Performance Evaluation of Interaction Translation between XML-based Manager and XML/SNMP Gateway

Mi-Jung Choi¹, So-Jung Lee¹, Sun-Mi Yoo¹, Hong-Taek Ju² and James W. Hong¹

¹Dept. of Computer Science and Engineering, POSTECH, Korea
²Dept. of Computer Engineering, Keimyung University, Korea

¹{mjchoi, annie, sunny81, jwkhong}@postech.ac.kr, ²juht@kmu.ac.kr

Abstract

Recently, there has been increasing research on the application of XML technologies to network and systems management. Researchers and developers have actively studied monitoring existing SNMP agents through an XML-based manager, and an XML/SNMP gateway approach. They have also investigated the interaction translation methods of the gateway between the XML-based manager and SNMP agents. However, the performance of the XML-based manager and XML/SNMP gateway has not been measured although it has been identified as an important issue. In this paper, we provide a performance evaluation of the XML-based manager and XML/SNMP gateway according to interaction translation methods; one HTTP-based, and the other SOAP-based. Our performance evaluation is based on our XML-based manager controlling network devices equipped with SNMP agents through our XML/SNMP gateway. We show the performance evaluation from the aspects of network traffic, response time, and computing resources, namely CPU and memory usage.

Keywords: Performance Evaluation, XML, SNMP, XML/SNMP Gateway, XML-based Manager, Interaction Translation

1. Introduction

There has been significant progress in the research of the use of XML [1] in network management. The use of XML in network management offers many advantages [2]. XML provides powerful modeling features for hierarchical management information in network management. XML-based network management (XNM) allows for the reliable transfer of a large amount of data. It can also be easily implemented using standard APIs and freely available software. This extensive applicability of XML technologies to network management alleviates many of the known problems of traditional SNMP-based management [3]. Much of the research on XNM has focused on the area of accommodating widely deployed SNMP devices. Specifically the specification and interaction translation for gateways between XML-based managers and SNMP-based devices.

Earlier XML/SNMP gateways [5, 6] applied an HTTP-based interaction translation between the XML-based

Numerous studies [9, 12, 13] have reported performance evaluation results comparing SNMP with HTTP or Web Services. However, performance evaluation of the interaction translation methods between the XML-based manager and XML/SNMP gateway has not yet been conducted. In our previous paper [8], we proposed three interaction methods between the manager and gateway: DOM-based, HTTP-based and SOAP-based. Today, the research of XML-based network management focuses on using Web Services based on SOAP/HTTP communication method. SOAP is useful in defining a XML-based interface and is easily implemented using freely-available source code. However, testing data to prove that SOAP is lightweight and efficient compared to HTTP is not available.

In this paper, we measure the performance of the HTTP-based and SOAP-based interaction translation methods of the gateway. We compare the performance evaluation results to determine more efficient interaction translation method between the XML-based manager and XML/SNMP gateway. The measured performance metrics between our manager and gateway are message size, response time, CPU usage and memory usage. We do not consider the DOM-based translation method since this interaction method is used in an internal type of gateway and is used in the other two interaction methods. We merely consider the HTTP-based and SOAP-based translations. Also, a performance comparison of XPath expressions is also necessary. The complexity of XPath expression affects the processing time of the gateway and manager. The complex XPath expressions generate more processing overhead but create little network traffic by reducing queried result data through XPath condition. We will also investigate appropriate XPath expression to reduce network traffic and generate a little processing time.

The organization of this paper is as follows. Section 2 explains the testing environment of our performance evaluation in the POSTECH campus and implementation details for understanding performance issues. Section 3 describes the measurement metrics and methods. In Section 4, we present the performance evaluation and analysis results in accordance to the interaction methods between the XML-based manager and XML/SNMP gateway. In Section 5, we compare our work with other recently reported performance related work. Finally, we conclude this paper and discuss possible directions for future research in Section 6.
2. Measurement Environment and Implementations

As depicted in Figure 1, we applied our XML-based manager and two types of XML/SNMP gateways to monitor in/out traffic octets of network devices such as routers and switches deployed in the POSTECH campus network. Our XML-based manager and XML/SNMP gateway are connected by a 100Mbps LAN. However, XML/SNMP gateways and SNMP agents are connected by not only a 100Mbps LAN but also a gigabit Ethernet backbone network. Both XML-based manager and the XML/SNMP gateways run on Linux servers with Pentium-III 800 MHz CPU and 256M RAM. SNMP agents are running in network devices deployed in the POSTECH campus network. POSTECH has a gigabit Ethernet backbone network, which is composed of eleven gigabit backbone switches (Catalyst 6500/5500 series) and hundreds of 100 Mbps client switching hubs that are deployed inside the buildings. Two Cisco 7513 and 7401 Internet routers are connected to the backbone network. We selected a dozens of devices among these Cisco backbone routers and switches that are equipped with SNMP agents and carried out performance tests.

![Figure 1. Performance Testbed](image)

We implemented two types of XML-based managers and two types of XML/SNMP gateways according to interaction translation methods: HTTP-based, SOAP-based. The implementation details of our manager and gateway are illustrated in our previous paper [7, 14]. A brief description of our servers implementation is necessary to understand the performance evaluations. The XML-based manager has been implemented on a Linux OS using Java language. We used Apache Tomcat 4.0 for the Web server and Servlet engine. Most of the XML packages used in the XML-based manager are Java-based and are part of the Apache project [15]. We used Xerces 1.4.4 [16] as a DOM parser, Xalan 2.4.0 [17] as an XPath/XSLT processor. In addition, we used Xindice [18], a native XML DB, to store the XML document of management data. Innovation’s Java-based HTTP Client V0.3-3 [19] was employed as our HTTP Client. Zlib [20] Java library was used to reduce network traffic by compressing the request/response messages.

The servers hosting the gateways were configured almost the same as the manager. Our gateways also used
Tomcat, Xerces, and Xalan as the manager. In addition, the gateways used OpenNMS’s joeSNMP 0.2.6 [21] for the SNMP handler and trap handler. In the HTTP-based translation method, we used Innovation’s HTTP Client. In the SOAP-based translation method, we used Apache Axis 1.0 [22] for the SOAP engine, which is SOAP 1.1 and WSDL 1.1 compliant.

Table 1 shows messages between the manager and gateways according to the two interaction translation methods. These messages retrieve incoming traffic volumes (inOctets) in interface group of MIB II from two SNMP agents embedded with a gigabit backbone switches (6500/5500 series). The HTTP request message includes a <XQuery> tag for indicating get message and a <XPath> tag for addressing a specific part of XML document to retrieve. The SOAP request/response messages are almost the same as the HTTP messages without the SOAP header. As one more SNMP agent is added to the retrieved agent lists, the added message includes device information such as device IP and gateway information such as gateway IP, community name, SNMP agent version, and MIB name in the <XQuery> tag.

### (a) HTTP-based Message Example

**Request**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:getRequest
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns1="Gateway">
      <XQuery xsi:type="xsd:string">
        <XQuery>
          //ifInOctets
        </XQuery>
      </XQuery>
    </ns1:getRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

**Response**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:getResponse
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns1="Gateway">
      <ns1:getReturn xsi:type="xsd:string">
        <instances>
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.1" position="0">1480</ifInOctets>
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.2" position="1">2518480530</ifInOctets>
          ...
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.25" position="24">12323345</ifInOctets>
        </instances>
      </ns1:getReturn>
    </ns1:getResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

### (b) SOAP-based Message Example

**Request**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:get
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns1="Gateway">
      <XQuery xsi:type="xsd:string">
        <XQuery>
          //ifInOctets
        </XQuery>
      </XQuery>
    </ns1:get>
  </soapenv:Body>
</soapenv:Envelope>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
  xmlns:xsd=http://www.w3.org/2001/XMLSchema"
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance>
  <soapenv:Body>
    <ns1:getResponse
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns1="Gateway">
      <ns1:getReturn xsi:type="xsd:string">
        <instances>
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.1" position="0">1480</ifInOctets>
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.2" position="1">2518480530</ifInOctets>
          ...
          <ifInOctets oid="1.3.6.1.2.1.2.2.1.10.25" position="24">12323345</ifInOctets>
        </instances>
      </ns1:getReturn>
    </ns1:getResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

**Table 1. Request/Response Message Examples: HTTP vs. SOAP**

<table>
<thead>
<tr>
<th>Method</th>
<th>Request/Response Message Example: HTTP vs. SOAP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HTTP-based</strong></td>
<td><img src="image1" alt="HTTP-based Message Example" /></td>
</tr>
<tr>
<td><strong>SOAP-based</strong></td>
<td><img src="image2" alt="SOAP-based Message Example" /></td>
</tr>
</tbody>
</table>

### 3. Measurements Metrics and Methods

We monitored incoming traffic octets (inOctets) of 100Mbps switching hubs in POSTECH campus network. The number of interfaces for each switch is 25. We therefore increased the number of MIB objects 1 to 250 by adding one more agent up to 10 devices.

First, we measured the network traffic volumes at Ethernet frame layer between our XML-based manager and XML/SNMP gateway according to the interaction translation methods. The messages are the sum of the request and response messages. The XML-based manager sends an HTTP or SOAP request message to the gateway to retrieve...
management information from multiple SNMP agents. The XML/SNMP gateway processes the request message and sends multiple SNMP request messages to each SNMP agent. We merely monitored the network traffic between the manager and gateway. The request/response message formats in relation to the interaction translation methods is shown in Table 1. We captured the network packet using network monitoring tool, Ethereal [23].

Next, we measured the processing times as a function of the number of SNMP agents, namely the number of MIB objects. Also, we separately measured the processing time in the XML-based manager and the processing time in the XML/SNMP gateway. The processing time in the manager includes generating a request message, DB processing, and XSLT processing to present in the Web. The processing time in the gateway includes perceiving the request message and translating the message from HTTP or SOAP format into SNMP. Also, we measured the processing overhead and network traffic reduction after compressing the message between the manager and gateway. That is, we measured how much the network traffic reduces and how long the processing time increases by compressing the request/response messages. We measured processing time using `System.currentTimeMillis()` in JDK 1.4.1.

We measured the processing time of the manager and gateway and network traffic volume between the manager and gateway in relation to the XPath expression. Various XPath expressions can be applied in the gateway. The complexity of XPath expressions impacts the network traffic volume and processing time.

Finally, we measured the processing overhead of the XML-based manager and XML/SNMP gateway in terms of CPU load and run-time memory usage. We measured CPU usage by using the ‘top’ command and memory usage by investigating ‘status’ and ‘statm’ file included in the directory ‘/proc’ and processor ID (pid) number. We divided memory usage into 2 categories: program size (static memory usage) and data size (virtual memory usage). The unit of CPU usage is percentage and that of memory usage is KB. To compare the processing time of the gateway based on the change in the computing resources, we upgraded the CPU and memory of the gateway.

The traffic pattern of POSTECH campus network is constant; there is always a large volume of traffic except for the period from 4 a.m. to 8 a.m. Since we were not sure of the effect of network traffic volume to the measurement of latency, we measured the latency at different times and calculated an average and standard deviation from measured values. Also, for statistical significance, we conducted 10 measurements for each test and averaged them.

4. Performance Evaluation

As explained in Section 3, we measured network traffic volume, processing time, computing resource usages, etc. In this section, we show the performance evaluation results in each test.

4.1. Network Traffic

First, we evaluated the network traffic volume at the Ethernet frame layer. Table 1 shows the network traffic
between the XML-based manager and XML/SNMP gateway in relation to the number of MIB objects. We retrieve inOctets objects in the ifTable of MIB-II. Each network device has 25 network interfaces, so the number of MIB objects (inOctets) from one agent is 25.

The manager sends an HTTP request message then the gateway processes the request message and sends multiple SNMP request messages to each SNMP agent [7, 13]. The network traffic between the manager and gateway is the HTTP message including TCP control information and connection setup. Figure 2 shows an HTTP message example of the interaction messages and size between the manager and gateway retrieving ‘/ifNumber’ from one SNMP agent. The HTTP Get request message is 642 bytes and the response message including ‘HTTP1.1/200 OK’ message is 417 bytes. As shown in Figure 2, the TCP connection is set-up using three-way handshaking. The sum of TCP SYN, TCP SYN/ACK and TCP ACK is 178 bytes, and this TCP connection overhead is added to every HTTP or SOAP messages in the request connection period. If the HTTP response continuation message at IP layer is longer than 1500 bytes, the messages is divided into multiple messages. Each message size is 1500 bytes because the maximum transmission unit (MTU) size of an IP packet is 1500 bytes.

Figure 2. HTTP Message Example between Manager and Gateway
Figure 3. Network Traffic of Get Operation for inOctets in ifTable

Figure 3 is the graph showing the network traffic between the manager and gateway. The solid line with squares shows the request/response network traffic from the XML-based manager to the gateway through HTTP-based interaction translation. The solid line with triangles is the request/response traffic through SOAP-based interaction translation. As one more agents is added, the increased number of MIB objects is 25. The network traffic of the request and response message from the manager to gateway linearly increases as one more SNMP agent is added. Also, the traffic volume of SOAP messages is added 800–950 bytes into HTTP messages. These added bytes are due to the SOAP header overhead in the request and response message.

4.2. Processing Time

Next, we measure the processing overhead of the manager and gateway to each request and responses. Figure 4 shows the average processing time of the Get operation for the inOctets objects of ifTable in relation to the number of MIB objects. The solid line with squares shows the processing time of the XML-based manager using HTTP-based interaction translation. The dotted line with squares shows the processing time of the XML/SNMP gateway using HTTP-based interaction translation. As one more SNMP agent is added, the processing time of the manager increases 100–150 ms and the processing time of the gateway increases 30–50 ms in the case of an HTTP-based interaction translation method. As shown in Figure 4, the processing time of the HTTP-based translation is longer in the manager than in the gateway by about 200–600 ms. As the number of MIB objects
increases, the processing overhead of the manager increases slightly more than the overhead of the gateway.

In the case of the SOAP-based translation method, the difference of processing time between the manager and gateway is slighter than that of the HTTP-based. Also, the processing time of the gateway in the SOAP-based is 2 times longer than that in the HTTP-based. The SOAP-based gateway needs more processing time to perform an interaction translation from SOAP request message to SNMP request message. That is, the SOAP-based interaction translation between the manager and gateway generates a little more network traffic volume, however, the processing the same request message in the gateway consumes more time than the HTTP-based interaction translation. The standard deviation of the processing time of the manager is about 50~100 ms and that of the gateway is 100~1500 ms in both the HTTP-based and SOAP-based interaction translation methods.

![Figure 4. Processing Time of Get Operation for inOctets using HTTP and SOAP-based Translation Methods](image)

4.3. Message Size vs. Processing Overhead by Compression

We compressed the HTTP and SOAP messages at the application layer in both interaction methods. The pattern of request and response message in both interaction translation methods remain almost the same when one more SNMP agent is added. Therefore, as shown in Figure 5, the compressed message increases slightly although the uncompressed origin message rapidly increases. The SOAP message is based on an HTTP message, but it is slightly larger due to the SOAP header. As shown in Figure 3 and Figure 5, the HTTP message of the SOAP-based interaction translation is a little longer about 800 bytes than that of HTTP-based interaction translation. However, the compressed message size of both interaction methods is little difference about 50~110 bytes. The processing
overhead of compressing and decompressing the request/response messages takes about 300 ms. Therefore, the processing time of the manager and gateway both increase about 300 ms when compression and decompression is applied. However, the transmission time gained after compressing the message is about 100 ms. Therefore, the network traffic overhead sharply decreases by applying a compression mechanism [20].

4.4. Computing Resources Usage

The computing resources of the manager and gateway we measured include CPU and memory usage. As mentioned in Section 2, the manager and gateway both run on Linux servers with Pentium-III 800MHz CPU and 256MB RAM. Figure 5 shows the average CPU usage of the manager and gateway. The solid line with squares shows the CPU usage of the manager using HTTP-based interaction translation method; the solid line with triangles shows the CPU usage using SOAP-based method. The SOAP-based interaction translation needs more CPU usage because it processes the SOAP message added to the HTTP message. The dotted lines illustrate the CPU usage of the gateway according to the interaction methods. The CPU usage of the manager and gateway increases about 2–4% as one SNMP agent is added. The standard deviation is about 2–5% in the case of the manager and about 3–8% in the case of the gateway as the number of SNMP agent increases. The CPU usage of the manager is about 4–6% higher than that of the gateway in both HTTP-based and SOAP-based interaction translation methods. This is
a result of the manager needing CPU processing time for the Apache Tomcat Web server for Web-based user interface and Xindice DB server.

![Graph showing CPU usage of Manager and Gateway](image)

**Figure 6. CPU Usage of Manager and Gateway According to Interaction Translation Methods**

Next, we measured the memory usage of the manager and gateway in accordance to executable program size and virtual memory size. Table 2 shows the measured results of the memory usage using the HTTP-based interaction translation method. As shown in Table 2, the program size is much smaller than the running memory usage in both the manager and gateway. The gateway requires more memory usage in the perspective of program code and virtual memory. As shown above, the manager demands higher CPU usage than the gateway. However, in the memory usage, the reverse is true since the gateway requires more memory usage to process DOM operation for the specification and interaction translations. We think the process of the gateway is concentrated in memory usage and the process of the manager in CPU usage. However, the difference in both cases of CPU and memory usages is small. The memory usage of the manager using SOAP-based interaction translation is much bigger about 500 Kbytes in the program code memory and 800 Kbytes in the virtual memory. The standard deviation of the program code memory in the manager is about 50Kbytes and that in the gateway is about 200–300 Kbytes in the case of the HTTP-based interaction translation method. The standard deviation of the virtual memory in the manager is about 200–500Kbytes and that in the gateway is about 500–900 Kbytes in the case of the SOAP-based interaction translation method.
4.5. Processing Time and Message Size According to XPath Expression

Finally, we evaluated the message size and processing time in relation to the different XPath expressions used by both gateways. If the manager monitors incoming and outgoing traffic volumes which the operation status of the interface is ‘up’ to perform a long term analysis, the manager can only retrieve in/out octets at a time and then extract in/out octets which the operation status is up. Or the manager can retrieve in octets and out octets separately and then extract the satisfactory in and out octets. Or the manager can retrieve the necessary in/out octets that satisfy the condition, that is the operation status is up. This retrieval condition is categorized by using XPath expressions as shown in Table 3. The processing time of the manager and gateway is very dependent on XPath expressions. If the gateway performs simple XPath expression, the manager must carry complex XPath expression to extract the necessary data, The network device (Internet router) has 15 network interfaces. The number of currently operating interfaces is 5. The XPath expression of ‘//ifInOctets’ is the abbreviation format of the ‘/RFC1213-MIB/internet/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInOctets’. In the cases of ‘//ifInOctets’ and ‘//ifOutOctets’, the processing time of the manager and gateway and the message size are almost the same. If the manager retrieves incoming and outgoing traffics simultaneously, it is efficient both in the perspectives of processing time and network traffic volume to use ‘//ifInOctets | //ifOutOctets’ rather than ‘//ifInOctets’ and ‘//ifOutOctets’ separately. In the case of ‘//ifOperStatus[.='1’]/following-sibling::ifInOctets | //ifOperStatus[.='1’]/following-sibling::ifOutOctets’, the processing time of the gateway is about 6 times as longer than the expression of ‘//ifInOctets | ifOutOctets’. However, the transferred network traffic of the fifth expression reduces to about half of the fourth expression. The fifth expression includes various XPath semantics [11] such as axes expression, node tests and abbreviated syntax which require additional processing time.

If the manager request complex XPath expression including a specific condition, the transferred message size can be reduced. However, the processing time of the gateway remarkably increases. The manager can take a burden of the gateway by performing complex expression in the manager not in the gateway. If the manager request simple XPath expression, the extracting process of the necessary Therefore, using the appropriate XPath expression is essential to retrieve the necessary information from SNMP agents considering the processing time and message size.

Table 2. Memory Usage of XML-based Manager and Gateway in HTTP-based Translation

<table>
<thead>
<tr>
<th>Inter. Type</th>
<th>System</th>
<th>Memory Type</th>
<th>1 (25)</th>
<th>2 (50)</th>
<th>3 (75)</th>
<th>4 (100)</th>
<th>5 (125)</th>
<th>6 (150)</th>
<th>7 (175)</th>
<th>8 (200)</th>
<th>9 (225)</th>
<th>10 (250)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP-based</td>
<td>Manager</td>
<td>Program</td>
<td>6843</td>
<td>7030.8</td>
<td>7048</td>
<td>7080</td>
<td>7118.5</td>
<td>7131.4</td>
<td>7135.8</td>
<td>7138.6</td>
<td>7172.8</td>
<td>7215.4</td>
</tr>
<tr>
<td></td>
<td>Virtual</td>
<td>479140</td>
<td>479342</td>
<td>479412</td>
<td>479522</td>
<td>479582</td>
<td>479672</td>
<td>480056</td>
<td>481472</td>
<td>487597</td>
<td>490762</td>
<td></td>
</tr>
<tr>
<td>Gateway</td>
<td>Program</td>
<td>12346</td>
<td>12568</td>
<td>13000</td>
<td>13023</td>
<td>13112</td>
<td>13160</td>
<td>13194</td>
<td>13397</td>
<td>13481</td>
<td>13494</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Virtual</td>
<td>480977</td>
<td>489002</td>
<td>489302</td>
<td>489382</td>
<td>489552</td>
<td>489592</td>
<td>489632</td>
<td>489672</td>
<td>489782</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Unit: KB
5. Related Work

Ricardo Neisse and Lisando Granville [9] performed a study on the performance of SNMP and Web services. They implemented both a SNMP gateway and Web services gateway. These gateways support two approaches for gateway translation: protocol level and object level. First, the protocol-level gateways provide the mapping of SNMP primitives to Web services operations (e.g. Get, GetNext and Set). Object-level gateways map MIB structures to Web operations (e.g. GetIfTable). They also measured the bandwidth consumption for two gateway approaches. They use SOAP messages over HTTP, HTTPS, and compressed messages through the ZLIB algorithm. For protocol level gateways, Web services are always required.

George Pavlou et al. [12] also performed a study on the performance of Web services and SNMP. In fact their study is much broader than just performance evaluation, and includes CORBA-based approaches. Instead of using a gateway, they implemented a Web services agent, using the WASP toolkit. As example, they study the retrieval of TCP MIB variables, and measure bandwidth and round trip delay. The performance of this agent was compared to a Net-SNMP agent. However, they did not investigate other SNMP agents nor the effect of fetching the actual management data from within the system. They said that Web services is a promising technology but, being XML-based, it has more overhead than SNMP and CORBA.

Another study on the performance of SNMP and Web services are done T. Drevers and A. Pras [13]. They compared the performance of Web services based monitoring to that of SNMP in the aspect of bandwidth usage, CPU time, memory requirements and round trip delay. For the study, they implemented several prototypes of Web services based agents which could retrieve single ‘ifTable’ elements, ‘ifTable’ rows, ‘ifTable’ columns or the entire ‘ifTable’. The performance evaluation showed that SNMP is more efficient in cases where only a single object should be retrieved, however for larger number of objects Web services may be more efficient. They concluded that if performance was the issue, the choice between SNMP or Web services is not the main factor that determines the performance.

These related works all compare the performance between SNMP and Web Services in the aspects of network

<table>
<thead>
<tr>
<th>XPath Expression</th>
<th># of MIB Objects</th>
<th>Message Size (Bytes)</th>
<th>Processing Time of Manager (ms)</th>
<th>Processing Time of Gateway (ms)</th>
<th>Message Size (Bytes)</th>
<th>Processing Time of Manager (ms)</th>
<th>Processing Time of Gateway (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC1213-MIB/internet/mgmt/mib-2/interfaces/ifTable/ifEntry/ifInOctets</td>
<td>15</td>
<td>2522</td>
<td>1606</td>
<td>348</td>
<td>3452</td>
<td>1965</td>
<td>696</td>
</tr>
<tr>
<td>//ifInOctets</td>
<td>15</td>
<td>2470</td>
<td>1602</td>
<td>350</td>
<td>3376</td>
<td>1967</td>
<td>695</td>
</tr>
<tr>
<td>//ifOutOctets</td>
<td>15</td>
<td>2503</td>
<td>1611</td>
<td>354</td>
<td>3408</td>
<td>1970</td>
<td>698</td>
</tr>
<tr>
<td>//ifInOctets</td>
<td>//ifOutOctets</td>
<td>30</td>
<td>3697</td>
<td>1620</td>
<td>365</td>
<td>4504</td>
<td>1981</td>
</tr>
<tr>
<td>//ifOperStatus[.='1']/</td>
<td>following-sibling:: ifInOctets</td>
<td>30</td>
<td>1905</td>
<td>505</td>
<td>1874</td>
<td>2778</td>
<td>680</td>
</tr>
</tbody>
</table>

Table 3. Processing Time and Message Size According to XPath Expression
traffic and processing time. Web Services is communicated between the manager and agent or gateway using SOAP protocol. This method is similar to our SOAP-based gateway. The network traffic overhead of Web Services is much greater than that of SNMP. However, network volume can be reduced by using a compression algorithm. Our work also proves that network traffic using the HTTP and SOAP communication protocols are much greater and can be remarkably reduced by a compression mechanism. Also, our work proves that HTTP-based and SOAP-based interaction translation methods show different performance results in their processing time. SOAP requires a slightly more computing resource overhead to process the SOAP messages in addition to HTTP messages.

6. Conclusion and Future Work

In this paper, we have presented a performance evaluation of our XML-based manager and XML/SNMP gateway in accordance to interaction translation methods. In particular, we measured network traffic volume between the manager and gateway and processing time, CPU usage and memory usage of the manager and gateway according to interaction translation methods. The message length of the SOAP-based is slightly longer than that of HTTP-based translation method because the SOAP header is added onto the HTTP message. The CPU usage and memory usage of both interaction translation methods show very little difference. However, the processing time of the SOAP-based gateway requires much more time than that of the HTTP-based gateway. The difference between processing time of the managers using two different interaction methods is lower than that of the gateways. From the performance evaluation results, we concluded that SOAP communication requires 2 times more processing time than HTTP in our interaction translation methods. That is, the HTTP method is more efficient in processing time creates less network overhead and require less computing resources.

Network traffic volume can be remarkably reduced by applying a compression algorithm although a slightly more processing time is required to compress and decompress the request/response messages. The CPU usage of the manager is a little higher than that of the gateway, however, the memory usage of the gateway is a little higher in both cases of interaction translation methods. Also, the XPath parsing overhead considerably affects the overall processing overhead of the gateway, and the use of an alternative expression or splitting the parsing overhead between the gateway and the manager increases the performance of the gateway.

In the future we hope to implement a tuning process to optimize the CPU usage of the XML-based manager and gateway. Also, we would like to devise a method for replacing the DOM parser with a SAX parser in both the manager and gateway in order to reduce memory usage. Specifically, the interaction translation methods of the gateway is different if we use a SAX parser because the SAX parser [24] does not support XPath expression. After developing a tuning process and replacing the a DOM parser with a SAX parser, we would again evaluate and compare performance using the same metrics we used in this current study.
References