A Proxy Creation Service for SNMP to XML Translations

Ricardo Neisse, Lisandro Zambenedetti Granville, Diego Osório Ballvé,
Maria Janilce Bosquiroli Almeida, Liane Margarida Rockenbach Tarouco

Federal University of Rio Grande do Sul - UFRGS
Institute of Informatics
Av. Bento Gonçalves, 9500 - Bloco IV
Porto Alegre, RS - Brazil
{neisse, granville, dob, janilce, liane}@inf.ufrgs.br

Abstract. Some proposals in the network management area use XML to encode the information model and the managed variables instances. In this paper we present an SNMP to XML dynamic proxy implementation to translate the information retrieved from network devices using SNMP to an XML representation. The proxy developed use the translation support between information models found in the libsmi and a SAX parser to instrument and dynamically create the proxy. We also present a monitoring supporting system to analyze an integrate various XML sources accessed through the proxy.

Keywords: Web-based Network Management, SNMP, HTTP, XML.

1 Introduction

The management of computer networks deals with the manipulation of management information typically located inside network devices. This information has to be defined according to some rules (e.g. SMIv2 [1], SPPI [2], XML [3]), and retrieved using some network protocols (CLI, SNMP [4], COPS [5], HTTP). Currently, an important problem is that the set of different options for the definition of management information and protocols increase the complexity of managing a network, since there is no consensus in a single definition language and protocol.

The IETF SMIng working group [6] has been working on the creation of a common definition language that could uniquely define management information. As part of this work, translations from other languages to the one defined in the working group was provided, for example, from SMIv2 to SMIng [7] and from SPPI to SMIng [8]. Once a unique definition language is provided and accepted, another problem will still remain: which unique protocol should be used? In our view, this question is unsolvable because we believe that several different protocols will be still required to manage older devices. However, from the network administrator point of view, the lack of consensus on a single protocol should not refrain the use of a single representation of the retrieved information. To allow that, protocol and information representation translations will be required.

In this paper we present a mechanism (and a supporting system) to translate SNMP-retrieved management information to XML files accessed through HTTP or HTTPS; thus, we have an SNMP to XML proxy. SNMP is used because it is the de facto management protocol widely supported in network devices. However, the information accessed by SNMP is originally defined using SMIv1 or SMIv2, which is not suitable when we are searching for a common representation. XML, on the other hand, seems to be more appropriated as a common language, besides being already addressed by the SMIng working group [6]. XML files can be easily parsed and manipulated, besides being text-based and human readable, protocol independent and supported by several current Web browsers.

We developed a system that automatically generates SNMP to XML proxies that reside in HTTP or HTTPS servers. The proxy generating system receives a SMIv1 or SMIv2 MIB definition as source parameter and creates a PHP4 script file that is the proxy itself. The just created proxy, when accessed via HTTP or HTTPS, contacts a target device via SNMP and generates an XML-based result. In our implementations we have used the `smidump` tool from the `libsmi` [9] package to

support the generation of the XML files, and the `expat` [10] package to provide the PHP4 support for the Simple API for XML (SAX) [11]. We have validated the proxy system through its use in a RRDTool-based [12] monitoring frontend.

The remaining of this paper is organized as follows. Section 2 presents related work. Section 3 presents the architecture and implementation of the proxy and proxy generator, while Section 4 shows how the proxies have been used in our monitoring tool. Finally, Section 5 finishes this paper with conclusions and future work.

2 Related Work

Currently, there are some proposals for network management that are not based on SNMP for the transfer of management information. Some of these proposals, for example, use HTTP to transfer the management information and XML to encode the managed objects, replacing the UDP and BER approach used in the SNMP architecture. The Distributed Management Task Force (DMTF) Web-Based Enterprise Management (WBEM) initiative [13] and the XML-based network management solution from Hong-Taek et al. [14] are examples of non SNMP-based architectures. Strauss et al., on their turn, use XML as an option for the SMIv1, SMIv2 or SMIng to encode management data model [9].

The DMTF WBEM initiative uses XML, HTTP, and object oriented concepts to manage networks through Web technologies. WBEM is based on the Common Information Model (CIM) [15] and uses XML to encode managed objects, while HTTP is used to transfer the data between manager and agent. Additionally, WBEM also provides a set of standards that defines the interaction between managers and agents.

Hong-Taek et al. present an SNMP to XML mapping in an embedded Web server architecture. They also define an SMI to XML schema mapping algorithm to translate an SMI definition to a XML schema. In order to access a MIB managed object instance inside a resulting XML, an access point to the MIB variables is defined through HTTP and XPath expressions in a DOM [16] tree. XML is also used to describe the information model and how the original MIB variables should be coded inside the messages exchanged between managers and agents. Actually, a proxy element is used to coordinate such interactions: a manager contacts a proxy that contacts the final agent.

The `libsmi` [9] is a library that allows management applications to access, check, dump and convert SMI MIB modules defined in SMIv1 or SMIv2 to various output formats such as SMIng, CORBA IDL and XML. The library provides a set of data structures, described in a UML model, to access the data definitions. Libsmi implements most of the SMIng functionalities, however, the DTD only consider the representation of an information model: no information about the values associated to the managed objects instances is supported in the DTD.

Although the schema used by Hong-Taek et al., WBEM, and the `libsmi` do not follow the same information model, the main purpose of these solutions is to reduce the latency between manager and agent interaction, and also to reduce the network overhead caused by excessive network management traffic [17].

3 Architecture and Implementation

In this section we present the architecture to SNMP to XML proxy operation and the system for SNMP to XML proxy creation.

3.1 SNMP to XML Proxy Operations

Figure 1 shows, how a proxy operates **after** its creation. A Web-based Network Management Station (NMS) requests management information accessing the SNMP to XML proxy through HTTP or HTTPS. In the simplest case, the management station is a standard Web browser that accesses an HTTP or HTTPS Web server. Such server (which contains the proxy) also receives from the NMS the address of a target device and an SNMP valid community. With this information, the proxy accesses the target device using SNMP in order to retrieve the management information. Normally, one single proxy access from the NMS generates several SNMP accesses to the target

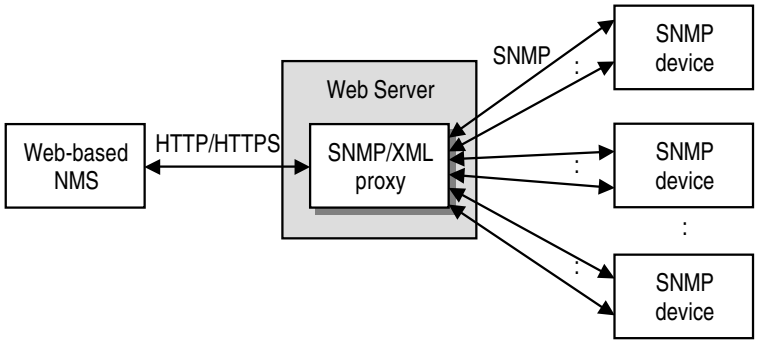

Fig. 1. SNMP to XML Proxy operations

Comparing, the amount of management information found in the NMS/proxy interactions is fewer than the amount of management information found in the proxy/target device interactions. Thus, pushing SNMP to XML proxies closer to the managed devices will reduce the overall amount of management traffic. From a certain point of view, the SNMP to XML proxy can be seen as an intermediate manager.

In this architecture, one single Web server can host several different proxies. When a request is fired from the NMS, the proxy to be used is determined in the URL passed in the HTTP or HTTPS request. The target device and SNMP community is also specified in the passed URL. Once a proxy is selected inside the Web server, **all** information that the proxy supports is retrieved from the target devices through SNMP. For example, in our current implementation, the URL `https://noc.metropoa.tche.br/interfaces.php.xml?ip=200.132.73.35&community=public` will contact the noc.metropoa. tche.br Web server and select the `interfaces.php.xml` proxy. The device whose IP address is `200.132.73.35` is accessed via SNMP using the `public` community. The information retrieved by the proxy is then compiled in a single XML file that contains all data related to the interface MIB-II group.

Since we based our implementation in the `smidump` tool, the XML returned to the NMS contains not only the value associated to the management information, but also the whole description of such information originally defined in SMIv1 or SMIv2. Thus, a new NMS that does not knows which information a managed device supports can discover that accessing an SNMP to XML proxy.

The proxy is implemented in a PHP4 script. New MIBs could be supported only through the development of new PHP4 scripts, creating new proxies. With the great variety of available proprietary MIBs, creating new PHP4 scripts every time a new MIBs should be supported turns to be a slow process. Trying to solve such problem, the next subsection presents a system that automatically creates new SNMP to XML proxies.

3.2 The System for SNMP to XML Proxies Creation

Figure 2 presents the simple architecture that supports the creation of new PHP4-based SNMP to XML proxies. The first step in a proxy creation is the transfer of standard SMIv1 or SMIv2 MIBs from a Web-based NMS to a Web server trough HTTP or HTTPS.

Internally on the server, the `smidump` tool checks the passed MIB and if no inconsistencies are found it generates an XML temporary file which is the XML version of the passed MIB. The next instrumentation step, on its turn, takes such XML temporary file and instruments it adding PHP4 code able to contact, via SNMP, a target device. The target device and its associated community string are treated within the instrumented code as variable whose values will be further defined when the proxy is accessed. Thus, the final instrumented XML file turns to be the new created proxy. In our current implementation, such file is stored in a standard directory in the Web server

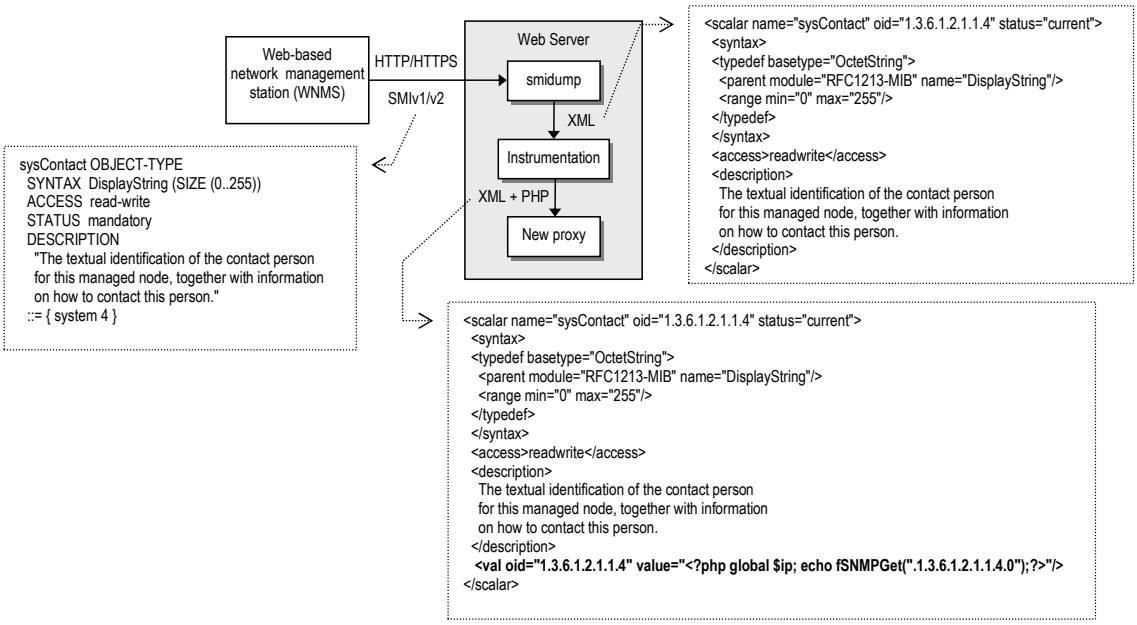

Fig. 2. Architecture for SNMP to XML proxy creation

and available to be accessed just after its creation. The original MIB passed from the the NMS station is also stored in another standard directory, for documentation purpose, as well as the intermediate XML file generated by the `smidump` tool prior to the instrumentation step.

With the proxy creation processes, new MIBs can be easily added to a Web-based management environment. In order to let this process work properly, however, the original MIB files should be properly defined according to SMIv1 or SMIv2. It is not rare to find today MIB files with definition problems. In this case, the intended SNMP to XML proxy will not be created, and the corresponding `smidump` message describing the errors found will be sent back to the Web-based NMS.

Finally, another point to be checked in the proxy creation is the fact that, although we have not changed the original `smidump` base DTD, we have arbitrarily included a <`val oid=... value=...` > XML construction in order to provide, in the final XML generated file, the OID and value associated to the retrieved management objects instances. Figure 3 shows, as an example, a portion of the XML returned to the Web browser generated by a previously created `interface` proxy. In this portion one can observe the information related to the `ifOperStatus` object from the MIB-II `interfaces` table.

4 Monitoring Tool

We developed an XML-based analysis and monitoring tool in PHP4 [18] that uses the SNMP to XML created proxies to retrieve management information in XML format. The analysis tool uses the Round Robin Database Tool (RRDTool) [12] library to store performance data and the MySQL database [19] to store analysis configuration data. Basically, the tool is an RRDTool frontend that acts as a generic XML monitoring, integration and analysis system. Any information available in XML format can be monitored and, considering that an SNMP to XML proxy can be generated dynamically for any MIB, the analysis tool can verify any information defined through SMIv1 or SMIv2.

The analysis tool is also based on Web technology. The network administrator access the tool through HTTP or HTTPS and defines which information have to be monitored. Such definition generates several configuration information stored in the MySQL database as a set of registers, each of them indicates the IP target device, the SNMP community string, the SNMP to XML proxy used to retrieve the data from the target device, the IP address of the Web server that contains

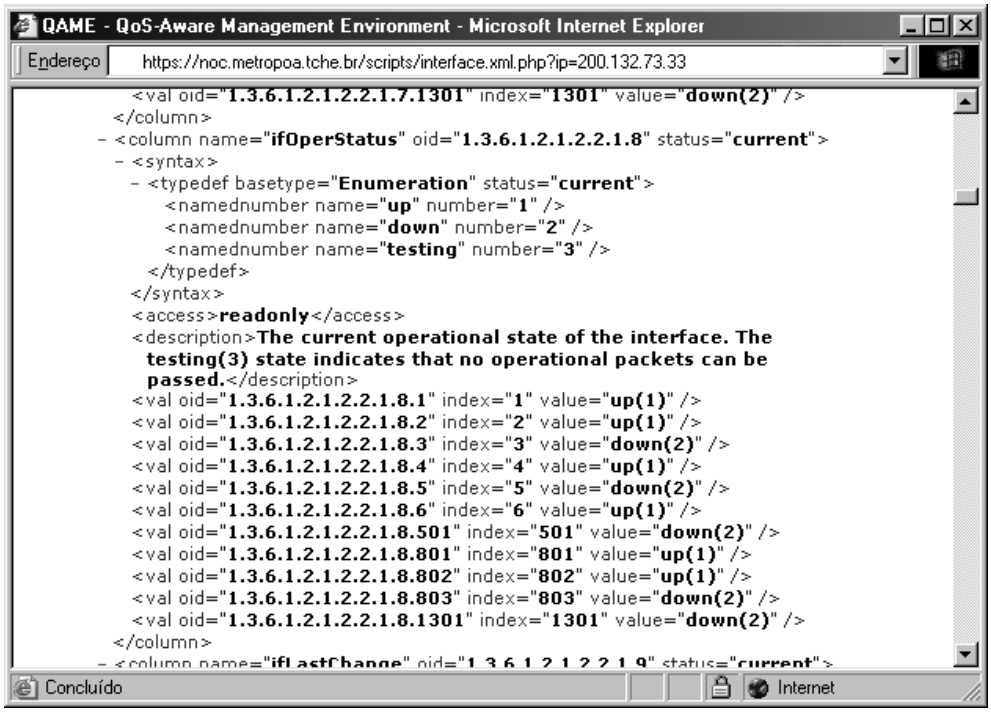

Fig. 3. SNMP to XML proxy example

the SNMP to XML proxy, and an XPath expression which locates, inside the retrieved XML, the specific information to be monitored. For example, one monitor configuration entry could be:

- Target device: 200.132.73.54
- SNMP to XML proxy: interfaces.xml.php
- SNMP community string: public
- Proxy Web server: noc.metropoa.tche.br
- XPath expression: //val[@oid="interfaces.ifTable.ifEntry.ifInOctets.2"]/@value

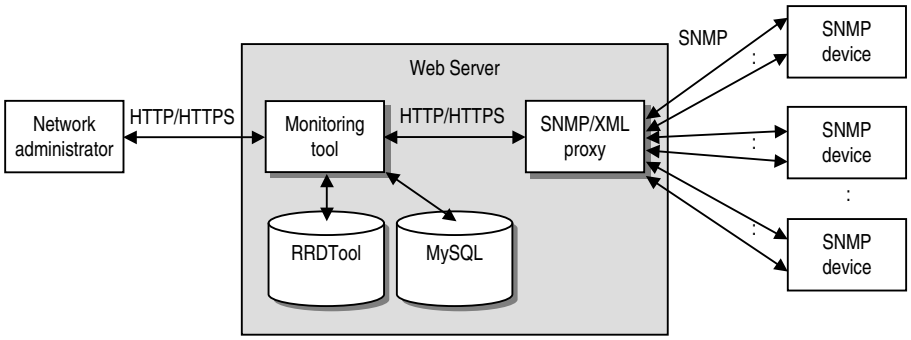

Fig. 4. Monitoring tool accessing an SNMP to XML proxy

This configuration will monitor the incoming traffic in the interface 2 of the device with IP address 200.132.73.54. The selection of the specific information, in the XML document, that will be monitored is done through a XPath selector dialog, which is an specialized PHP4 script for the SMIng DTD. The XPath selector locates, in the XML document, the original SNMP management objects values and suggests an XPath expression to further access.

Figure 4 presents one possible configuration for the analysis tool and a proxy interaction. In this case, both analysis tool and the accessed proxy are location within the same Web server. Due

to this configuration there are no network traffic overhead between the proxy and the analysis tool: only the SNMP traffic generated by the proxy to poll the target device.

As a result, figure 5 presents a real traffic data analysis generated through the Aberrant Behavior Detection (ABD) [20] algorithm of an university campus link in the Brazilian National Research Network. The ABD algorithm provides time series values prediction and is available only in the development version of RRDTool. The thick line is the observed value of the incoming traffic and the thin lines are the confidence band. Whenever the thick line crosses outside the confidence band it suggests an abnormal behavior in relation to the expected behavior, indicating an increasing or decreasing traffic faster than the past history.

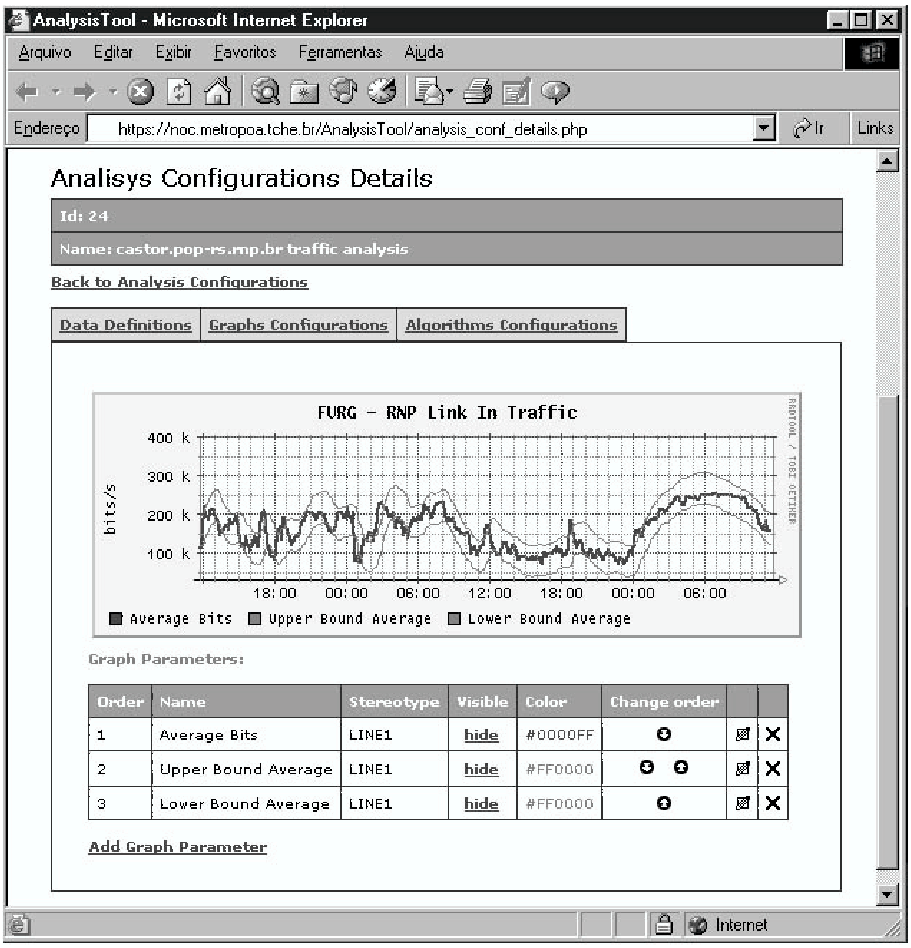

Fig. 5. Anomalous Behavior Detection

If desired, the monitoring tool could be located into a different device, differently from the previous example presented in figure 4, and no modifications are need to the architecture, as the access to the proxy is done through HTTP or HTTPS. Therefore, the physical location of the proxy is transparent to the analysis tool.

5 Conclusions and Future Work

We presented in this paper a dynamically SNMP to XML proxy creating tool that produces SNMP to XML proxies from standard SMIv1 or SMIv2 MIBs. Since the created proxies reside inside Web servers, they act as monitoring intermediate managers that uses SNMP to retrieve management information and generates XML document as a result. We have used the libsmi package to produce

an intermediate XML that is further instrumented with PHP4 code. We also provided a monitoring tool, accessible via HTTP or HTTPS, using XPath and DOM to monitor, integrate and analyze the information provided in XML.

One improvement for the SNMP to XML proxy is the implementation of a filter, internal to the proxy, that would receive and XPath [21] expression as an extra parameter in order to especify only the specific needed information that should be fetched and transferred to the management application. This will reduce the traffic between the NMS an the SNMP to XML proxy and also will reduce the processing overhead in the target device, because the filtering will be implemented in an intermediated agent, such as suggested by a discussion in an IRTF meeting [22]. The XPath expression combined with an Uniform Resource Identifier (URI) [23] is called into the XML terminology and XPointer [24] and provide some form of filtering mechanism for XML documents that can be used in the SNMP to XML proxy.

References

1. K. McCloghrie D. Perkins J. Schoenwaelder Structure of Management Information Version 2 (SMIv2), RFC 2578, April 1999.
2. K. McCloghrie, M. Fine, J. Seligson, K. Chan, S. Hahn, R. Sahita, A. Smith, F. Reichmeyer. Structure of Policy Provisioning Information (SPPI), RFC 3159, August 2001.
3. Extensible Markup Language (XML), http://www.w3.org/XML
4. Case, J., Fedor, M., Schoffstall, M. and J. Davin. The Simple Network Management Protocol, RFC 1157, May 1990.
5. D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, A. Sastry. The COPS (Common Open Policy Service) Protocol, RFC 2748, January 2000.
6. Next Generation Structure of Management Information (sming) Charter, The Internet Engineering Task Force (IETF), http://www.ietf.org/html.charters/sming-charter.html
7. F. Strauss, J. Schoenwaelder SMIng Mappings to SNMP, draft-ietf-sming-snmp-02, July 20, 2001.
8. Harsha Hegde, Ravi Sahita. SMIng Mappings to COPS-PR, draft-ietf-sming-copspr-01.txt, July 2001.
9. F. Strauß. Libsmi - A library to access SMI MIB information, http://www.ibr.cs.tu bs.de/projects/libsmi/.
10. The Expat XML Parser http://expat.sourceforge.net/
11. SAX - The Simple API for XML http://www.saxproject.org/
12. Oetiker T. Round Robin Database Tool (RRDTool) http://www.rrdtool.org
13. WBEM - Web-Based Enterprise Management (WBEM) Initiative, Distributed Management Task Force (DTMF), http://www.dmtf.org/wbem/
14. Hong-Taek Ju; Mi-Jung Choi; Sehee Han; Yunjung Oh; Jeong-Hyuk Yoon; Hyojin Lee; Hong, J.W. An embedded web server architecture for XML-based network management , Network Operations and Management Symposium, 2002, IEEE/IFIP , 2002 Page(s): 5 -18
15. Common Information Model (CIM) Standards, Distributed Management Task Force (DTMF), http://www.dmtf.org/standards/standard_cim.php
16. Document Object Model (DOM) http://www.w3.org/DOM/
17. Philippe Jean, Martin-Flatin, Ron Sprenkels. Bulk Transfers of MIB Data, The Quarterly Newsletter of SNMP Technology, Comment, and Events Volume 7, Number 1 March, 1999, http://www.simple-times.org/pub/simple-times/issues/7-1.html
18. PHP: Hypertext Preprocessor http://www.php.net
19. MySQL Open Source Database http://www.mysql.com
20. Brutlag, J. D. Aberrant Behavior Detection in Time Series for Network Monitoring, Proceedings of the 14th Systems Administration Conference (LISA 2000) New Orleans, Louisiana, USA December 3-8, 2000.
21. XML Path Language (XPath), http://www.w3.org/TR/xpath
22. Minutes of 1st NMRG meeting (Lausanne, Switzerland), November 1998, http://www.ibr.cs.tu-bs.de/projects/nmrg/minutes/minutes-001.txt
23. T. Berners-Lee, R. Fielding, U.C. Irvine, L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax, RFC 2396, August 1998.
24. XML Pointer Language (XPointer) Version 1.0, http://www.w3.org/TR/xptr/
25. F. Strauss, J. Schoenwaelder. SMIng - Next Generation Structure of Management Information, draft-ietf-sming-02, July 20, 2001.
26. M. Eder and S. Nag, *Service Management Architectures Issues and Review*, RFC 3052, Jan. 2001.