An Embedded Web Server Architecture for XML-Based Network Management

Hong-Taek Ju, Mi-Jung Choi, Sehee Han, Yunjung Oh, Jeong-Hyuk Yoon,
Hyojin Lee and James W. Hong
Dept. of Computer Science and Engineering, POSTECH,
Pohang, Korea
{juht, mjchoi, sehee, bheart, infobank, really97, jwkhong}@postech.ac.kr

Abstract

Embedded Web servers are widely used today for IP-based element management. In this paper, we present a new management architecture that combines this technology with XML, DOM, and XPath to unify element management and network management. XML is used for both management information modeling and manager-agent communication. By taking advantage of modern Web technologies, the proposed architecture provides a method to develop management applications efficiently and to manage network devices effectively. We also explain how legacy SNMP agents are integrated into our proposed architecture.

Keywords

Web-Based Management, Network Management, XML, DOM, XPath, SNMP.

1. Introduction

A typical case of applying Web technology to network management is to embed a Web server into a network device for element management. This type of Web server is called an Embedded Web Server (EWS) [1]. An EWS provides users with a Web-based management interface constructed using HTML [2], graphics, and other features common to Web browsers. The status of a device is provided to the user by simply retrieving pages, and an operator command is sent back to the device using forms that the user completes. Accessing Web-based management user interfaces through embedded Web servers offers many advantages: ubiquity, user-friendliness, low development cost, and high maintainability.

However, Web-based element management has 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, which is typical for large networks, an administrator cannot afford controlling and configuring only one system at a time through a Web browser—although bookmarks, device lists, and the browser's back and forward buttons can help. This

scenario ignores the realities for large networks. This approach also tends to be device-centric and may not be able to provide logging, network topology, trend analysis, and other high-level management capabilities that are essential to manage networks.

The Simple Network Management Protocol (SNMP) [3] is the network management solution used by most of the industry since the early 1990s. It provides an interoperability framework to collect instrumentation-level measurement data to support fault and performance management. Compared to SNMP-based element management, Web-based element management significantly improves configuration management, remote diagnosis, and remote troubleshooting. Each management scheme presents advantages and disadvantages. Consequently, the two coexist in the real world.

In addition to the SNMP- and Web-based management stacks, many network devices also support a *telnet* server to enable operators to use the Command Line Interface (CLI) from a remote console. The issue of transferring CLI commands through XML/HTTP is relevant but outside the scope of this paper.

The concurrent support of SNMP- and Web-based element management poses a problem, because it requires more computing resources in each network device. If the same device supports two agents, one for SNMP [3] and another for HTTP [4], the management overhead can become too high. Also, it is difficult to guarantee consistency and access control because there are multiple access paths to the real resource being managed. If multiple processes can access and change variables pertaining to a real resource, a critical section problem can occur. To circumvent this problem, we can map SNMP requests onto HTML/HTTP requests within a device.

Since an embedded Web server normally provides management information in the form of HTML/Java, a natural language or display logic program, an operator can understand the meaning of management information by looking at the display of the browser. However, it is almost impossible for computer software to understand such information by interpreting HTML/Java. Therefore, management information disseminated by an embedded Web server in HTML/Java was not collected or processed by a Web-based manager application based on the manager-agent paradigm. Due to the absence of a centralized management capability in this case, management functions such as data logging, aggregation, and report generation were almost impossible. In this paper, we present a method that makes it possible to build centralized management applications based on the manager-agent paradigm.

The main motivation behind our new management architecture is to use embedded Web servers for network management as well as element management. A secondary motivation is to leverage Web interfaces. A unifying management protocol into HTTP at the network device can solve the dual stack problem. The second motivation is to apply Extensible Markup Language (XML) [5] for encoding management data into the HTTP payload. Such thinking is reasonable because XML is the data-encoding standard over the Web. The third motivation is to use XML Schema for specifying management information. XML is more than a

document markup language: it also helps model content and create standards for content.

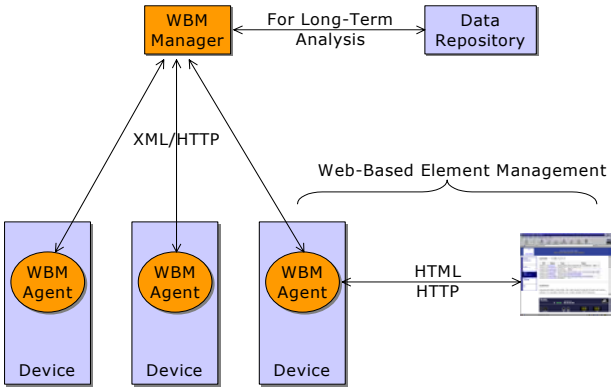

Figure 1. Web-Based Network Management Architecture

Our new management architecture is depicted in Figure 1. It consists of Web-Based Management agents (WBM agents) and a Web-Based Management manager (WBM manager) which follow the standard manager-agent paradigm. WBM agents exchange management information with the WBM manager over XML/HTTP. The WBM manager collects management information from multiple WBM agents and stores it in the information repository for long-term analysis. The WBM manager is an XML-based Web client application for network management, and the WBM agent is a specialized Web server for providing management information in XML form.

The remainder of this paper is organized as follows. Section 2 briefly describes XML [5] and work related to XML-based management. Section 3 describes our approach for XML-based network management. Section 4 discusses the implementation of our proposed architecture. Finally, Section 5 presents a summary and describes future work.

2. Background

In this section, we briefly introduce XML and present related work.

2.1. XML

The eXtensible Markup Language (XML) [5] makes it possible to define markup languages with an emphasis on specific tasks, such as electronic commerce, supply-chain integration, data management, and publishing. XML is rapidly becoming the strategic instrument for defining corporate data across a number of application domains. The properties of XML markup make it suitable for representing data, concepts, and contexts in an open, platform-, vendor-, and

language-neutral manner. XML Schema describes the data model of the XML document.

There are two fundamental approaches to defining XML documents: Document Type Definition (DTD) [6] and XML Schema [7]. Either approach is appropriate for most documents, yet each one offers unique benefits. DTD is in widespread use and has extensive tool support, while XML Schema is more powerful and provides advanced features such as open content models, namespace integration, and rich data typing. Because we value rich data typing, we decided to use the XML Schema approach to specify management information.

The Document Object Model (DOM) [8] specifies the means to access and manipulate XML documents. Without DOM, XML is nothing more than a storage system for data that can be accessed by various proprietary methods. With DOM, XML gets one step closer to a truly universal, cross-platform, application-independent programming language. DOM allows the reconstruction of the complete document from the input XML document, and gives access to any part of the document. Also, it supports standard methods for manipulating and modifying the original document. In relation to the WBM agent, we use DOM as a uniform access to management data and a central storage area for management data.

XPath [9] is an expression language used to address parts of an XML document. The syntax used by XPath is designed for use in URIs [10] and XML attribute values, and must be very concise. The name of the XPath is based on the idea of using a path notation to address an XML document. XPath operates under the assumption that a document has been parsed into a tree of nodes. We use XPath as the basis for addressing managed objects between the WBM manager and WBM agents.

The use of XML in network management presents many advantages [11,12], especially the following:

- better integration of management data from disparate sources;
- more flexible link between managed object and management application;
- tighter interoperability among management applications from different vendors;
- easy and powerful rendering and transformation of management information;
- automatic and centralized validation for management data.

2.2. Web-Based Management

2.2.1. WBEM

Web-Based Enterprise Management (WBEM) [13] is an industry initiative to develop a standard, non-proprietary means for accessing and sharing management information in an enterprise network. WBEM defines an information model called the Common Information Model (CIM) [14]. CIM is an object-oriented schema of managed objects. These managed objects are representations of real resources, and the schema provides a single data description mechanism for all types of resources.

WBEM provides an information standard that defines how data is represented, and a process standard that defines how the components interact.

Obviously, a method for accessing CIM data is required. The DMTF defined the CIM to XML mapping [15] and CIM operations over HTTP [16]. The specification of the CIM to XML mapping defines the XML Schema used to describe the CIM object in XML for encapsulation over HTTP. Both CIM classes and instances must be valid XML documents for this schema. The CIM operations over HTTP allow implementations of CIM to operate in an open, standard manner. It describes how CIM operations are encoded in the HTTP payload using XML, and defines the syntax and semantics of the operation requests and their corresponding responses. WBEM uses XML only for object and operation encoding.

2.2.2. WIMA

To date, the most sophisticated and comprehensive research work for Web-based management is Web-based Integrated Management Architecture (WIMA) [11] by Martin-Flatin. He identified the following problems in SNMP-based management:

- *Scalability and efficiency issues*: difficulties in ever more management data, too many messages for bulk transfer, latency in sparse table retrieval, verbose OID naming, no compression of management data, polling, and unreliable transport protocol.
- *Missing features*: low-level semantics of MIBs, poor protocol primitives, limited language to specify MIBs, and no support for methods.

Using Web technologies, WIMA solved most of the listed problems. WIMA proved that push-based network management is more appropriate than pull-based network management. A WIMA-based research prototype, JAva MAnagement Platform (JAMAP) [17], implemented push-based network management using Java technology. WIMA provided a way to exchange management information between a manager and an agent through HTTP. HTTP messages are structured with a MIME multipart. Each MIME part can be an XML document, a binary file, BER-encoded SNMP data, etc. By separating the communication and information models, WIMA allows management applications to transfer SNMP, CIM, or other management data. Martin-Flatin also demonstrated that XML is especially convenient for distributed and integrated management, for dealing with multiple information models, and for supporting high-level semantics.

In our management architecture, we also use push technologies, together with a translator that integrates legacy SNMP agents by performing SNMP-to-XML model-level mapping. As in WIMA, we have adopted XML/HTTP to exchange management information between an agent and the manager. However, we took a different approach to address managed objects. The dissimilarity of addressing methods comes from the difference in information modeling approaches. WIMA does not propose new information model approaches, but we use XML and DOM for management information modeling.

2.2.3. XNAMI

In 1999, John *et al.* proposed an XML-based architecture for SNMP management of network and application, called XNAMI [18]. The architecture permits a manager system to extend the agent's MIB within the SNMP framework. SNMP GET and SET operations on extension objects are implemented in Java. The XNAMI manager transfers the Java bytecode to the XNAMI agent using an SNMP SET operation. The XNAMI agent executes it upon receipt of SNMP GET to the extended object. Also, the XNAMI agent supports runtime representation of its MIB in XML. An XML document describing the MIB is transferred and retrieved by the SNMP operation. In this architecture, XML is used to represent the MIB definition and store it in a DOM Tree at the agent, and to browse the MIB module at the manager. XML is used for transferring and processing MIB definitions, as opposed to MIB values.

3. XML-Based Network Management

In this section, we describe the information, communication, and organization models of our new management architecture: XML-based Network Management (XNM).

3.1. Management Information Model

The management information model defines the modeling approach: entity-relationship diagrams, data types, object-oriented models, etc. It also defines a unique notation for describing the management information. We decided to use the XML Schema to express our management information model. The reasons for not relying on standard languages such as the Structure of Management Information (SMI), used by SNMP, or the Managed Object Format (MOF), used by CIM, are threefold:

- We must define new management information. Our development target is to enhance the Web-based element management so as to be comprehensible to the WBM manager. A significant amount of management information provided by the Web-based element interface is not yet defined in the standard information model, so we must define new management information.
- Information modeling using the XML Schema is a widespread approach in other application areas. XML permits the modeling of content, ranging from book-oriented document types to commercial transactions. XML is an enabling technology for business-to-business integration, data interchange, e-commerce, and the creation of application-specific vocabularies. Database developers are particularly interested in XML as a building block for creating middle-tier servers to integrate data from disparate databases.
- Compared with SNMP SMI and WBEM CIM, the XML Schema has many advantages—in addition to the advantages of using XML in network management, already listed in Section 2.1. The XML Schema is easy to learn. Developers can use many powerful and convenient XML editors for creating XML documents. Another advantage is that XML data is concise and easy to

read because there is no need for automatic translation. It is a prerequisite to translate from SMI, CIM, or UML into XML if the modeling and encoding techniques are different. Generally, translation results in verbose XML data and low readability. Another advantage is that XML can be utilized for various purposes, including validation. Using XML authoring tools, a developer can generate sample XML data, database schema, and other forms of data structures based on the XML Schema.

By adopting some simple conventions, the XML Schema can successfully model management information displayed in the Web browser through Web-based element management. By capturing data types, key relationships, and structure, developers can easily produce an XML Schema by using an XML editing tool. Note that the XML Schema in XNM corresponds to SMI in SNMP and GDMO in OSI management.

3.2. Communication Model

Management entails configuration, monitoring and control of potentially physically dispersed resources. An intrinsic part of this process is the exchange of management information. The communication model defines the concepts for this exchange of information between management applications. for the sake of readability, we limit our management application to one WBM manager and one WBM agent in this paper.

A communication model must deal with the specification of services and protocols for the exchange of management information. Also, it must define the syntax and semantics for the protocol data unit. With respect to service and protocols, our XNM architecture follows the original model of structuring data over HTTP without any extension. Further, it uses XML for management information encoding syntax. This means management data is transferred over the HTTP payload in the form of an XML document.

For notification delivery, the communication model must provide an asynchronous communication method. However, HTTP is a strictly request-response protocol from a client to a server. This means that the WBM agent cannot send an event message to the WBM manager asynchronously. In WIMA, the solution to this problem is to add an HTTP client to the agent and an HTTP server to the manager. XNM uses the same solution as WIMA.

Another important issue regarding the communication model is addressing managed objects. When a manager requests management information, it must specify a unique name of the managed object to be retrieved. XNM uses the XPath standard for addressing managed objects. This solution offers the following advantages. First, XPath is a standard Web technology for addressing parts of an XML document. It is already used in conjunction with eXtensible Stylesheet Language Transformations (XSLT) [19, 20] to identify pieces of an XML document targeted for transformations. XPath is also widely supported.

Second, the WBM manager can effectively query the managed objects of the WBM agent. XPath expressions are formed using element names, attributes, and

built-in functions. Given the hierarchical nature of XML, one of the simplest expressions is to follow the tree structure to point to a particular element. For example, if a WBM manager makes a request with the expression "/MgtConf/ManageList", the WBM agent extracts all <Manager> elements and sends them to the WBM manager, as in the example illustrated in Figure 2. Using the // operator, one can retrieve an element regardless of its position in the hierarchy. The character "*" is used as a wildcard. Also, the format //element_name[@attribute_name], is used to select an element based on its attribute. Using "//*[@Operation=ADD], the WBM manager can search for all elements with an attribute named "Operation" with the value of "ADD" regardless of its position. A WBM manager can use a number of functions with XPath expressions. A rich addressing mechanism can provide a manager with the ability to make an efficient query.

Figure 2 is a data exchange example between a WBM agent and a manager. In this example, Figure 2(a) illustrates a monitoring interaction whereby a WBM manager requests all registered "Managers" from a WBM agent. Figure 2(b) illustrates a control interaction whereby a WBM manager adds a "Manager" to the "ManagerList" in a WBM agent. Figure 2(c) illustrates a notification interaction whereby a WBM agent sends an asynchronous alarm to a WBM manager indicating that "FanStatus" changed to "Fail".

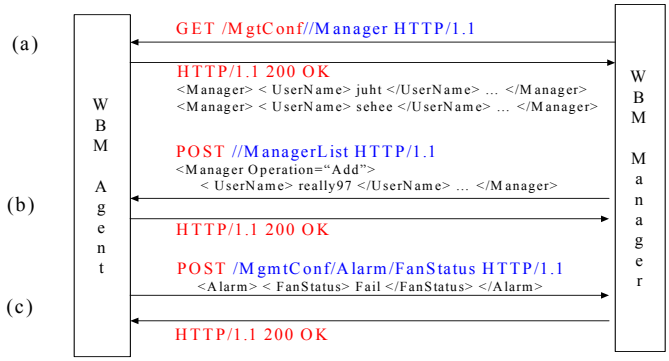

Figure 2. Interaction Examples Between WBM Manager and Agent

When managed objects are created, they must be named so that they can be addressed unambiguously. With respect to naming, XPath in XNM corresponds to Object IDentifier (OID) in SNMP and Distinguished Name (DN) in CMIP. A key difference between OID and XPath is that the latter supports filtering and scoping in object selection, which is sufficiently powerful that the logic of the manager application can be simplified and management traffic can be reduced.

3.3. Organization Model

The organization model of management architecture defines the actors, their roles, and the fundamental principles of their cooperation. The most well-known organization model in the network management area is the manager-agent

paradigm. There have been many proposals for new organization models, including management by delegation, policy-driven management, push-based network management, mobile agents, and intelligent agents.
In XNM, we use push-based network management. Push technologies automate information delivery based on the publish-subscribe paradigm [21]. Sometimes, they use true push (server-initiated communications); but usually, they use a sort of scheduled pull. There is a multitude of Web push applications: stock quotes, live game score updates, etc.

There are many methods for Web push from HTML refresh meta-tag [2] to Java applet and Channel Definition Format (CDF) [22]. However, applying these current incarnations of Web push to XNM has the following problems. First, they are inefficient in using bandwidth because they are not true push. Second, they rely on a display technique such as HTML and the Java applet.

We integrate push-based network management with XML as follows. The management information model of the WBM agent is automatically published in XNM because XML has a self-describing capability. Further, the WBM manager can discover the management information model by using DOM. With respect to the communication model and information model, distribution of management information and notification delivery is identical: asynchronous communication and no additional information model. Subscription is different with control, monitoring, and notification of the manager-agent paradigm. From the information model viewpoint, subscription information must be added to the original information specification. We define a new information specification for a subscription in a form of an XML Schema. The WBM manager sends subscription information which conforms to the new information specification. After receiving the subscription information, the WBM agent schedules a series of message distribution and sends it to the subscriber according to the schedule.

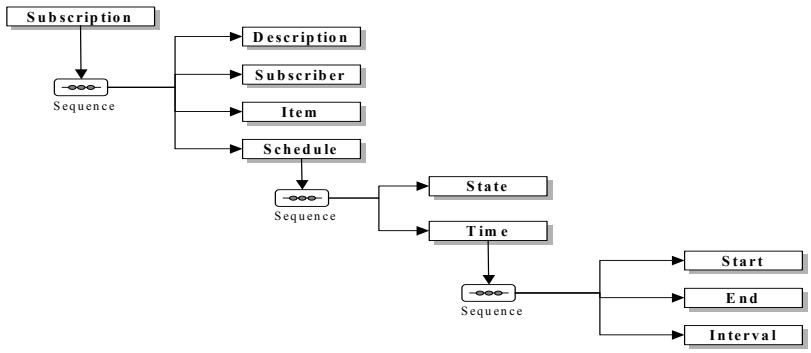

Figure 3. XML Schema for Push-based Subscription

An XML Schema for subscription and example communication is depicted in Figure 3. Subscription information includes a description of itself (Description),

subscriber's URL [23] for receipt (Subscriber), managed object's XPath expression (Item) and trigger information (Schedule).

4. Implementation of our XNM Architecture

In previous work, we had proposed an element management architecture whose core component is an embedded Web server [1]. To demonstrate its feasibility, we had developed an HTTP/1.1-compliant embedded Web server (called POS-EWS) that implemented this architecture. This time, we extended our software to support not only element management, but also network management. In this section, we describe this implementation of our XNM architecture.

4.1. WBM Agent

Figure 4 shows the structure of the WBM Agent. The components that we added to our old element management architecture are the DOM Tree, the XPath Handler, the Push Scheduler, and the HTTP Client Engine. These components are shaded on Figure 4.

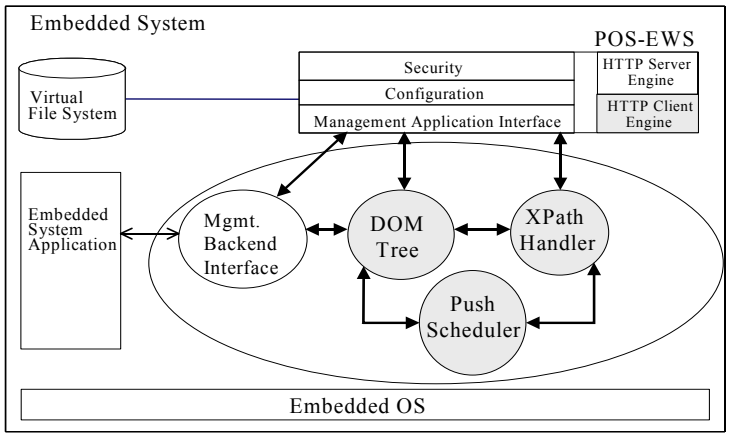

Figure 4. WBM Agent Architecture

The HTTP Client Engine sends asynchronous messages to the WBM Manager to report alarms and distribute management data according to the schedule. The XPath Handler selects a managed object in the DOM Tree, interpreting the XPath expression sent from the WBM Manager. A DOM Tree is a virtual repository of management data and provides a manipulation point for managed objects. The Push Scheduler manages subscription information and the schedule for message distribution, and sends the scheduled messages.

When the WBM Agent receives a request message, the Management Application Interface of POS-EWS selects specified nodes in the DOM Tree using the XPath Handler. For the selected nodes, the agent retrieves management data from the DOM Tree through the DOM interface, and sends the data to the WBM

Manager. In order to send up-to-date information, the DOM Tree updates the node value for the selected node through the Management Backend Interface before replying to the WBM Agent. This type of update of the DOM node value from real resources is called a *pull-based update*. The DOM Tree requests a value to the embedded system application through the Management Backend Interface and receives a response.

For certain nodes, it is not necessary to update the DOM node value before replying because this value is up-to-date. In this case, the Management Backend Interface module is responsible for the update. Periodically, or when events occur, the Management Backend Interface updates the DOM Tree node value. This type of update is called a *push-based update*. For frequently changing data, such as traffic measurement data, the pull-based update is more appropriate than the push-based update, and static or semi-dynamic information can benefit from using a push-based update rather than a pull-based one. In case of pull-based update, the DOM node is updated by replacing the text of the node value with the Processing Instruction (PI) node—a standard node type of DOM.

When the WBM Agent receives a control message, the Management Application Module follows the same procedure as in the case of a request message. The only difference is that it executes a registered handler instead of retrieving information from DOM. The Management Backend Interface can send a notification message by calling the registered handler at the subject node after a push-based update.

4.2. WBM Manager

Figure 5 illustrates the structure of a WBM Manager.

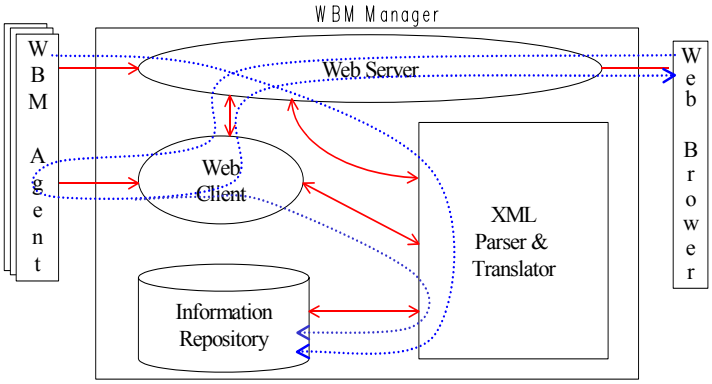

Figure 5. WBM Manager Architecture

The Web Server is used for providing operators with a Web interface and for receiving asynchronous messages from the WBM Agent over HTTP. Each function is implemented as a different URL. The Web Client exchanges

synchronous management information with the WBM Agent. The Information Repository is used for storing management information for long-term analysis.

The XML Parser & Translator module provides a basis to implement most management application functions because management information is represented in XML data. These functions include filtering, logging into the Information Repository, and collecting data from multiple WBM agents.

4.3. SNMP Integration

SNMP is currently the most widely used solution for managing network devices in the Internet. Its simplicity enables it to be implemented on small platforms with ease. To date, most network devices are equipped with an SNMP agent. Thanks to the integration of SNMP with XNM, the advantages of XNM are preserved without losing the SNMP functionality.

We regard the SNMP agent as a special form of the Management Backend Interface module in Figure 4, because an SNMP agent serves a WBM Agent as a real resource interface within a network device. Management information emitted by the SNMP agent is defined in an SNMP MIB and retrieved via the SNMP communication protocol. However, the WBM Agent relies on XML for processing management information. Therefore, we need an SNMP-to-XML management gateway. To develop such a gateway, both specification translation and interaction translation are required. For specification translation, we designed a MIB-to-XML translation algorithm. We implemented this algorithm in a MIB-to-XML translator that produces an XML Schema for creating a DOM Tree in the WBM Agent [24].

For interaction translation, we defined a mapping between each SNMP operation and the DOM API. Each DOM node represents a node in the SNMP agent MIB tree and has the ability to issue an SNMP message in order to update its node value. When the WBM Manager requests management information, each node updates its value and replies to the request with this new value. For example, when the WBM Manager sends a message with "GET mib-2/system/interfaces/ifTable/ifEntry/ ifType[oid=1]", the DOM node, which is selected by the XPath handler, issues an SNMP GET to the local host with the community string and OID value that was saved for this node. After updating the node value via SNMP, the WBM Agent replies with "HTTP/1.1 200 OK <mib-2><system>…<ifType oid=1>24</ifType>…</mib-2>".

5. Conclusion

In this paper, we have extended POS-EWS from element management to network management by using standard Web technologies. We leverage XML Schema for specifying management information and use XPath for the addressing scheme. Management information is encoded in XML and transferred over HTTP. We use DOM to unify the representation of and the access to management data in a WBM agent. Thanks to an effective use of modern Web technologies, we have maximized the advantages of using XML in network management.

We are currently developing a global management system for a commercial Linux server (Netsech EnterFlex series [25]) based on our proposed architecture. We are also looking to integrate other organization models such as policy-driven management, management by delegation, mobile agents, and intelligent agents into our XNM architecture.

References

[1]. H. T. Ju, M. J. Choi, and J. W. Hong, "EWS-Based Management Application Interface and Integration Mechanisms for Web-Based Element Management", *Journal of Network and Systems Management*, Vol. 9, No. 1, pp. 31-50, 2001.

[2]. D. Raggett, A. Le Hors, and I. Jacobs, "HTML 4.01 Specification", IETF HTML WG, http://www.w3.org/TR/html401, Dec. 1999.

[3]. J. Case, M. Fedor, M. Schoffstall, and C. Davin, RFC 1157, "The Simple Network Management Protocol (SNMP)", IETF, May 1990.

[4]. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee (Eds.), RFC 2616, "Hypertext Transfer Protocol -- HTTP/1.1", IETF, Jun. 1999.

[5]. W3C, "Extensible Markup Language (XML)", http://www.w3.org/XML.

[6]. T. Bray, J. Paoli, and C. M. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0", W3C REC-xml-19980210, Feb. 1998.

[7]. W3C, "XML Schema Part 0,1,2", W3C REC-xmlschema, May 2001.

[8]. W3C, "Document Object Model (DOM) Level 1 Specification," W3C Recommendation, Oct. 1998.

[9]. W3C, "XML Path Language (XPath) Version 1.0", W3C Recommendation, Nov. 1999.

[10]. T. Berners-Lee, R. Fielding, and L. Masinter, RFC 2396, "Uniform Resource Identifiers (URI): Generic Syntax", IETF, Aug. 1998.

[11]. J. P. Martin-Flatin, "Web-Based Management of IP Networks and Systems", Ph.D. Thesis, EPFL, Lausanne, Switzerland, Oct. 2000.

[12]. DMTF, "XML As a Representation for Management Information - A White Paper. Version 1.0", Sep. 1998.

[13]. WBEM, "WBEM Initiative", http://www.dmtf.org/wbem/.

[14]. DMTF, "Common Information Model (CIM)", http://www.dmtf.org/standards/standard_cim.php

[15]. DMTF, "Specification for the Representation of CIM in XML Version 2.0", DMTF Specification, Jul. 1999.

[16]. DMTF, "Specification for CIM Operations over HTTP Version 1.0", DMTF Specification, Aug. 1999.

[17]. J. P. Martin-Flatin, L. Bovet, and J. P. Hubaux. "JAMAP: a Web-Based Management Platform for IP Networks", Proc. 10th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM'99), LNCS 1700:164-178, Zurich, Switzerland, Oct. 1999.

[18]. A. John, K. Vanderveen, and B. Sugla, "An XML-based Framework for Dynamic SNMP MIB Extension", IFIP/IEE International Workshop on

Distributed Systems Operations and Management (DSOM), pp 107-120, Zurich, Oct. 1999.
[19]. W3C, “Extensible Stylesheet Language (XSL) Version 1.0”, W3 Consortium Candidate Recommendation, Nov. 2000.
[20]. W3C, “XSL Transformations Version 1.0”, W3 Consortium Recommendation, Nov. 1999.
[21]. J. P. Martin-Flatin. “Push vs. Pull in Web-Based Network Management”, Proc. 6th IFIP/IEEE International Symposium on Integrated Network Management (IM'99), pp. 3-18, Boston, MA, USA, May 1999.
[22]. C. Ellerman, “Channel Definition Format (CDF)”, W3 Note, Mar. 1997.
[23]. T. Berners-Lee, L. Masinter, and M. McCahill, RFC 1738, “Uniform Resource Locators (URL),” IETF, Dec. 1994.
[24]. J. H. Yoon, “Design and Implementation of SNMP Gateway for XML-based Network Management”, M.S. Thesis, POSTECH, Pohang, Korea, Dec. 2001.
[25]. Netstech Inc., http://www.netstech.com