An XML-Based System for Configuration Management of Telecommunications Networks Using Web-Services

Adnan Umar, James J. Sluss Jr. and Pramode K. Verma

The University of Oklahoma,
4502 E. 41st Street, Building 4, Room 4403
Tulsa, Oklahoma 74135, USA

Abstract. As the utilization and the application base of the Internet grows, the need for an improved network management system becomes increasing apparent. It is generally accepted that SNMP is not capable of tackling the arising network management requirements and needs to be replaced. Also, configuration management has been identified as one of the most desired network management functionality. Recent research publications suggest a growing interest in replacing SNMP by a Web Services (XML)-based network management solution. In this paper we present our methodology and design of our complete XML-based network management system developed with the specific aim of performing configuration management. [1], [2]

1 Introduction

As the utilization and the application base of the internet grows, the need for an improved management system becomes increasingly apparent. The management of the Internet is traditionally based on the framework of SNMP. The SNMP was designed almost fifteen years ago to address the network management needs of that time. Back then, the networking environment was very different. The primary goals of SNMP were to perform device-level management, be extensible, and efficient in using communication and processing resources. Today, advances in technology have dramatically changed the networking environment. This dramatic change has altered the management requirements significantly. Scarcity of bandwidth and processing power is no longer an issue and heterogeneous networks are commonplace. Configuration management has been identified as the most desired management functionality. Inadequacies of SNMP and the need for a new management technology have also been brought to light. The two prime candidate technologies for the development of a new management system appear to be XML and Java. Currently many companies and standards bodies are working on developing an XML-based network management system. To study the use of XML in network management system within the wider research community, there needs to be a design and open-source implementation that would facilitate research. In this paper, we present our methodology and design of our complete XML based network management system developed with the specific aim studying configuration management. To the best of our knowledge, no such effort is being undertaken at this time [1], [2], [3], [4].

Umar A., J. Sluss Jr. J. and K. Verma P. (2005).
An XML-Based System for Configuration Management of Telecommunications Networks Using Web-Services.
In *Proceedings of the Joint Workshop on Web Services and Model-Driven Enterprise Information Systems*, pages 3-10
DOI: 10.5220/0002544200030010

2 Methodology

The task of developing an XML based system includes specifying design requirements, choosing the appropriate XML technologies, and testing the XML design using generic software tools. This task can become very challenging since the XML technologies are constantly changing and the software tools are often playing catch-up. For this reason we have adopted a methodology with which we can organize our design process. Our methodology consists of three activities. An ‘activity’ can be defined as the process of taking iterative-steps to accomplish a task. First, we decompose our management problem into functions and map these functions to XML technologies. Second, using the results of our first activity, we piece together our XML design. Third, we test our design by implementing a subset of it using generic tools.

3 Network Management Functions

Our first activity in designing the management system was to identify the major functions and map them to the appropriate XML technology. We have identified the following functions that we believe an XML based network management system using Web Services requires. This approach allows us to keep track of the evolving XML technologies and facilitates the implementation process. The guideline we followed for mapping the network management functions and XML technology is that, they should be closely aligned such that only generic tools are required for implementation. That is to say, if implemented correctly using generic XML tools, our design should perform the required configuration management task. These network management functions and their respective XML technologies are listed below:

- **Defining structure of management information (XML Schema [9], [10], [11]).** The ability of representing a very large variety of information in a homogeneous fashion is crucial to success of a management system. This function can be performed by the XML Schema technology. XML Schema is an XML language that is capable of defining the structure of a XML document. That is, it specifies which tags are permitted and in which nesting order, and constrains on the number of occurrences of a particular tag, etc. A tag can be made optional or required and the value’s data-type can be declared. Another impressive feature of XML Schema is its ability to validate a XML document. This feature can be used to reduce code complexity by catching erroneous XML documents, that do not match the defined tag-value structure, before they get passed on to the application.
- **Handling the management data (XML Document [6]).** All information needs to be represented in a form that allows information to be accessed, modified, searched, and retrieved. This function can be performed by XML documents. These documents will be based on a XML Schema that defines its structure and the way information should be represented and

can be validated against it. XML documents are also a convenient way for storing information.

- **Navigating in the management documents (XPath [7] and XPointer [18]).** In an XML based management system, all information exists in the form of XML documents. The number, size and complexity of these documents can get rather large. Therefore, a function is required that is capable of navigating through a maze of XML documents. The XML community has addressed this issue by developing two specification called XPath and XPointer. XPath is a recommendation that defines how nodes within an XML documents can be accessed by forming an expression. These expressions play a major role in other XML standards, such as, XSLT and XQuery. XPointer is built on XPath and includes URI addressing making it possible to address fragments of an XML file.
- **Providing an interface between XML document and management applications (DOM [12] and SAX APIs).** In any XML based system there is a need for an interface between XML documents and the application. Although applications can treat the XML documents as a text document and use their own parsing scheme, it is much better to use a standard parser. Currently there are two popular XML interfacing standard parsers: SAX and DOM.

  SAX (Simple API for XML) is a joint development of the members of the XML-DEV mailing list. SAX is very simple, easy to learn, and not demanding on resource such as memory. Various SAX parsers are available for Java, C++, Python, Perl and Delphi.

  DOM (Document Object Model) is a complete interface to an XML document. Using DOM applications can parse, retrieve, add, modify, and delete sections of the XML documents. Since DOM stores the entire XML document in memory, it can be very intensive on resources. Popular implementations for DOM are MSXML from Microsoft for Windows and Apache Soft ware Foundation's Xerces that exists in Java and C++.
- **Changing format of XML documents (XSLT [8]).** An important issue for XML in general is document transformation. Often information stored in XML documents is formatted for the purpose of storage. To make the information useful, often re-formatting is required. For his purpose XML community has developed XSLT. XSLT is a powerful technology capable of transforming one document into another document with a different format. Several implementation tools for XSLT are available for various platforms, such as, Java, C++, and Perl.
- **Describing the management interface (WSDL [13]).** Since we are employing Web Services we cannot avoid the use of WSDL (Web Services Description Language). WSDL is an XML language that describes a web service. WSDL compilers are used to generate executable code and are available for various platforms and programming languages.

  SNMP systems handle this function by providing MIBs can be used by the MIB compilers. Data can by accessed by SNMP functions such as Get, GetNext, and Set.

- **Transporting parts of or entire XML documents (SOAP [14], [15], [16]).** This function is responsible for providing the means for communication. We require this means of communication to be capable of using any transportation protocol. We can map this function to the XML standard called SOAP. SOAP is one of the Web Services standards developed for exchange of information. SOAP is not tied to any transport protocol and can use SMTP, FTP, etc. However, HTTP is the most popular transportation protocol used by SOAP.

4 Design

The second activity in our methodology is creating a design. This design is realized by using the results of the first activity. The proposed design (Fig. 1) creates a system of XML documents that are capable of representing all desired management functions. These documents can be implemented on any OS using any of the large number of software available for XML. Currently, XML programming tools are available for all major software platforms including .NET, COM, C/C++, and Java. Furthermore, XML can also be implemented using scripting languages such as, PHP, Cold Fusion, Perl, Python and Tcl. For every design there needs to be a set of requirements that are to be fulfilled. We are using the list of requirement mentioned in [1]. Following is the list of requirement and an explanation on how our design fulfills them:

- **Maintaining a clear distinction between configuration and operational data.** In our design we maintain a clear distinction between configuration and operational data by keeping them in separate XML documents. Both can have their respective XML Schema document to help validate the XML documents containing the data. It also appears desirable to handle the two separately in management application as well.
- **Providing functionality to download and upload a small part or entire configuration files.** Our design is capable of transporting XML documents by using SOAP between the network manager and network device. These XML documents can be a small part of or entire configuration file. Operational information is also to be passed using this mechanism.
- **Ensuring that configuration data is kept in text format for interoperability.** Everything in XML is kept in a text format. Therefore, this requirement is met inherently by using the XML technology.
- **Enabling the devices to hold multiple configurations, one of which can be active at any given time.** If one configuration can be stored in one XML document, then by having multiple XML files multiple configurations can be kept in one device. To have only one active however, appropriate functions need to be implemented at the application level.

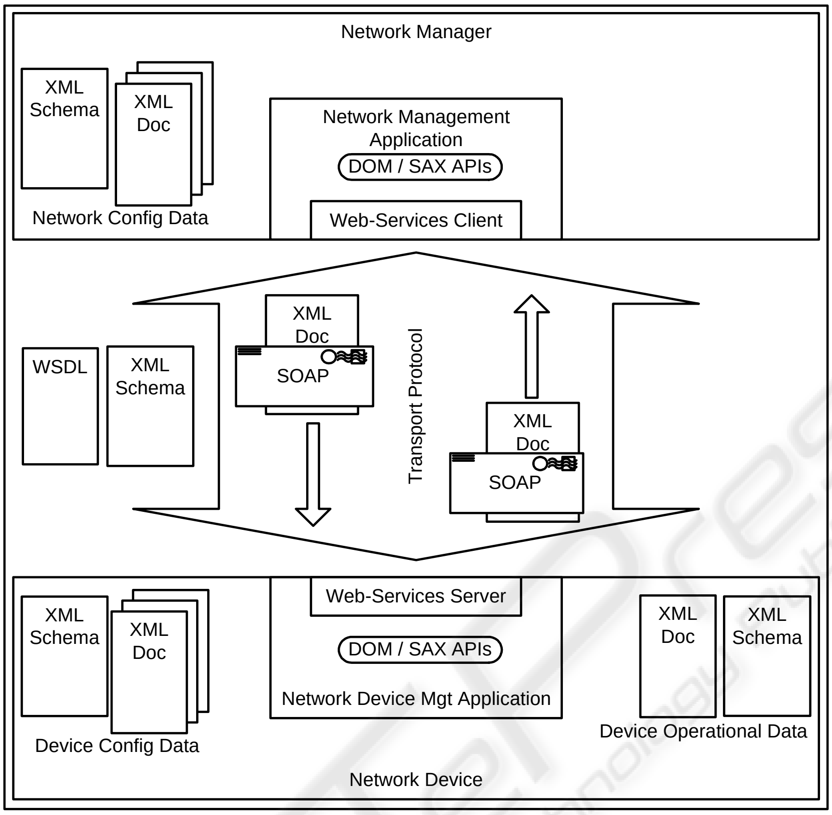

Fig. 1. Proposed XML-Based system for configuration management using web-services

- **Providing a simplified mechanism for coordinated activation of configuration taking into account loss of connectivity during a management transaction.** A simplified mechanism for coordinating activation of configuration will need to be handled at the application level. In our design SOAP provides the connectivity and the application can benefit from its functionality.
- **Being easy to use and cost effective. Ease of use and cost effective requirement is met inherently by using XML.** Unlike SNMP, XML is a generic technology which makes XML tools, open-source software, and developers with XML expertise much easier to find and is cost effective.

5 Testing

Testing of the XML design is the third activity in our methodology. In this section we show how we tested our XML design. The main objective of this activity is to verify that our design can be implemented using generic tools. We did this verification by implementing a subset of our design that included all the functions that we identified in our first activity. In our implementation we are using Jakarta Apache Tomcat [20] and Apache Axis [19] as the web-server and SOAP engine respectively. The challenge we face is that Apache Axis's does not fully support all the features of web services and supported features are often difficult to implement. For these reasons we follow the technique suggest in [5] (Fig. 2). By adopting this technique we can easily achieve document/literal style SOAP encoding and XML Schema validation support by only performing a few steps. This technique demonstrates how Castor [22] can be used in conjunction with Apache Axis to easily implement any web-service. We tested our network management web-service using XML Spy [23] by placing a sample XML document with sample configuration data. JDOM [21] was used to access the XML document. The steps of the implementation can be found in [5].

Some problems however, still remain. Notification operation is not supported by Apache Axis's WSDL2Java tool. Furthermore, it is not entirely clear how web-services handle the notification operation, since there are no clear guidelines on how a client can register itself with the server. A custom solution can be developed, but will lead to interoperability issues. We believe that these issues will be resolved by the XML community in the near future.

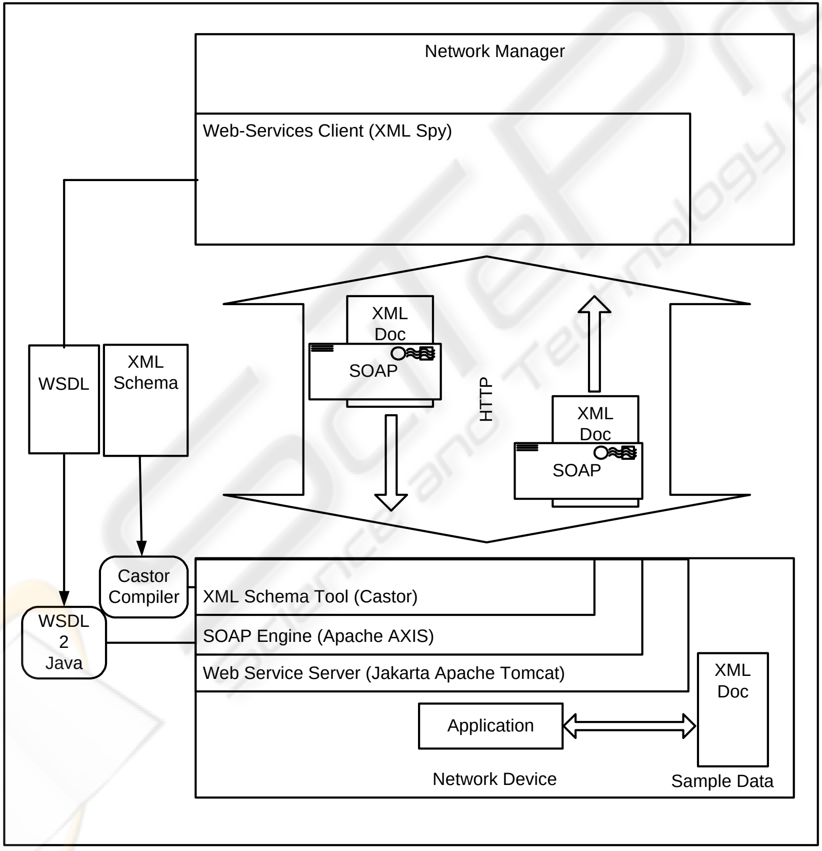

Fig. 2. Prototype and testing setup

6 Conclusion and future work

In this paper we have presented our methodology and design of our XML-based network management system using web services. This methodology and design is intended to provide a framework which will enable us to study configuration management problems using XML based management system in great detail. Future work currently underway entails an open source implementation of our design. This implementation is intended to be a tool for research related to configuration management.

Acknowledgements

Authors would like to acknowledge Keith Allen, Weijing Chen, and Will Chorley of SBC Labs for providing an opportunity to work with the XML technology in network management.

References

1. Jürgen Schönwälder, Akio Pras, Jean-Philippe Martin-Flatin, "On the Future of Internet Management Technologies," IEEE Communication Magazine, October 2003
2. L. Sanchez, K. McCloghrio, and J. Saperia, "Requirements for Configuration Management of IP-based Networks," RFC 3139, June 2001
3. Jae-Oh Lee, "Enabling Network Management Using Java Technologies," IEEE Communication Magazine, January 2000
4. Frank Strauβ, Torsten Klie, "Towards XML Oriented Integrated Network Management," 2003. IFIP/IEEE Eighth International Symposium, March 2003
5. Kevin Gibbs, Brian D Goodman, and Elias Torres, "Create Web Services using Apache Axis and Castor," IBM developersWorks, http://www-136.ibm.com/developerworks/
6. T. Bary, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation, October 2000
7. J. Clark and S. DeRose. XML Path Language (XPath) Version 1.0. W3C Recommendation, Inso Corp. and Brown University, November 1999
8. J. Clark. XSL Transformation (XSLT) Version 1.0. W3C Recommendation, November 1999
9. David C. Fallside. XML Schema Part 0: Primer. W3C Recommendation, IBM, May 2001
10. Henry. S. Thompson, David Beech, Murray Maloney, and Noah Mendelsohn, XML Schema Part 1: Structures, W3C Recommendation. University of Edinburgh, Oracle Corporation, Commerce One, Lotus Development Corporation. May 2001
11. Paul V. Biron and Ashok Malhotra. XML Schema Part 1: Datatypes. Kaiser Permanente, Microsoft. May 2001
12. L. Wood, et al. Document Object Model (DOM) Level 1 Specification. W3C Recommendation, SoftQuad, October 1998
13. E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. W3C Note, Microsoft, IBM, March 2001
14. Nilo Mitra. SOAP Version 1.2 Part 0 : Primer. W3C Recommendation, Ericsson, June 2003

15. Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik F. Nielsen. SOAP Version 1.2 Part 1: Messaging Framework. Microsoft, Sun Microsystems, IBM, Canon. W3C Recommendation. June 2003
16. Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik F. Nielsen. SOAP Version 1.2 Part 2: Adjuncts. Microsoft, Sun Microsystems, IBM, Canon. W3C Recommendation. June 2003
17. Hugo Haas, Oisin Hurley, Anish Karmarkar, Jeff Mischkinsky, Mark Jones, Lynne Thompson, Richard Martin. SOAP Version 1.2 Specification Assertion and Test Collection. W3C, IONA Technologies, Oracle Corp. AT&T, Unisys, Active Data Exchange.W3C Recommendation June 2003
18. Steve DeRose, Ron Daniel Jr, Eve Maler, Jonathan Marsh, Norman Walsh. XML Pointer Language (XPointer). Brown University, Interwoven, Arbortext Inc, Sun Microsystems, Microsoft. August 2002
19. Apache Axis, Apache Software Foundation. http://ws.apache.org
20. Jakarta Apache Tomcat, Apache Software Foundation. http://jakarta.apache.org/tomcat
21. JDOM, http://www.jdom.org
22. Castor, http://www.exolab.org
23. XML Spy, Altova, http://www.xmlspy.com