A WEB-BASED SERVER MANAGEMENT
SYSTEM WITH IPMI AND WMI TECHNIQUES

Yi-Hsuan Yeh
Department of Information
Management
Oriental Institute of
Technology
58 Sze-Chuan Rd. Sec 2
Pan-Chiao, Taipei, Taiwan
National Taiwan University of
Science and Technology
amyyeh@mail.oit.edu.tw

Yu-Chin Szu
Department of Information
Management
St. John's University
499,Sec. 4, Tam King Road
Tamsui, Taipei, Taiwan
National Taiwan University of
Science and Technology
belinda@mail.sju.edu.tw

Yuan-Cheng Lai
Department of Information
Management
National Taiwan University of
Science and Technology
No. 43, Sec. 4, Keelung Road,
Taipei, Taiwan
laiyc@cs.ntust.edu.tw

ABSTRACT

With the rapid development of many new services in the Internet, more and more servers exist. IT managers need a convenient management tool to manage the servers of an enterprise. In this paper, we develop a Web-based Server Management system (WSM system) to help IT managers ease management of servers. The WSM system is run under Windows 2000/XP/2003 operating system. It can manage the server’s hardware health via Intelligent Platform Management Interface (IPMI) commands, and manage the server’s system resources via Windows Management Instrumentation (WMI) commands. The WSM system also provides the functionalities of managing server groups and scheduling the task. To easily control lots of servers, a three-tier management architecture is adopted. IT managers connect the servers via a centralized managing site, which is installed the WSM system, through Internet. This managing site will centrally manage a farm of servers.

KEY WORDS
IPMI, WMI, three-tier management architecture

1. Introduction

The rapid development of Internet and Web services lead to today's server environments increasingly diverse, distributed, and complex. Furthermore, the continuous changes in business conditions, mergers and acquisitions, and rapidly changing technologies have created a dynamic server environment that is difficult to manage. IT managers face with the more difficult challenge in managing the complex server environment, while providing stable services to customers. Therefore, they need a more effective management tool to reduce the complexity of managing diverse server environments by automating and unifying server management across the enterprise to reduce costs, improve service levels, and lower business risks.

Some management ways, such as SNMP (Simple Network Management Protocol), IPMI, WMI, were commonly used to manage servers. SNMP is an application layer protocol for allowing network administrators to manage network performance and find network problems [1]. IPMI is a specification for providing system management capability in hardware [2], and it includes a rich set of predefined commands and interfaces organized by some types of operations, such as reading temperature, voltage, fan speed, and chassis intrusion. WMI is a component of the Microsoft operating system for allowing users to virtually monitor each piece of the system either locally or remotely [3].

In our opinions, an ideal server management system should include several main characteristics as follows.

(1) Remote Management: The IT manager needs to have the capability of managing servers whenever and wherever.
(2) Task Scheduling: A task will be executed at the predefined time.
(3) Group Management: Many servers need to be managed.
(4) Out-of-band Management: The IT manager requires to grasp the newest status of servers, even when they are in power off.
(5) In-band Management: The IT manager needs to obtain system information when servers are working.

Some server management tools such as Arima’s RioView v1.1 [4] and Acer’s Acer Server Manager v5.3 [5] existed on the market. Both server management tools apply IPMI to support out-of-band management. Acer Server Manager also provides in-band management and group management. However, they do not provide the functionality of task scheduling. Besides, they need to install some specific software into the client. Therefore, the IT manager can’t execute remote management except the software has been installed.
In this paper, a total solution for server management is proposed. We use the techniques of Web, IPMI, WMI, and database to implement a new server management system, called a Web-based Server Management (WSM) system. Our WSM system supports the Web-based interface, causing that the IT manager can easily handle server’s status through a browser. IPMI can monitor and control server no matter the server is power on or power off. WMI is used to provide a common way to manage Windows operating system of a server when it is in power on. Lastly, using database and AT, which is a Windows schedule service, to implement the functionalities of group management and task scheduling.

The organization of this paper is as follows. Some backgrounds concerning IPMI, WMI, and the related works, are presented in Section 2. Section 3 describes the design and functionalities of our WSM system in details. Section 4 presents the implementation details of our WSM system. Some conclusions are given in section 5.

2. Backgrounds

The WSM system chiefly uses IPMI and WMI techniques to manage the server, where IPMI is out-of-band management and WMI is in-band management.

2.1 IPMI

The Intelligent Platform Management means that autonomous monitoring, controlling, and recovery features are implemented directly in Platform Management hardware and firmware. The key characteristic of Intelligent Platform Management is that platform management functions, such as monitoring, logging, and recovery control, are independent of main processor, operating system, and BIOS. Thus, even when the computer is in power off, these management functions can be still accessible.

IPMI is a standard which defines a common hardware interface to effectively support the health monitoring of the system hardware, such as system temperatures, voltages, fans, power supplies, etc. In addition to health monitoring, IPMI also includes local or remote system resets and power on/off capabilities. IPMI is a kind of hardware interface specification that is independent of the management software. Its monitoring and control functions can be exposed through some standard management interfaces, such as SNMP, WMI, CIM (Common Information Model), DMI (Desktop Management Interface). Fig. 1 depicts IPMI and the management software stack.

The heart of IPMI architecture is a small and separate processor, called the baseboard management controller (BMC), on the motherboard. BMC connects to the main processor, on-board sensors, and other hardware elements. It manages the interface between system management software and platform management hardware, provides monitors, and also controls the health of the system’s physical hardware.

Fig. 1 IPMI and the management software stack

2.2 WMI

WMI is a core management capability built-in the operating system of Windows 2000, Windows XP, or Windows 2003. The purpose of WMI is to provide a common way to manage almost all of the Windows resources, such as system information, BIOS information, drive partition information, operating system information, and environment variables.

Most IT managers use lots of graphical administrative tools to manage Windows resources, such as disks, folders, file systems, printers, processes, registry settings, security, services, users, groups. Prior to WMI, these Windows graphical administrative tools relied on Win32 application programming interfaces (APIs) to access and manage Windows resources. Because the only way that users could programmatically access Windows resources before WMI was through Win32 APIs, this situation lets the IT managers have no easy way to manage Windows resources using popular script languages since most script languages cannot call Win32 APIs directly. WMI uses the Component Object Model (COM) interface to make system management easier and more secure. In addition, WMI offers a variety of programming interfaces, such as Visual Basic, C++, open database connectivity (ODBC), Perl, Active Server Pages (ASP), ASP.NET, and scripts within standard HTML pages.

Fig. 2 illustrates the WMI architecture. The WMI Consumer is a script application, management application, Web-based application or administrative tool, that accesses and controls management information through the WMI infrastructure. The system service, WinMgmt.exe, being the core of WMI and known as Computer Information Model Object Manager (CIMOM), handles the interaction between WMI consumers and WMI providers. When the WMI consumer sends a request for
obtaining management information, CIMOM gets this request, identifies which provider has the information, gets the data through this identified provider, and finally returns it to the WMI consumer. The WMI Provider acts as an intermediary between the Managed Resource and CIMOM. The repository maintained by CIMOM contains object, class, and instance definitions that can be used to access and maintain system configuration information. The Managed Resources, such as the computer system, disks, files, folders, and applications, are the resources which can be managed using WMI.

3. System Design
This section presents an overview of the WSM system. In the following, we will describe the system architecture, the management flow, the design issues, the user interface, and the functionalities of the WSM system.

3.1 System Architecture
As shown in Fig. 3, our WSM system adopts a 3-tier structure, which composed of the managing site, the client cluster and the server cluster. The reason of adopting a 3-tier structure is that a 2-tier structure is not an appropriate way to manage lots of servers because the users need to remember each server’s IP, account, and password. Therefore, the 3-tier structure having a managing site, which centrally manages each server’s information, is adopted and can lighten the load of the client side.

In IPMI, BMC has different communication interfaces, such as IPMB (Intelligent Platform Management Bus), serial/modem, LAN (Local Area Network), ICMB (Intelligent Chassis Management Bus), PCI Management Bus, and System Interface. Since we want to use a centralized managing site to manage all servers, the LAN interface is the best choice, thus causing that the managing site and the server cluster must be in the same LAN.

In the server side, each managed server must support IPMI and install Windows operating system like Windows 2000/XP/2003 or above. The client side needs using a WWW browser to connect with the managing site via Internet. The managing site needs installing IIS, SQL server 2000, .NET Framework environment, Windows operating system, and the WSM system.

3.2 Management Flow
Fig. 4 exhibits the management flow which steps are summarized as follows.

- Step 1: Users connect to the managing site and make the authentication operation. After authentication, users can manage servers via the managing site.
- Step 2: Users select a managed server or group. After this selection, they can execute any function in the managing site. The WWW browser takes the argument values from the input components, for example, like INPUT tags, and passes them to the managing site in an URL (Uniform Resource Locator) request that identifies which ASP.NET program shall be used. The managing site then executes this corresponding ASP.NET program, such as Group Management (go to step 3), Task Scheduling (go to step 4), WMI (go to step 5), or IPMI (go to step 6).
- Step 3: The managing site executes “add node/group” command or “delete node/group” command, and also updates database information.
- Step 4: The managing site adds a task to “Scheduled Tasks” or deletes a task from it, and also updates database information.
- Step 5: The managing site connects to the managed server by using WMI message. After successfully connecting to the managed server, the ASP.NET program sends a WMI command to the managed server and receives its response.
- Step 6: This step is similar to step 5, except using IPMI message.
3.3 Design Issues
In this subsection some design issues are described.
• Finding IPMI-enabled servers: The managing site uses IPMI’s “Ping” message to discover the managed servers that support IPMI. The message is similar with the ping application that is used to test host responses over networks. If a managed server supports IPMI, it returns IPMI’s “Pong” message to the managing site.
• Group Management: The WSM system uses a two-layer structure to implement the functionality of Group Management. The first layer includes group name and the second layer includes the list of the servers. These managed servers are displayed in a tree, making IT managers easily manage each server or group.
• Security: The WSM system uses Secure Sockets Layer (SSL) to transmit private document via the Internet.
• Task Scheduling: Two methods, AT [6] and Schtasks [7], are used to schedule a task in Windows operating system. Schtasks has more options and more completeness than AT, but it is not supported in Windows 2000 operating system. Therefore, we decide to use AT, although it has less schedule options.
• Database: A database is created in SQL Server 2000. The WSM database has seven tables, namely, Account, Config, Found, Root, Node, Schedule, and Schedule Log. The Account table storing user id and password is used to check the login. The Config table stores Retry and Timeout values. When IT managers perform “Search BMC Server” by specifying the IP range, the system will store these found servers’ IPs into the Found table. The Root table stores each group’s name. The Node table stores each server’s information such as Server IP, WMI Account and Password, and Group ID. The Schedule table stores each scheduled task’s data including Task ID, Task, and Start Time. Once a task is activated by the schedule service in the specified time, the Schedule Log table saves the execution result including task name, execution time, target server, and the status of this task, i.e. success or failure.
• Authentication: When an IT manager connects to the managing site by using a WWW browser, a login page will be displayed for requesting the IT manager to provide the username and password for authentication. After the IT manager has been authenticated, the authentication information will be stored in the session. After authentication, the IT manager can use the functions in the managing site until this session storing authentication information expires.

3.4 User Interface
The WSM system’s interface is divided into three parts, including top frame, left frame, and right frame, as shown in Fig. 5.

The top frame is the Main Menu Toolbars, which has four tabs in the top-level menu bar. These tabs delineate the functions of the managing site – Configuration, Command, Monitoring, and System Information. To operate any of these functions, the user moves the mouse pointer to an appropriate tab, and then the secondary menu will be shown according to the selected function.

The left frame is the Managed Server Group, which displays the managed server groups by using a tree-based structure. When the mouse pointer is moved on a server name, the IP address of this server will be shown on the screen. Once a server name is clicked, it will be displayed with red color, implying this server is being managed now.

The right frame is the View Window, which displays the information according to the selected server and function.

3.5 Functionalities
The WSM system has four major functionalities, including Configuration, Command, Monitoring, and System Information, which are plotted in Fig. 6 in detail.

3.5.1 Configuration
The Configuration menu offers five basic configuration settings.
• Connection Settings: This function edits the Retry and Timeout values that will be stored in database.
When users attempt to connect to an IPMI-enabled server, the WSM system will load Retry and Timeout values from database.

- **Sensor Settings**: This function sets the sensor thresholds of various system components. This page shows information including Sensor Name, Sensor Number, Sensor Type, Upper and Lower Thresholds. The first three are not editable. For setting Upper and Lower Thresholds, three kinds of thresholds, Warming, Critical, and Non-Recoverable thresholds, can be set to a given sensor. The Upper and Lower Thresholds are used to determine when a possible component failure is imminent. Once a sensor detects the value above Upper Threshold or below Lower Threshold, an entry will be recorded in the System Event Log.

- **Privilege Settings**: This function sets access permissions for up to four users. IT managers can disable or enable each user individually. Each user will have one of four different privileges:
  1. **Administrator**: Users have all access rights to a managed server.
  2. **Operator**: Users have the same access rights as Administrator except that they cannot change user privileges.
  3. **User**: Users can only view information.
  4. **Callback**: Users do not have any right to access any information.

- **Set SEL (System Event Log) Timestamp**: This function gets the time or initializes the time in the SEL device, which uses this time for Event Timestamping.

- **Group Management**: The function is divided into three sub functions, Manual Management, Search BMC Server, and Found Server Management. Manual Management adds, deletes, and edits the server and server group configuration, such as Display Name, Server IP, BMC Account, BMC Password, Server Name, Server Account, Server Password, and Server Description. The Server IP, BMC Account, and BMC Password are used when executing IPMI commands. The Server Name, Server Account, and Server Password are used when executing WMI commands. In Search BMC Server, users can specify IP range to search IPMI-enabled servers. If any IPMI-enabled servers in the specified IP range are found, these IPMI-enabled servers will be shown in a list box on the browser and saved in the database. The Found Server Management moves the found IPMI-enabled servers to a specified group.

### 3.5.2 Command

The Command menu has five functions as follows.

- **Get Status**: This action checks whether a server is in power on or power off. It also shows whether the last power on was activated via IPMI command or not.

- **Power Control**: This function supports a mechanism for providing Power Up, Power Down, Power Cycle, Hardware Reset, and Software Shutdown. The Hardware Reset is same as pressing the reset button on a computer. It is similar to power cycle but does not shut down OS software. The Software Shutdown shuts down the OS software only, but it does not shut down the managed server.

- **Schedule**: This function sets a schedule to run a task in the specified time. Users can select one of above five power management commands, a single server or group, starting time, and days of week or month. This scheduled task will be continuously repeated until users delete this schedule.

- **Schedule Information**: This function shows the information, including Task ID, Task Name, Days of Month, and Days of Week, of each scheduled task.

- **Schedule Event Log**: Once a scheduled task has been started, the system will execute this task and record the results into the database. This function captures history reports, such as Server Name, Task, Execution Time, and Result, about task execution. It also supports the delete mechanism of event log.

### 3.5.3 Monitoring

This menu has three functions, which mainly monitor server’s hardware information and read some useful information.

- **Sensor Monitoring**: This function supports monitoring various server components such as Fan Speed, CPU Temperature, and System Temperature. This function also sets a periodical monitoring interval that can be enabled or disabled. Once the Periodical Monitoring was enabled by users, this page will refresh every interval until it is disabled.
● Device Information: This function shows the information, including Chassis Information, Board Information, and Product Information, of a managed server.
● System Event Log: This function reads the SEL records from the BMC chip. Each page shows 20 SEL records at most.

3.5.4 System Information
This menu gets systematic information form a managed server by using WMI technology. Follows are the list of system asset able to be viewed.
● BIOS: Display Manufacturer and Release Date information about BIOS.
● Physical Drives: Get information about Floppy, Hard Disk, and Compact Disk.
● Logical Drives: Show each drive’s information such as Drive Letter, Description, Volume Name, File System, Free Space, and Size.
● Memory: Display Total Physical Memory, Total Virtual Memory, and Available Virtual Memory.
● Processor: Display processor’s information, including Description, Processor, Manufacturer, Processor Speed, and Revision.
● Network: Show the information of all network adapters, including Domain Name, Network Adapter, MAC Address, IP Address, Subnet Mask, Primary Gateway, and DNS Server.
● Operating System: Show information such as Operating System, Manufacturer, Version, Service Pack Version, and Registered User.
● System Drivers: Display information about the running system drivers that have been installed on the managed server, including Driver Name, Description, Type, and State.
● Process: List all the currently active processes on the managed server with their Name, Process ID, and Memory Usage.
● Services: Show information about the running services, including Name, Type, Status, and Start Mode.

4. Implementation
In this section, we will describe how to implement the WSM system in details. Fig. 7 shows the architecture of the managing site. There are four modules in the managing site including IPMI Module, WMI Module and Task Scheduling Module.

4.1 Development Environment
Our development environment is similar to the managing site’s environment, but needs to additionally install Visual C++ 6.0 and Dreamweaver 2004 MX additional. The former is used to develop IPMI DLL and IPMI COM object, and the latter is a web page editor used to develop ASP.NET Website.

4.2 IPMI Module
We use the DLL (Dynamic Link Library) and COM (Component Object Model) object to implement this module. IPMI DLL implements low level actions, namely, IPMI packet transmission. It receives parameters from the IPMI COM object, fills these parameters into the proper packet fields, and then sends the packet on a socket. The IPMI COM object loads this IPMI DLL and implements various IPMI functions. ASP.NET creates an IPMI COM object and uses it to invoke all functions related to IPMI functions.

An example about IPMI command is presented. The IPMI command basically uses three methods in the IPMI COM object. The first method, Connect() method, is used to connect with a managed server. This method takes five arguments, namely, ipArray, username, password, retry, and timeout. The ipArray is an array that stores an IP address, and the username and password are BSTR type that is COM object’s string type. The retry and timeout are integer type. This method returns a Boolean value to judge whether this Connect() method succeeds or fails. The second method is one of the IPMI commands, such as PowerOn() and PowerCycle(). After connecting to a managed server, an IT manager can execute lots of IPMI commands. The last method is Disconnect(), which is used to disconnect from a connected server.

4.3 WMI Module
WMI is based on the concept of the WMI provider, namely, the Win32 provider. Many of WMI classes are available and organized into groups identified by namespaces, which will be used when users attempt to create a WMI object.

Now, we demonstrate how to use a WMI Win32_Service class to display the list of all services installed on the system. First, some basic information, which is used in connecting with a managed server, is set. These basic information include server’s name or server’s IP address, login account, and login password. The login account must be in the administrator group of the managed server. Then an object SWbemLocator is created by using VBScript CreateObject() function call. After creating the SWbemLocator object, we use the SWbemLocator:connectServer() method to obtain a
SWbemServices object that represents a connection to a namespace on a local or remote host. The SWbemServices::InstancesOf( ) method creates an enumerator which returns the instances of a selected class. Herein all instances of the Win32_Service class are requested. After getting the Win32_Service instances, the interested information is iteratively displayed through them.

4.4 Task Scheduling Module
The Win32_ScheduledJob WMI class is used to create a task. When a scheduled task is executed, a batch file is called to run a WSH (Windows Script Host) program [8], which will query the database to find out all servers’ information related to this task. Then this program creates an IPMI COM object to execute this task. WSH allows users to write scripts to automate tasks on a Windows operating system.

The Win32_ScheduledJob class represents a task created with the AT command in a command window from Start/Run, or from Control Panel/Scheduled Tasks. We can enumerate or delete any existing tasks displayed in the Control Panel/Scheduled Tasks by using the Win32_ScheduledJob instances.

Each scheduled task is stored persistently, implying that the scheduled task will not disappear after computer reboot. Tasks are scheduled according to UTC (Universal Coordinated Time) with bias offset from GMT (Greenwich Mean Time). The Win32_ScheduledJob class returns the local time with UTC offset when enumerating an object, and converts it to local time when creating a new task.

The Win32_ScheduledJob class defines two methods, Create( ) and Delete( ). The parameters of Win32_ScheduledJob::Create( ) method are as follows.

- Command: A batch program or command that the schedule service uses to invoke the task.
- StartTime: This parameter is UTC time to run the task.
- RunRepeatedly: This parameter sets that a scheduled task should run repeatedly or once.
- DaysOfWeek: This parameter is used to specify the days of the week to run this task. If the task is scheduled to run on multiple days of the week, the values must be jointed using logical OR.
- DaysOfMonth: This parameter is similar to DaysOfWeek, but it is for the days of the month.
- InteractWithDesktop: If the value is true, the user can give some inputs to a scheduled task while the task is executing.
- JobId: It identifies the task number.

When a scheduled task is activated by the system, it executes a batch file with a task number. Then this batch program calls the WSH program to perform this task. The WSH program first searches this task’s information including Task, Type, and ID from database. The Type value is used to distinguish that the target is a single server or a group. If Type is “Server”, then WSH queries this server’s information from the database according to the ID value, and connects to this managed server. According to the specified Task content (Power On, Hardware Reset, etc.) to execute this task, and then disconnect this connection from the managed server. Finally, it records the task’s results into the database. If Type is “Group”, then it searches all servers’ information belonging to this group, and iteratively executes each server’s task.

5. Conclusion
The growing up of network environments causes that IT managers look for best software to conveniently manage pervasive servers. A system, Web-based server management system, is developed to help IT managers to easily manage lots of servers. We adopt two kinds of techniques, IPMI and WMI, for out-of-band management and in-band management, respectively. This system lets IT managers monitor and control servers easily and effectively.

Reference