Towards Ubiquitous EWS-based Network Management

Hong-Taek Ju and James Won-Ki Hong
DP&NM Lab.
Dept. of Computer Science and Engineering
POSTECH, Pohang Korea
Email: {juht, jwkhong}@postech.ac.kr
http://dpnm.postech.ac.kr/

Abstract

Most Internet networking devices are now equipped with a Web server f or providing Web-based element management so that an administrator may t ake advantage of this enhanced and powerful management interface. On the other hand, for network management, an administrator normally buys and de ploys SNMP-based network management platform to be customized to his n etwork. Each management scheme has mutually exclusive advantages; conse quently, two schemes coexist in the real world. This results in both a high de velopment cost and a dual management interface for administrator. We prop ose an embedded Web server (EWS)-based network management architectur e as an alternative to an SNMP based network management and to leverage on already existing embedded web server.

We extend EWS-based element management architecture to the network management architecture. Our proposed architecture uses HTTP as a commu nication protocol with management information and operation encoding. Fur ther we designed a management system on the basis of our proposed architec ture that supports basic management functions.

Introduction

- World-Wide Web (WWW) is one of the most widely used Inter net applications
- Web-based Network Management is the use of this technology to manage networks and systems
- Key technologies
  - HTML, HTTP, Web Browser & Servers, Java, CGI, XML, etc.
- Industry Standards for Web-based Network Management
  - Web-Based Enterprise Management (WBEM from DMTF)
  - Java Management eXtension (JMX from Sun)
- Benefits of Web-based Network Management
  - Reduced development costs by using open technology
  - Unification for separated management platforms
  - Simplification by ubiquitous and standard user interface

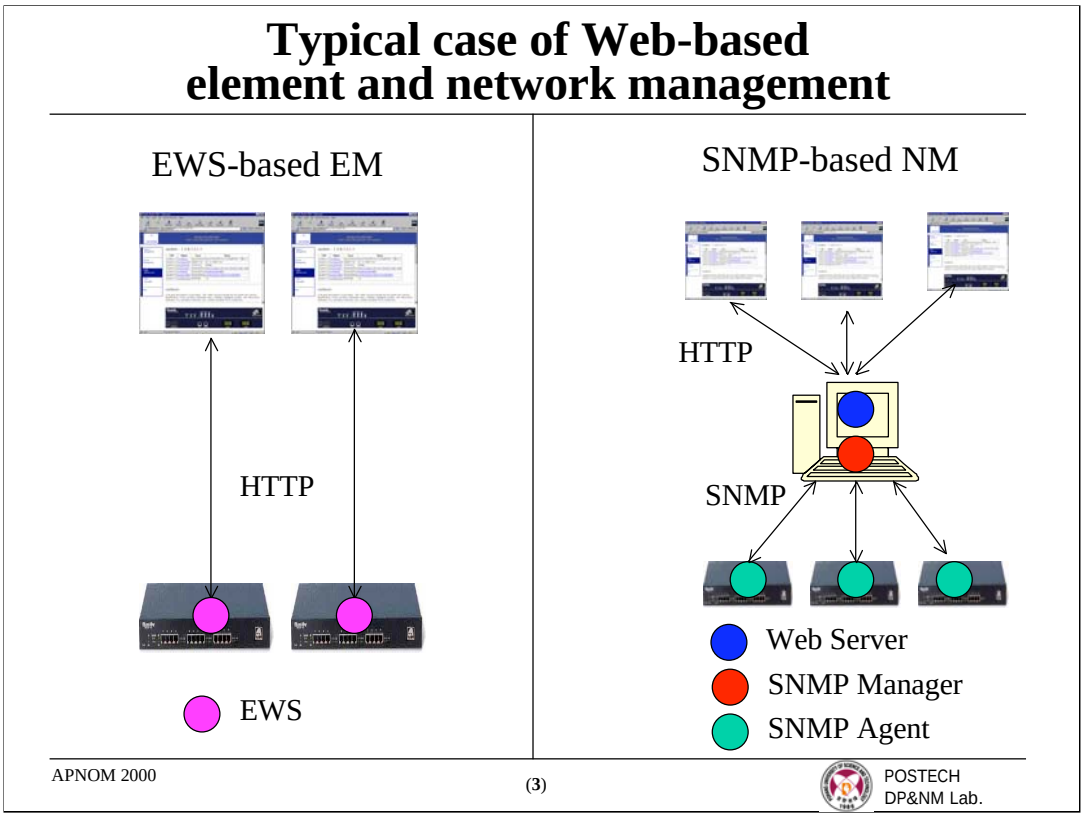
1. Introduction

World-Wide Web (WWW) is one of the most widely used Internet application [3]. The technology is very rapidly penetrating many social and business areas. The system and network management are n o exception. Web-based network management is the use of this technology to manage networks and sy stems.

The HTTP (Hypertext Transfer Protocol)[5] is the primary transfer protocol used by Web and the HTML (Hypertext Markup Language)[6] is a platform-independent document description language. T ypically in Web-based Network Management, HTTP is used as the transport of management informat ion in HTML format between communicating entities: Web server and browser. One of the key techno logies in Web is the Java that has high portability and code mobility. These unique features of Java ma ke it a challenge for building new management solutions. The eXtensible Markup Language (XML)[1 4] are designed to add structure and convey information about documents and data. Management infor mation represented in the form of an XML document can be useful for passing data between managem ent applications. All of these HTML, HTTP, Java, XML and the other Web technologies are playing a n important role in Web-based network management and recent advance in this technology astonishin g; in fact, experts cannot predict improvements beyond several years in Web-based network managem ent [5, 6, 10, 14].

Two principal industrial bodies are playing a leading role in contributing standardization to Web-ba sed network management: the Web-Based Enterprise Management (WBEM) [8] and the Java Manage ment eXtension (JMX) [9]. The WBEM from Distributed Management Task Force (DMTF) is an initi ative based on a set of management and Internet standard technologies developed to unify the manage ment of enterprise computing environments. WBEM provides the ability for the industry to deliver an integrated set of standard-based management tools leveraging the emerging technologies such as XML [14] and CIM [15]. The JMX is based on Java technology so it draws on Sun's experience with Java management. JMX provides the tools for building Java based solutions for managing devices, applicati ons and service-driven networks.

There are three primary benefits in applying Web technology to network management. The first is t hat development costs can be reduced by using open technology; plenty of open source and supporting tools. Further, the platform-neutral feature of Web technology makes it possible to unify the network a nd system management for separated management platform. Finally, the Web browser is ubiquitous, s o that Web-based management user interface is easy and simple for most, if not all, operators.



2. Typical case of Web-based element and network management

2.1 HTTP-based element management using Embedded Web Server

Network devices can be equipped with Web server to provide Web-based element management. Th is type of Web server is called an Embedded Web Server (EWS) [1, 2, 4]. Most commercial network d evices, such as routers, bridges, and hubs, are equipped with EWS. For the EWS equipped devices, ad ministrators point their Web browsers at the home page residing within the devices in order to configu re, monitor and control the network device.

This element management scheme provides an administrator with a simple but enhanced, more po werful and ubiquitous user interface. Moreover, there are no porting and distribution efforts for user ap plication program since the Web browser of administrator's computer is enough for managing network element.

2.2 SNMP-based network management using front-end Web Server

The SNMP manager program runs as an application program over the operating system in general c omputer and collects management information from network and system devices based on the SNMP f ramework [7]. When a user points his or her browser at the Web page provided by front-end Web serv er of the SNMP manger, aggregated management information is relayed to the browser.

This network management scheme can leverage the capabilities provided by the host operating syst em, such as increased memory, processing power and storage space. A more important point is that thi s mechanism provides management information for network level that was aggregated, processed by a SNMP manager.

This type of Web access serves as a useful addition to existing management platform, particularly t hose based on SNMP. The leading providers of management platforms, such as Cabletron, Computer Associates, IBM/Tivoli, Hewllet-Packard and Sun equip their products with Web-extensions.

Dual Management Interfaces

- Two models have pros and cons
  - Flexibility and Development cost: EWS-based EM >> SNMP-based NM
  - Scalability and Interoperability: EWS-based EM << SNMP-based NM
- Therefore, two models coexist in real world
- For users
  - Enhanced user interface
  - But still too expensive for network management by SNMP-based NM
  - Another interface needed for element management by EWS-based EM
- For equipment vendors
  - Put EWS and SNMP agent into network devices
  - Poor time-to-market and high cost

Why not extend the EWS-based element management to the network management?

2.3 Dual Management Interface

We introduced two typical Web-based network management schemes: an HTTP-based element management using Embedded Web server (EWS-based EM) and an SNMP-based network management using front-end Web server (SNMP-based NM). These two schemes have their own pros and cons.

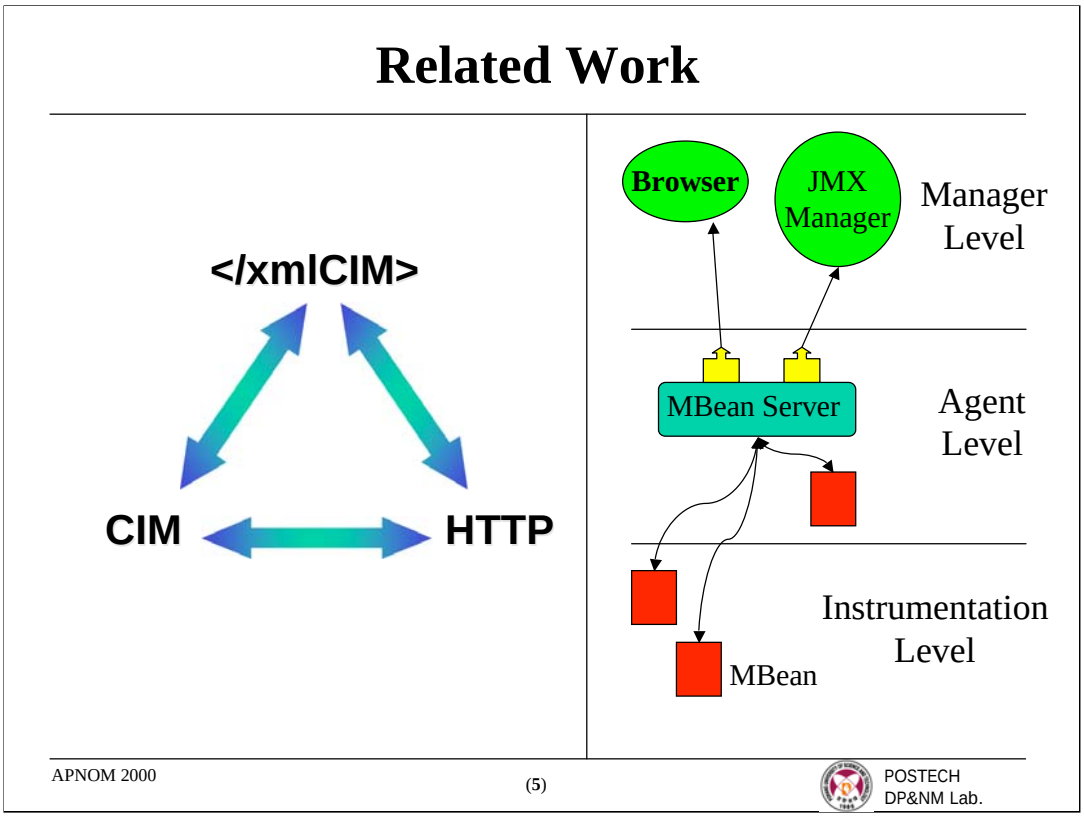
With respect to flexibility of management function, EWS-based EM is better than SNMP-based NM. Through the EWS, the device vendor controls everything about the device and its management from operation to user interface. The management function only depends on Web browser. This feature makes it possible to create device-specific Web pages, with control and user interface. On the other hand, SNMP-based NM has a limitation in functions it provides. Functional limitation results from the simplicity of SNMP. For instance, a version control function of the device program is provided by most management user interfaces from EWS-based EM, not from SNMP-based NM. Version control is essential in device management, especially the firmware upgrade capability.

EWS-based EM has its limitations with respect to scalability – configuring hundreds of routers and switches via a Web browser is simply not scalable. If there are many EWS-equipped network elements typical for large systems and networks, an administrator must open a Web browser for each device. This approach also tends to be device-centric and may not be able to provide logging and other high-level management capabilities that are normally essential for network management. On the other hand, SNMP-based NM normally has no limitation in scalability.

Each model (EWS-based EM and SNMP-based NM) has mutually exclusive advantages, which coexist in real world. Moreover, most devices support two models: network management such as topology information, alarm correlation and history can be made available to external SNMP manager based on SNMP. And element management such as system and protocol configuration and firmware upgrade is integrated within the device and provided through the Web browser.

Administrators can enjoy the enhanced user interface by way of applying Web technology to the network management. They still need an expensive network management platform for SNMP-based network level management. And they must use another user interface for device specific element management provided by EWS. For device vendors, they must put EWS and SNMP agent into the network device. Difficulty in supporting dual management interfaces result in poor time-to-market and high cost in development.

The obvious questions is **why not extend the EWS-based element management to the network management?** The goal of our work is to answer this question, that is, to build an EWS-based network management framework.



3. Related work

Recently, there are two promising approaches in Web-based management from industrial standardiz ation bodies: Web-Based Enterprise Management (WBEM) [8] and Java Management eXtension (JM X) [9]. WBEM multi-vendor alliance launched in July 1996 and worked for establishing standards for Web-based network management software. In 1997, WBEM adopted HTTP as its transfer protocol an d selected the Extensible Markup Language (XML) [14] as a representation for management informati on.

DMTF and WBEM worked together by giving the way for the encoding of the Common Informatio n Model (CIM) schema in XML [14, 15]. The CIM is an object-oriented information model, standardi zed within the DMTF for the purpose of providing a conceptual framework within which any manage ment data may be modeled. Allowing CIM information to be represented in the form of XML brings a ll the benefits of XML and its related technologies to management information which uses the CIM m eta-model [15]. The XML encoding specification defines XML elements, written in Document Type D efinition (DTD), which is used to represent CIM classes and instances. The encoded XML message co uld be encapsulated within HTTP. Further, WBEM defines a mapping of CIM operations onto HTTP t hat allows implementations of CIM to operate in a standardized manner. Much work in WBEM is curr ently under way: seventeen working groups are updating specifications. The result from WBEM is fair ly stable, but still not quite ready for deployment.

Another promising approach to the Web-based management is being realized by Sun: JMX (forma lly JMAPI) [9, 16]. Sun announced JMX in order to provide ubiquitous management framework and p romote an abundance of management application in Java. Based on the early JMAPI work as well as r esearch taken from Java DMK development, JMX ended public review in July 1999 and is awaiting co mpletion of the reference implementation [16].

The JMX specification defines the interface for basic services as a registry (*Mbean Server*) for *Mbe ans* (JavaBeans for management) [9, 10]. These services enable agents to manage their own resources and let managers forward information back and forth between agents and management applications. In the JMX architecture, both services and devices are treated as managed objects [9]. The components, *Mbeans*, can be added and removed as needs evolve. Appropriate protocol adapters can provide a reco gnizable object to the Browser or JMX manager whose specification is under way. JMX depends great ly on Java. In order to be instrumented in accordance with the JMX, a resource must be fully written in the Java programming language or just offer a Java technology-based wrapper. Java Virtual Machine i s a basic requirement for the management application. This heavy technology dependency on Java res ults in less generality.

Target Domain

- Closed target network
  - Results from standard bodies are still immature
  - There are few commercial devices conforming to the standard s
  - Outputs from EWS varies in format and are hard to manipulat e
- Enough computing resources in device
  - Recently, great advancement in hardware technology made it possible for a device to have plenty of computing resources
  - This advantage must be of benefit to EWS-based NM
- No SNMP integration
  - To avoid duplicated investment in closed targets
  - Devices are equipped with EWS only (i.e., SNMP Agent not n ecessary)

4. Designed Architecture

4.1 Target domain

As mentioned earlier, our goal is to develop an EWS-based network management system (NMS). T he management system assumes that all devices are equipped with EWS. And it must provide all the f unctionalities and information that SNMP-based NMS provides and more. Moreover, low developmen t cost and reduced development time must be considered in management system development. After al l, powerful, user-friendly, ubiquitous, flexible and scalable management interfaces will be offered to th e administrator.

There are some issues involved in selecting a target domain. The first issue concerns the organizati on of the target network. A management target network can be classified into two categories: closed (h omogeneous) and open (heterogeneous) network. A closed target network is composed of the same fu nctional devices from different vendors or different devices from one vendor. Usually, a management system targeting open network is developed as a family of products for managing a single vendor's de vice. An open network is composed of devices having different functionalities or made by different ve ndors. Usually, a management system targeting open network is deployed for managing an enterprise network as a management platform. Intuitively, it is easier to apply an EWS technology to the closed t arget network than the open target network.

A second issue concerns the extent of device's computing resources. Network devices have differe nt computing resource levels, such as speed of CPU, size of RAM & ROM, etc. The functionality of a device is very important factor in architecture design. Differences in device's functionality have effect on functionality assignment.

A third issue deals with integration with an SNMP. SNMP is a very important legacy protocol in the systems and network management application. Even though the target device for management is equip ped with EWS, most development features, such as design and implementation, depends on whether t he devices have SNMP agent. Another point to be considered is whether devices having SNMP agent only must be included or not in the target managed network.

Considering these, we narrow down the target network domain into a closed network with rich com puting resources in the device and no integration with SNMP. Results from standard bodies that are int roduced before are still incomplete. There are few commercial devices conforming to the standards. O utputs from EWS varies in format and are hard to manipulate. With respect to interoperability, it is mo re realistic to select closed network as a target domain. Recently, great advancements in hardware tech nology make it possible for a device to have richer computing resources. This advantage must be of be nefit to EWS-based network management development. To avoid duplicated investment in closed targ ets, we assume that devices are equipped with EWS only.

Architecture for EWS-based NM

- Extended version of EWS-based element management architecture: POS-EWS
- Thin-Client & Fat-Server paradigm
- Two model: 2-tier and 3-tier architectures
- Uses Java technology
- Communications using HTTP
- Operation & Information Encoding
- Supported basic management functions
  - Notification, Data collection, Agent discovery, Data setting

4.2 Design concept

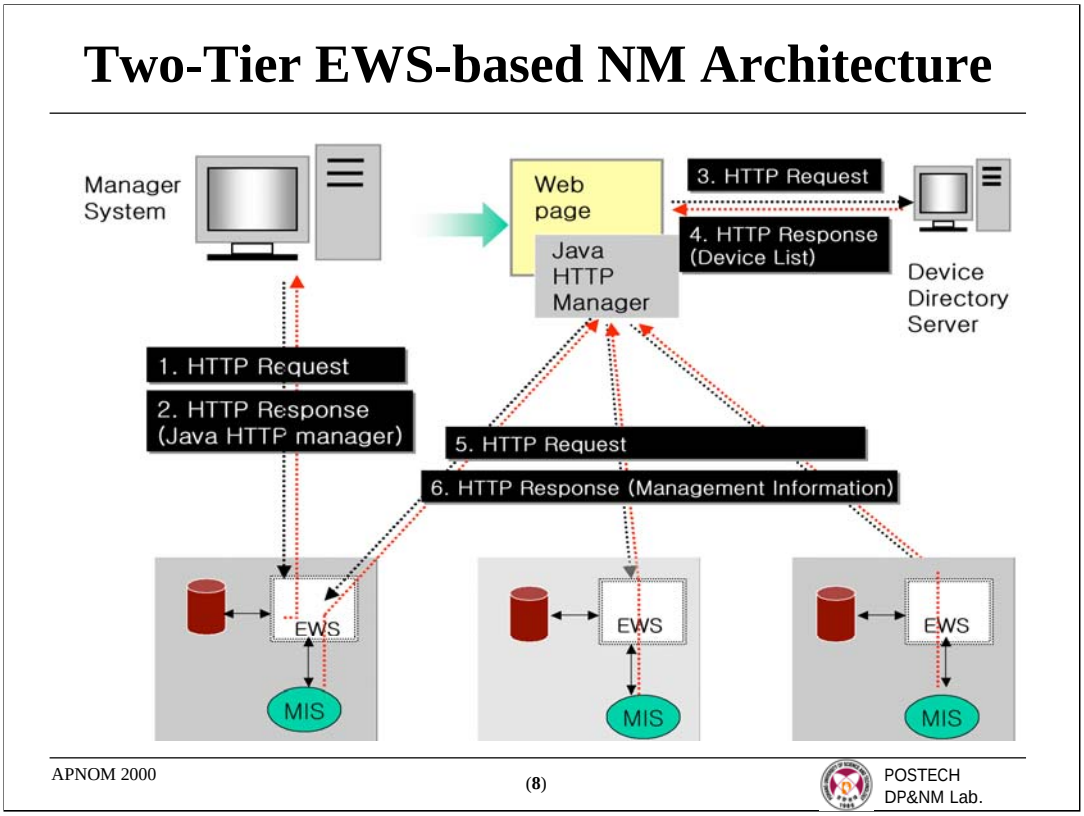
With the defined target domain, we designed the architecture for an EWS-based network managem ent system. In the previous work, we proposed an element management architecture having an EWS as a core component. We have developed an HTTP/1.1 compliant embedded Web server (called POS-E WS) that supports our proposed architecture [4, 13]. We also extended the EWS-based element manag ement architecture to the network management architecture.

We applied the thin-client and fat-server paradigm to the architecture. From numerous computing r esources in the device, we deduced this design concept. The first extended version from element mana gement architecture is 2-tier architecture, thereafter modified to a 3-tier architecture as a provision for the lack computing resources in device.

We use Java technology, especially the Java applet. The Java applet is downloaded from the Web s erver and run on a Web browser. The Java applet is a mobile code over the Internet. They are stored i n Web server and executed on the browser. There is no Java execution environment on devices. The Ja va applet has an inherent security problem: it is restricted in accessing local disk, executing another pr ogram and network connection to other hosts by Java applet security mode (Sandbox model) [10]. Cod e signing extends applet capabilities to make network connection with other hosts.

HTTP is used as transport protocol between EWS-based agent and management station. We define d an information encoding scheme for management data. The encoded message is encapsulated into H TTP message. With the encoding scheme, management operation such as get and set are encoded into URL that is part of HTTP message [5, 11].

Our EWS-based network management system supports four basic management functions: notificati on, data collection, agent discovery and data setting. Notification determines which events in device o ccurs on the basis of the event message and customizes the event message to notify administrator. Dat a collection gathers management data from the device and stores the data it collects in the database. It also performs threshold monitoring and generates threshold events. Agent discovery polls an EWS age nt to initially discovery EWS equipped device and then detects configuration and status changes in the network. Data setting provides an administrator a mechanism for changing management information o f a device.



4.3 Two-tier EWS-based NM architecture

HTTP is a client-server scheme. One of its side effects is that first Web page is served from a Web server to a Web browser, and subsequent Web pages cannot be included in the Web display except for image. For a network management system, this is very strict limitation. Network management system must gather management information from multiple devices, and formatted management data from mu ltiple devices can be placed in the same page. This is why Java applets are used [10, 12].

Java applets are downloaded by a browser. Once the applet is loaded, it can control the location fro m where it receives its data and how to display or manipulate that data. Java applets by nature are cros s-platform and act the same within any browser. Fortunately, it is a straightforward task to design an a pplet to make connections with multiple devices if Java applets are programmed on the basis of a Java security model and signing utility.

Java implementation of an HTTP manager is a key component in a 2-tier architecture. The Java HT TP manager source code is written and compiled to produce a Java HTTP manager applet. This applet is stored in a network device and is transferred by the EWS to the browser over the network at run tim e. After loaded on the JVM of a browser, the Java HTTP applet communicates with EWS agents in the network and enables the administrator to control and monitor the network devices using HTTP [5, 10, 12].

An administrator starts the management task by making Web browser connection to one of the man aged devices. The connected device responds with a Java HTTP manager applet. First operation of the Java HTTP manager applet is to retrieve registered device list from the Device Directory Server. After retrieving device list, The Java HTTP manager communicates with devices specified in the list to perf orm a management task. Management Information Server (MIS) responds to the Java HTTP manager r equest. It performs basic management functions explained before: Notification, Data collection, Agent registration.

Communications using HTTP

- Java HTTP manager and management information ser ver communicate using HTTP
- Avoid new specific management protocol
  - Reuse existing communication protocol
- Avoid overhead of frequently setting up and tearing d own connections
  - By persistent TCP connection of HTTP 1.1
- Management operation encoding in URL
- Management information encoding in HTTP

4.4 Communication using HTTP

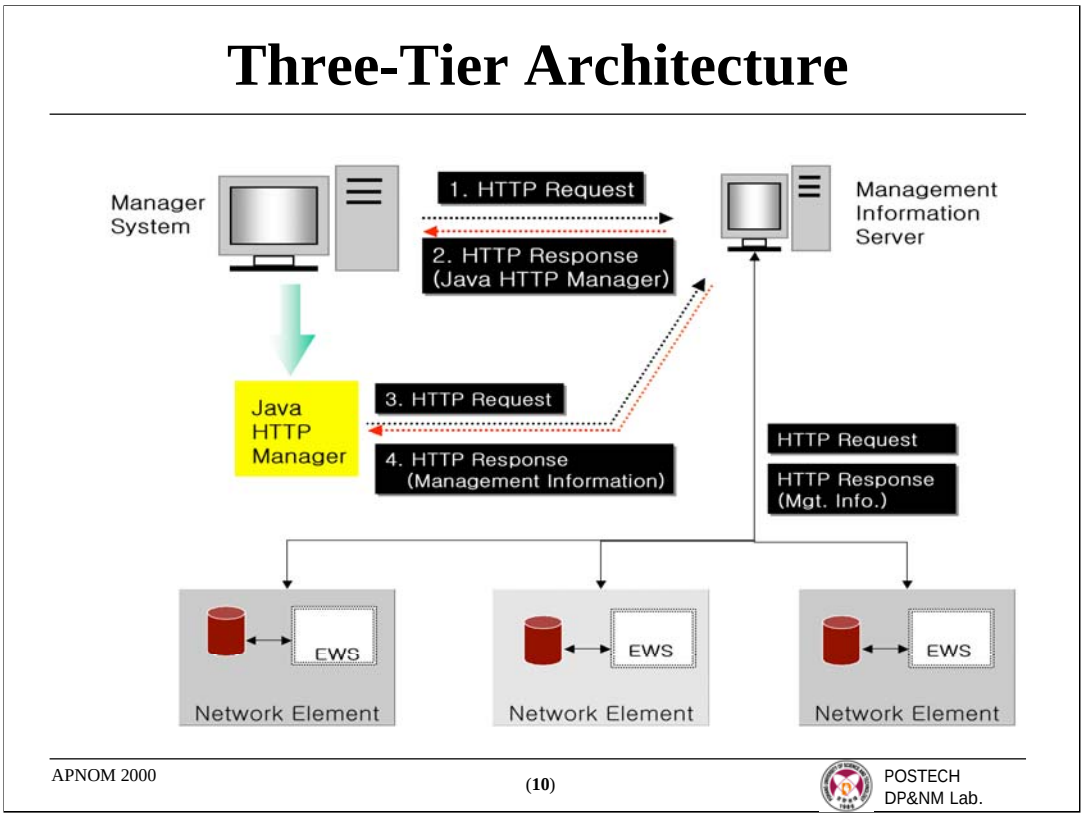
Java HTTP manager and management information server communicate using HTTP. By reusing ex isting communication protocol, developers can avoid adding new specific management protocols. HT TP 1.1 compliant Web Server supports persistent TCP connection [5]. Persistent connection allows mu ltiple requests to be pipelined on a single connection, with a mechanism to encode each response as a s eries of chunks, making it unnecessary to buffer the entire response before transmission. This mechani sm avoids overhead of frequently setting up and tearing down connections. HTTP is a TCP-based appl ication protocol, therefore it is more reliable than UDP-based application protocol such as SNMP.

In order to manage network resources using HTTP, a Java HTTP manager can specify managemen t operation with the name of managed resource. We define a mapping between URL and management operation with the name of managed resource. The mapped URL is compliant with the standard URL s yntax, therefore it can be handled by a conventional Web server and Web browser [11]. We define thr ee management operations: get, set, getnext. The format of mapped URL syntax is depicted below.

http://host/resource/management-operation?parameter

Management information is encoded into the HTTP data part in chunk of binary data of arbitrary s ize. Java HTTP manager request management operation to the management information server [5]. Th e management information server responds with an HTTP header followed by a MIME-typed propriet ary encoded data. This data can be compressed with gzip if the Web browser supports the decompressi on capability. The format of TCP payload of HTTP message transferring encoded management inform ation is depicted bellow.

HTTP Header	**MIME Type**	***gzip*'ed data**



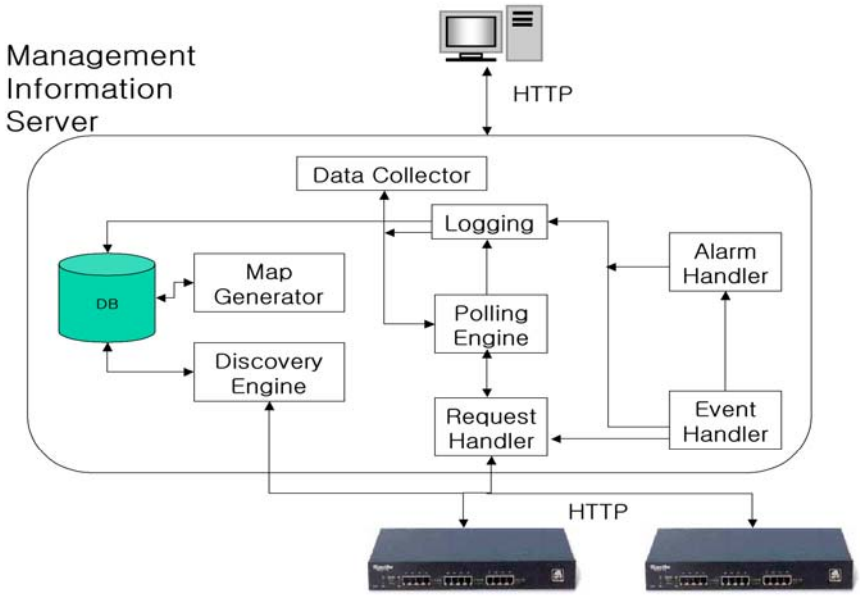
4.5 Three-tier architecture

We modified the two-tier architecture into a three-tier architecture by removing the management in formation server from the device and putting it into a stand-alone system. A two-tier architecture assu mes that the network devices have enough computing resources so that they can support most manage ment functions with the management information server. But this assumption is not always true and th e administrator may prefer a three-tier architecture to have a more stable manager system.

A module acting like a browser must be attached to the Management Information Server in order to request raw management information from the network element. The communication scheme between Management Information Server and network element is in the same way as two-tier architecture, as w ell as between Java HTTP manager and Management Information Server.

Management Information Server performs the predefined basic management tasks as usual using H TTP. When an administrator points his Web browser at Management Information Server's Web page, he can retrieve the aggregated or processed management information and controls a device through the provided Web page by the Management Information Server.

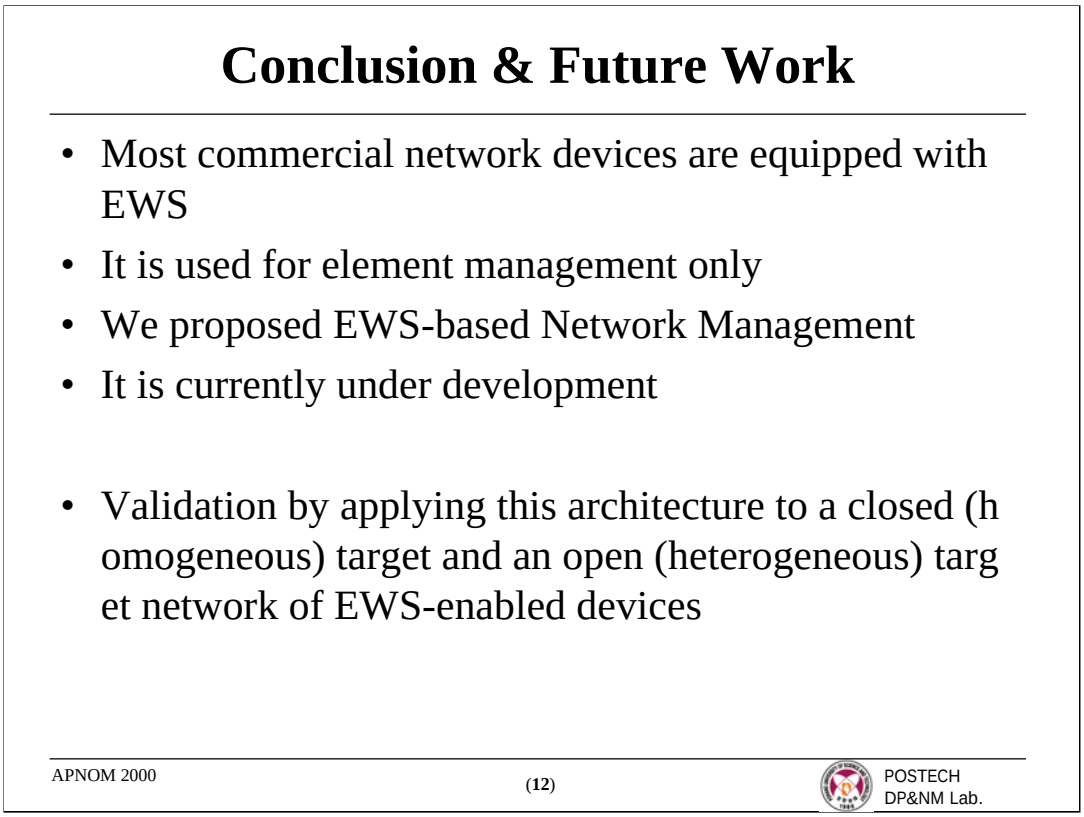
Management Information Server



4.6 Management Information Server

This slide shows the detailed design of the management information server. The Data Collector gat hers information data and performs threshold monitoring. This module requests polling data with polli ng duration and frequency to the Polling Engine. The Polling Engine schedules a series of HTTP reque sts based on requests from Data Collector. The Request Handler maintains communication channels fo r requests from management server to EWS agent. In order to manage multiple channels, this module acts asynchronously.

The Event Handler periodically polls each device to discover the existence of events and acts as a b uffer if necessary. This module logs polled events to the Data Base by use of a Logging module and ne xt forwards them to the Alarm Handler. The Alarm Handler executes scripts or command for notifying administrator upon receipt of an event and predefined action schedule. The Discovery Engine polls the EWS agent to initially discover EWS equipped device. The Map Generator manages and detects netw ork topology.



5. Conclusion and Future work

There are lots of benefits in applying Web technology to network management. Most commercial network devices are equipped with EWS, which are used for element management only. We have proposed an EWS-based network management architecture. We assume that the target network is composed of the same functional devices from different vendors or different devices from one vendor.

We have applied the thin-client and fat-server paradigm to the architecture and used Java technology, especially the Java applet. HTTP is used as transport protocol between management entities. On the basis of proposed architecture, we are currently implementing the management system. Our future work involves enhancing our architecture for application to an open (heterogeneous) target network.

References

[1] McCombie, B., “Embedded Web servers now and in the future,” Real-Time Magazine, no.1, Mar. 1998, pp. 82-83.

[2] Ian Agranat, “Embedded Web Servers in Network Devices,” Communication Systems Design, Mar. 1998, pp. 30-36.

[3] Ching-Wun Tsai and Ruay-Shiung Chang, “SNMP through WWW,” International Journal of Network Management, Vol 8. 1998, p. 104-119.

[4] M. J. Choi, et. al., “An Efficient and Lightweight Embedded Web Server for Web-based Network Element Management,” IEEE/IFIP Network Operations and Management Symposium, Hawaii, April 2000, pp.187-200.

[5] R. Fielding, et. al., “Hypertext Transfer Protocol - HTTP/1.1,” RFC 2616 IETF HTTP WG, June 1999.

[6] D. Raggett, et. al., “HTML 4.01 Specification,” IETF HTML WG, http://www.w3.org/TR/html401, Dec. 24 1999.

[7] J. Case, et. al., “The Simple Network Management Protocol (SNMP),” RFC 1157, May 1990.

[8] WBEM, “WBEM Initiative,” http://www.dmtf.org/wbem/.

[9] Sun, “Java Management Extensions (JMX),” http://java.sun.com/products/ JavaManagement/, Jun. 1999..

[10] Sun, Java Management Programmer’s Guide, Developer’s Release, June 1996.

[11] T. Berners-Lee, L. Masinter and M. McCahill, “Uniform Resource Locators (URL),” RFC1738, Dec. 1994.

[12] K. Arnold and J. Gosling, The Java Programming Language, Addison-Wesley, 1996.

[13] H. T. Ju and J. W. Hong, “POS-EWS Web Compiler for Supporting An Effective Application Interface,” PIRL-TR-99-003 Technical Report, POSTECH, Korea, Dec. 1999.

[14] XML, http://www.w3.org/XML

[15] DMTF, “Common Information Model (CIM) Specification version 2.2” June 14, 1999

[16] J. W. Hong, et. al., “Web-based Intranet Services and Network Management,” IEEE Communications Magazine, Vol. 35, No. 10, Oct. 1997, pp. 100-110.