web-based applications to be written on top of WMI, thus giving a real reason for calling it “Web Based Enterprise Management (WBEM)”. Also the XML/HTTP Transport allows a user to access XML representations of WMI data locally, or any against a remote server accessed via DCOM.

The server component is already under development in accordance with the DMTF “Specification for CIM Operations Over HTTP version 1.0” encoding [11]. Clients will be able to use the server component through the use of XML/HTTP. Much of the fundamental design work has been provided via, and based on, the DMTF standards such as CIM XML DTD [12], Specification for the Representation of CIM in XML, Version 2.0 [13], and Specifications for CIM Operations Over HTTP Version 1.0 [11]. The XML/HTTP functional diagram is presented in Figure 4.
3.4 Security impact

The XML HTTP Transport uses Impersonation and/or a stronger form of security called Delegation for all accesses. Using Delegation you can effectively delegate to another process the right to pass onto a third process the right for that process to impersonate you, thus allowing chains of impersonating processes to be established. There are two cases as far as the system clients are concerned (see also Figure 5):
Figure 5. Security Implementation
Internal Clients (or clients that understand NT Challenge Response Authentication): In this case, NTLM Authentication is enabled on IIS (Internet Information Server). Nothing more needs to be done for Windows platforms cloaking allow the ISAPI Extension to use the identity of the remote web client in making calls to WMI.

External Clients (these are clients that do not understand NTLM authentication): The following choices will be made available: (a) Plug-ins could be used (for example, an NTLM plug-in for Netscape); or (b) for Windows 2000 environment: IIS 5.0 supports Digest Authentication – a HTTP standard. IE 5.0 supports this. (Though, current versions of Netscape do not support this, it is highly likely that it will be supported in the coming versions of Netscape). Digest authentication also requires that the domain of authentication be a Windows 2000 domain with certain special attributes enabled on any user that is authenticated in this way.

4. INTERNET ENABLED MANAGEMENT SERVICES

In order to enable management of IT infrastructure on an enterprise-wide basis a system should be able to (a) operate over Internet/Intranet protocols, (b) manage heterogeneous data centers via access to other CIM implementations over an industry standard protocol, and (c) encode and exchange information using an industry standard scheme. WMI uses HTTP to satisfy requirements (a) and (b). In order to meet requirement (c), WMI has
extended to support XML encoding of requests and responses. The following three cases/scenarios describe how the implementation of these functions provides for:

1) Enabling the management of storage devices attached to a PC over the Internet using WMI.

2) Managing an enterprise through a firewall connection

3) Supporting distributed data centers.

4.1 Storage Device management

SNIA (Storage Network Industry Association) members successfully demonstrated the first instance of Internet Enterprise Storage Resource Management (ESRM) at the Storage Networking World event (October 19-21, 1999 in Seattle [14]. This prototype was an interactive browser-based ESRM application, based upon the DMTF’s WBEM standard, that performed some basic capacity and asset management functions. Using the WMI implementations, the demonstration accessed information, in the form of CIM Schema objects, from disk storage resource facilities located around the US and in Europe. The ESRM server resided at a site in San Jose, California.
4.2 Through a firewall

WMI web-based enterprise management also supports management through a firewall (see section 3 for more detail). Using HTTP as the transport, WMI operations encoded in XML are forwarded through the firewall to WMI implementations on web-exposed servers. Note that in most significant sites on the web today, these web-exposed servers are dual-homed with connectivity to both a web-exposed network and to a private internal network. Typically the private internal network enables the web servers to access data stores and, via further firewalls, internal corporate networks (see a DNA example in [15]). Via WMI internal proxy mechanisms, the XML-encoded operations can be forwarded to hosts operating on the private internal network that are not directly exposed to the Internet. This capability enables web-based management of back-end data servers as well as front-end web servers, and is crucial to effective management of websites via Internet connections. Note that due to the nature of HTTP (asymmetric, connectionless) and its status (i.e. no IETF or W3C standards for event delivery over HTTP) the web-based management capabilities include schema and instance puts, gets, and enumerations, and generalized queries, but not event delivery.

4.3 Distributed Data Centers

WMI web-based enterprise management also supports the global operations center/distributed data center model. In this scenario an organization maintains a central operation/management center at one or two global locations (for example, New York and
Singapore). Data centers (or branch offices) themselves can be distributed far more broadly, such as data centers in London, Chicago, and Tokyo, with management driven by the centralized operations centers. Management architectures can include the web-based approach described above, or access to servers configured to act as management hosts with connectivity over management LANs to target hosts.

Distributed management is illustrated in the way that WMI is used in the maintenance of trust relationships between Domain Controllers. Domain Controllers require trust relationships between them that can be broken as a result of network outages. These trust relationships must be reestablished by a suitably authorized user – they cannot be reestablished automatically. WMI event detection was used to determine when the trust relationship is broken and the event delivery mechanism was used to send a mail message to the administrator responsible for reestablishing the relationship. The event detection and event delivery mechanisms are both defined as ordinary WMI instances. These instances can be represented in a compileable form and delivered using a directory-based policy to all possible Domain Controllers. Further as a result of the use of Policy, any time a server becomes a Domain Controller the outage detection mechanism will be installed as a result of the application of the policy without any intervention being required on the part of the user.

5. **WMI IMPLEMENTATION LESSONS**

Many lessons have been learnt since WMI implementation of the WBEM initiative started. However, we feel that the issues concerned with the CIM and schema
development are of primary interest to the journal audience.

Controlling the schema extension process is extremely difficult and requires that a given class have a single primary owner. It was assumed initially by the DMTF CIM community that the combination of inheritance and associations would allow additions to schemas to be made by multiple independent contributors without requiring any specific negotiation process between the contributors. In practice, this has not worked as it is essential for a given class to have a well define population that is established as the definitive set of instances within a given namespace. Multiple conflicting extensions of such a class are very difficult to detect ahead of time requiring considerable care on the part of users and class owners with respect to the circumstances under which extensions are permitted.

The DMTF attempted to address this issue by the introduction of the concept of a CreationClassName property. The intent of the CreationClassName property was to allow instances supplied by different vendors to be distinguished by adding to the key value the name of the leaf-most class used in the definition of the instance. This however defeated the purpose it was intended to address, namely schema extension. If an existing populated class acquires a new subclass through extension, the instances thereby acquire a new CreationClassName property value, thereby changing the key. Object identifiers (OIDs) cannot be assumed because the underlying instrumentation may not have persistent storage available to store an OID value. It follows that keys must be used as fundamental identifiers, therefore, key changes cannot be allowed.
The providers rely on an arbitrary range of data sources for the information they supply. These sources may include algorithms such as searching for specific file directories, interrogating data stores such as the Windows Registry or accessing remote devices such as network attached printers. Although the combination of Object Manager and providers can hide the structure of these various data sources from the client, their performance characteristics are evident in query performance. The issue is that the performance characteristics of instrumentation data are inherently unpredictable. The response to this is to provide caching using the event system to maintain the currency of the cache and assume that the majority of queries will be processed over cached data.

6. SUMMARY AND CONCLUSIONS

The purpose of the DMTF WBEM initiative is to define a non-proprietary set of environment-independent specifications to allow management information to be logically organized, making the structure of the management environment as homogeneous as possible. This homogeneity facilitates the development of distributed management applications and allows information to be shared between management applications operating in similar and dissimilar operating system environments. This helps reduce the total cost of IT management in the enterprise, allowing system problems to be diagnosed and resolved from a central location, thus making distributed networks much easier to manage.

Windows Management Instrumentation (WMI) can be summarized as follows:
• WMI is a key component of Microsoft Windows management services.
• WMI is a Windows-based implementation of the DMTF Web-Based Enterprise Management (WBEM) initiative, and is fully compliant with the DMTF CIM version 2.0 management schema definitions.

The WMI WBEM-compatible management architecture provides fully integrated operating system support for uniform systems and applications management based on CIM. Management applications can use the WMI technologies to provide a consistent approach that will reduce the maintenance and life cycle costs associated with managing web-based enterprises.

WMI can use information originating from diverse sources to monitor the health of an application, service, or an entire PC-based network. Thresholds and aggregate views of data can reconcile disparate information and events to diagnose problems and provide an accurate, detailed picture of the network—including potential for serious problems. When used in combination with scripting capabilities, WMI-supplied data can be used (on the local machine or remotely in a seamless way) for load balancing and event-triggered alarm, backup, or system shutdown decisions. And, when combined with the other Windows management technologies, WMI can help to simplify the task of developing well-integrated management applications that provide end-to-end Web-based network and systems management.

There are already several successful commercial applications produced by both DMTF member companies and independent companies representing a variety of software and
hardware vendors, and client organizations. Being built around WBEM technologies, these applications facilitate management of distributed environments by either making an extensive use of WMI data and events or producing new tools built upon WMI infrastructure.

Acknowledgement

The authors acknowledge the contribution made by the whole WMI team, and especially P. Thompson, A. Boshier, V. Joanovic, T. Muhlestein and R. Rao, for their constructive comments.
References


   Network and Systems Management Paradigm, Journal of Network and Systems

   Management of Enterprise Networks, Journal of Network and Systems Management,

5. F. Stamatelopoulos and B. Maglaris, Performance and Efficiency in Distributed
   Enterprise Management, Journal of Network and Systems Management, Vol.7, No.1,
   1999.

   Protocol for Multimedia, Journal of Network and Systems Management, Vol.8, No.1,
   2000.


   Framework for Systems Management, Journal of Network and Systems Management,

9. E.S. Al-Shaer, Active Management Framework for Distributed Multimedia Systems,

11. Specifications for CIM Operations over HTTP, Version 1.0, August 1999,

12. CIM XML DTD, Version 2.0, June 1999
    http://www.dmtf.org/download/spec/xmls/cim_dtd_V20.txt


14. R. Glasgow, The SNIA demonstrate the future at Storage Network World,

Authors’ short biography:

A. Retik is a Schema Program Manager in the WMI group. Before joining Microsoft, he was engaged in both industry (as a project manager) and academy (as a university researcher and lecturer). The latter included research, development and teaching in the areas of Knowledge Based Systems and advanced IT applications.

D. Fischer is an XML/HTTP Program Manager in the WMI group. Before joining the WMI group he worked in the Microsoft Internet Services Business Unit. Prior to joining Microsoft, he ran his own software business for five years, and before that worked for an international accounting/consulting firm.

S. Muzio is a Lead Program Manager in the WMI group. He has been with the group since its establishment.
Appendix: XML Access to WMI data via HTTP

a. an XML representation of a WMI class. A screen capture of Win_32LogicalDisk Class is shown.

b. an XML representation of WMI instances. This screen capture shows an enumeration of Win_32LogicalDisk instances with all properties.

c. a MOF representation of the WMI Win_32LogicalDisk instances.

d. a WMI query. This screen capture shows the result of the query related to the WMI instances in b:

   (select Description from Win_32LogicalDisk where Name = “A:").

This query reduces the amount of the data ‘on the wire’ by retrieving a specified data only.
a. an XML representation of the WMI Win32_LogicalDisk class

```xml
<?xml version="1.0" ?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0" >
 <MESSAGE ID="877" PROTOCOLVERSION="1.0" >
  <SIMPLERSP>
   <IMETHODRESPONSE NAME="GetClass" >
    <IRETURNVALUE>
     <VALUE.NAMEDOBJECT>
     <CLASS NAME="Win32_LogicalDisk" SUPERCLASS="CIM_LogicalDisk" >
     <PROPERTY NAME="Compressed" TYPE="boolean" />
     <PROPERTY NAME="DriveType" TYPE="uint32" />
     <PROPERTY NAME="FileSystem" TYPE="string" />
     <PROPERTY NAME="MaximumComponentLength" TYPE="uint32" />
     <PROPERTY NAME="MediaType" TYPE="uint32" />
     <PROPERTY NAME="ProviderName" TYPE="string" />
     <PROPERTY NAME="SupportsFileBasedCompression" TYPE="boolean" />
     <PROPERTY NAME="VolumeName" TYPE="string" />
     <PROPERTY NAME="VolumeSerialNumber" TYPE="string" />
     </CLASS>
     </VALUE.NAMEDOBJECT>
    </IRETURNVALUE>
   </IMETHODRESPONSE>
  </SIMPLERSP>
 </MESSAGE>
</CIM>
```
b. an XML representation of WMI Win32_LogicalDisk class instances

```xml
<?xml version="1.0" ?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0">
    <MESSAGE ID="877" PROTOCOLVERSION="1.0">
        <SIMPLERSP>
            <IMETHODRESPONSE NAME="EnumerateInstances">
                <IRETURNVALUE>
                    <VALUE.NAMEDOBJECT>
                        <INSTANCENAME CLASSNAME="Win32_LogicalDisk">
                            <KEYBINDING NAME="DeviceID">
                                <KEYVALUE>A:</KEYVALUE>
                            </KEYBINDING>
                        </INSTANCENAME>
                        <INSTANCE CLASSNAME="Win32_LogicalDisk">
                            <PROPERTY NAME="Caption" TYPE="string">
                                <VALUE>A:</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="CreationClassName" TYPE="string">
                                <VALUE>Win32_LogicalDisk</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="Description" TYPE="string">
                                <VALUE>3 1/2 Inch Floppy Drive</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="DeviceID" TYPE="string">
                                <VALUE>A:</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="DriveType" TYPE="uint32">
                                <VALUE>2</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="MediaType" TYPE="uint32">
                                <VALUE>5</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="Name" TYPE="string">
                                <VALUE>A:</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="SystemCreationClassName" TYPE="string">
                                <VALUE>Win23_ComputerSystem</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="SystemName" TYPE="string">
                                <VALUE>WMI</VALUE>
                            </PROPERTY>
                        </INSTANCE>
                    </VALUE.NAMEDOBJECT>
                    <VALUE.NAMEDOBJECT>
                        <INSTANCENAME CLASSNAME="Win32_LogicalDisk">
                            <KEYBINDING NAME="DeviceID">
                                <KEYVALUE>C:</KEYVALUE>
                            </KEYBINDING>
                        </INSTANCENAME>
                        <INSTANCE CLASSNAME="Win32_LogicalDisk">
                            <PROPERTY NAME="Caption" TYPE="string">
                                <VALUE>C:</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="Compressed" TYPE="boolean">
                                <VALUE>FALSE</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="CreationClassName" TYPE="string">
                                <VALUE>Win32_LogicalDisk</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="Description" TYPE="string">
                                <VALUE>Local Fixed Disk</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="DeviceID" TYPE="string">
                                <VALUE>C:</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="DriveType" TYPE="uint32">
                                <VALUE>3</VALUE>
                            </PROPERTY>
                            <PROPERTY NAME="FileSystem" TYPE="string">
                                <VALUE>NTFS</VALUE>
                            </PROPERTY>
                        </INSTANCE>
                    </VALUE.NAMEDOBJECT>
                </IRETURNVALUE>
            </IMETHODRESPONSE>
        </SIMPLERSP>
    </MESSAGE>
</CIM>
```
c. a MOF representation of the WMI Win32_LogicalDisk class instances

```c
instance of Win32_LogicalDisk {
    string Caption = "A:";
    string CreationClassName = "Win32_LogicalDisk";
    string Description = "3 1/2 Inch Floppy Drive";
    string DeviceID = "A:";
    uint32 DriveType = 2;
    uint32 MediaType = 5;
    string Name = "A:";
    string SystemCreationClassName = "Win32_ComputerSystem";
    string SystemName = "WMIXML";
};
instance of Win32_LogicalDisk {
    string Caption = "C:";
    boolean Compressed = FALSE;
    string CreationClassName = "Win32_LogicalDisk";
    string Description = "Local Fixed Disk";
    string DeviceID = "C:";
    uint32 DriveType = 3;
    string FileSystem = "NTFS";
    uint64 FreeSpace = 1275462656;
    uint32 MaximumComponentLength = 255;
    uint32 MediaType = 12;
    string Name = "C:";
    uint64 Size = 2111832064;
    boolean SupportsFileBasedCompression = TRUE;
    string SystemCreationClassName = "Win32_ComputerSystem";
    string SystemName = "WMIXML";
    string VolumeName = "";
    string VolumeSerialNumber = "18852B7C";
};
instance of Win32_LogicalDisk {
    string Caption = "D:";
    string CreationClassName = "Win32_LogicalDisk";
    string Description = "CD-ROM Disc";
    string DeviceID = "D:";
    uint32 DriveType = 5;
    uint32 MediaType = 11;
    string Name = "D:";
    string SystemCreationClassName = "Win32_ComputerSystem";
    string SystemName = "WMIXML";
};
```
d. the result of a WMI query

```xml
<?xml version="1.0" ?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0">
  <MESSAGE ID="877" PROTOCOLVERSION="1.0">
    <SIMPLERSP>
      <IMETHODRESPONSE NAME="ExecQuery">
        <IRETURNVALUE>
          <VALUE.OBJECT>
            <INSTANCE CLASSNAME="Win32_LogicalDisk">
              <QUALIFIER NAME="dynamic" PROPAGATED="true" TYPE="boolean" TOSUBCLASS="false" TOINSTANCE="true">
                <VALUE>TRUE</VALUE>
              </QUALIFIER>
              <QUALIFIER NAME="Locale" PROPAGATED="true" TYPE="sint32" TOSUBCLASS="false" TOINSTANCE="true">
                <VALUE>1033</VALUE>
              </QUALIFIER>
              <QUALIFIER NAME="provider" PROPAGATED="true" TYPE="string" TOSUBCLASS="false" TOINSTANCE="true">
                <VALUE>CIMWin32</VALUE>
              </QUALIFIER>
              <QUALIFIER NAME="UUID" PROPAGATED="true" TYPE="string" TOSUBCLASS="false" TOINSTANCE="true">
                <VALUE>{8502C4B7-5FBB-11D2-AAC1-006008C78BC7}</VALUE>
              </QUALIFIER>
              <PROPERTY NAME="Description" TYPE="string">
                <QUALIFIER NAME="CIMTYPE" PROPAGATED="true" TYPE="string" TOINSTANCE="true">
                  <VALUE>string</VALUE>
                </QUALIFIER>
                <VALUE>3 1/2 Inch Floppy Drive</VALUE>
              </PROPERTY>
            </INSTANCE>
          </VALUE.OBJECT>
        </IRETURNVALUE>
      </IMETHODRESPONSE>
    </SIMPLERSP>
  </MESSAGE>
</CIM>
```