An ICT-oriented Management Solution for NGNs

Pedro Gonçalves
ESTGA/IT
Universidade de Aveiro
Aveiro, Portugal
pasg@ua.pt

Ricardo Mendes, Rui Aguiar
DETI/IT
Universidade de Aveiro
Aveiro, Portugal
ramm@ua.pt, ruilaa@ua.pt

***Abstract*— NGN architecture reused several standards from the IP world, as exemplified by the Session Initiation Protocol SIP, which is ubiquitous in the majority of these network components. However, the NGN management architecture simply presented a very generic management model that follows TMN. Several management technologies are proposed, such as Web services, CORBA and SNMP, to implement management solutions. Network and systems management standardizing bodies currently promote newer technologies that aim to solve known shortcomings to these. This paper proposes a management solution for NGNs based on recent IP world technologies. The presented solution was implemented in the form of a middleware to manage NGN elements. This middleware was used in the management of an element belonging to the IP Multimedia Subsystem platform, namely the Policy and Charging Rules Function.**

Keywords-component - NGN management; Middleware; technology integration; NETCONF; WBEM.

I. INTRODUCTION

Next Generation Networks (NGNs) started a new era, where the convergence of different worlds truly began: both the convergence of fixed and the merger of telecommunications and the Internet technology become prevalent in NGN. In terms of network convergence, NGN proposed an extremely modular network architecture where control of the transport stratum is agnostic in terms of transport technology, thus allowing to integrate traffic from / to different access networks technologies. As related to technology integration, NGN reused several standards from the IP network world, as exemplified by the Session Initiation Protocol SIP that is ubiquitous in many of these networks components.

Contrary to what happened with the innovative network architecture development, the management architecture specification is quite conventional. The existing documentation specifies a very generic architecture that follows the *Telecommunication Management Forum* (TMF) [1] management model. It proposes several management technologies such as Web services, CORBA and SNMP to implement the management solutions. The standardization bodies that have been working in the definition of management technologies in the area of network and enterprise management are *Internet Engineering Task Force* (IETF) [2] and *Distributed Management Task Force* (DMTF) [3].

In the enterprise management area, the DMTF standardized several technologies like the *Desktop Management Interface* [4], the *Web Based Enterprise Management* (WBEM) [5] and the WS-Management [6], while promoting a vision of integrated management and including the management support for network equipment. In the area of network management, IETF has developed several technologies such as SNMP [7], COPS [8] and more recently NETCONF [9] that addresses several issues raised to the SNMP technology. NGN management has not cope with these evolutions.

This paper proposes a new management approach, exploring the IP-world technologies for the management of NGN. The use of new standards is compatible with the novel TMF management model, but brings the inherent advantages associated with these standards. Given that NGN architecture scope includes aspects related to services and networks, we chose a management approach able to address both areas, relying in technology from system management (WBEM) and a technology from network management (NETCONF). WBEM is an open technology, very flexible, that can easily carry out integrated management of a complex management scenario such as a NGN network. Additionally, and being a popular technology with a vast number of implementations, it enables rapid prototyping of management solutions. NETCONF, on the other hand, has been specified by the IETF to manage network equipment and has been receiving much acceptance by both academy and industry.

In order to allow an integrated management of various aspects of NGN, we proposed a solution that integrates, in the form of an adaptive middleware, the network and systems management. In order to validate our technology adaptation approach, a component from the IMS platform, the Policy Charging and Rules Function (PCRF) [10], was used as a test element. This paper thus describes a management solution that combines the flexibility and comprehensiveness of the WBEM approach with the NETCONF suitability to the computational requirements of network elements. Although our management concept is broader in order to be concise, we focused our implementation on the policy provisioning process, an adapter that allows transfer rules between a central server and a PCRF [10]. Of course, we need to show the reliability of such a mixed approach, witch requires some sort of technology adaptation, able to cope with data model transformation and protocol adaptation.

This paper is organized as follows. Section II gives a general overview of the management technologies involved in this work, and Section III describes related work. Section IV describes the developed system and Section V analyses its performance, and conclusions are finally provided in Section VI.

II. TECHNOLOGY OVERVIEW

WBEM [5] was initially proposed by companies from the desktop management area, and was later developed by the DMTF. WBEM specification includes a set of technologies imported from the web world, such as the HTTP based transport mechanism (*CIM operations over HyperText Transfer Protocol* (HTTP)) [11] and the XML based specification for the information encoding (CIM-XML). The data model used in the WBEM technology is the *Common Information Model* (CIM) [12], a data model proposed by the DMTF that aims to integrate management information of the desktop and of the network areas, witch thus seems specially indicated to our objectives.

CIM is a three-layer object oriented data model, composed of a set of abstract classes and associations that model the generic common characteristics of the management fields such as networks, systems, users, etc., that developers extend in the form of derivate classes. CIM specification describes its basic modeling concepts and meta-schema design; the *Managed Object Format* (MOF) language specifies how information is rendered; and a Schema defines the semantics for a wide range of managed objects and relationships between them. Such a modular and extensible data model allows the integration of multiple management data and enables the development of integrated management solutions. CIM data model was later reused by DMTF in the specification of a new technology based on Web services, named WS-Management.

CIM operations over HTTP [11] specify a vast diversity of operations (named methods in the specification terminology) including methods for classes and qualifier manipulation, methods for instance and property handling, methods for indication dispatch and for class method invocation.

WBEM solutions include four components: the CIM client typically used by the human operator during management tasks, a *CIM Object Manager* (CIMOM) that is the main component of the system maintaining the dialogue with the CIM client and the management information, a CIM repository and CIM providers, that perform the interface between the CIM server and specific managed equipment such as a managed server or a router.

WBEM technology received significant attention by both industry and academy, having been used as the enabling technology management solutions for various scenarios [13-16]. Despite the completeness of its data model, WBEM technology is mainly used in the area of system management.

WBEM architecture requires that providers be created for handling the CIM extensions. Given that, providers act as an adapter between the management server and the managed elements. Providers are the most appropriate element in the architecture to implement adaptation to management technologies. This can explain the associate amount of work in the literature [14, 15, 17, 18]. In [15], Yoon et al. describe a WBEM-based management system for residential gateways that interfaces the managed equipment through a SNMP interface. System features include equipment configuration, performance monitoring and equipment fault detection. Seo et al. [14], describe a *Network Management System* (NMS) for managing DiffServ-over-MPLS QoS in an inter-domain scenario, where in addition to interconnect SNMP and WBEM, it performs admission control using a COPS-RSVP interface, implemented in a dedicated provider.

On the other hand NETCONF [9] is a management protocol standardized by the IETF in 2006 that defines operations for managing network devices, allowing to upload, retrieve and manipulate management configuration data. The protocol is based on a XML-encoded Remote Procedure Call (XML-RPC) over secure transports as SSH [19], SOAP [20], TLS [21] or BEEP[22]. In order to maintain the interoperability between NETCONF management solutions it was decided that SSH transport implementation was mandatory. The protocol modularity was promoted by means of a architecture composed of four layers: the content layer containing configuration data; the operations layer implementing the management operations; the RPC layer implementing the XML-RPC remote procedure call; and the transport protocol layer that implements the information transport between management entities. Although initial specification just included configuration operations, it was later standardized the NETCONF monitoring support [23].

The protocol was designed independently of the management data model, and therefore the RFC that specify the protocol does not include any considerations about the data model. So, the IETF NETMOD working group proposed a data modeling language named YANG [24] for the generation of the management data models. YANG defines a set of data nodes organized as a hierarchical tree. Each data node in the tree has a name and either a value or a set of child nodes. YANG schema is structured into modules and sub-modules that may be published by a standards organization, an enterprise or an industry forum [25]. A key YANG design feature is the modular extensibility. One YANG module may define additional data nodes augmenting the data nodes defined in another YANG module. YANG was adopted in this work as the NETCONF data model language.

Although additional operations could be provided based on the capabilities advertised by the devices, the base protocol defines nine operations [9] for datastore data management and two operations [23] for the management of event notification information.

During the recent years NETCONF has received much attention by both academia and industry, and several NETCONF *Software Development Kits* were developed [26-29].

Ozianyi et al. [13] describe an XML-driven framework for Policy-based QoS management for a IMS network. The proposal consists in a NETCONF *Network Management System* (NMS) that implements policy based management of IMS *Policy Control and Charging* subset [10], integrated with a variety of network elements that include COPS-PR, Diameter and NETCONF support. They assess the bandwidth utilization and the communication delay between the management elements of the NETCONF and the COPS-PR technologies, concluding that compressed NETCONF interfaces perform better than COPS-PR interfaces, for some higher values of information (transferred configuration composed of more policies).

More recently Enns et al. proposed an update [30] to RFC 4741 where they add a YANG module for NETCONF operations, removing his description from the XML Schema Definition. Also they create a username and the requirement for NETCONF servers to perform user authentication and permissions authorization according to the user profile. It has been developed by Perelman et al. [31] a reduced version of NETCONF original protocol, named *NETCONF Light*, that includes a subset of the original protocol, envisioned for devices with limited computing resources. Among the main differences from the original protocol highlights the lack of support filtering configuration for *NETCONF light* operations. Despite maintaining original protocol operations, this light version removes the possibility of defining a filter that limits the operations scope over the equipment configuration.

III. SYSTEM DEVELOPED

In our management concept, CIM server acts as the central management point of the IMS platform providing all the flexibility required for an integrated service and network management we then resort to NETCONF to network management, for flexibility and efficiency [32]. A new middleware then performs the technology adaptation between WBEM and the devices running NETCONF. Figure 1 illustrates the overall system architecture from the management application to the managed device.

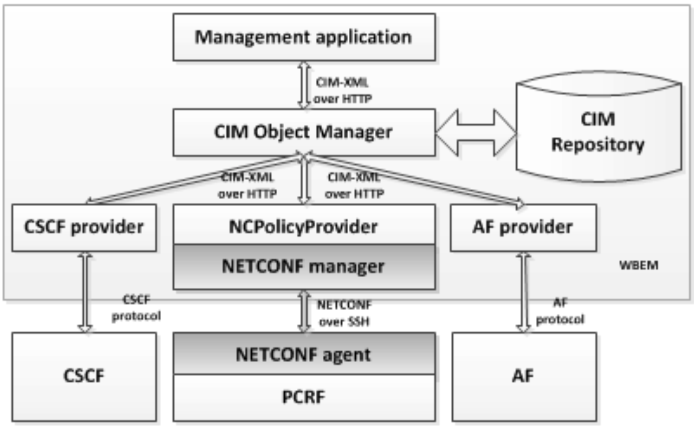

Figure 1. System architecture

As an implementation use case, we centered in the management of the PCRF, an IMS functional entity that performs admission control, resource management and charging, based on policies. For simplicity of discussion, we focused here on the development of a middleware that allows that policies specified by the system administrator in the NMS, to be pushed via NETCONF over SSH interface to the PCRF. In this section we describe the information model and the message sequences between the CIM server and the NETCONF agent.

The NMS has been developed based on a CIM server with some extensions to the CIM data model. It was used a CIM client where the administrator performs the specification of policies, which in turn are delivered to the NMS. The system includes a repository that maintains the previously defined policies. A CIM provider that implements a NETCONF interface and communicates with the NETCONF agent makes the adaptation of policy information. The developed middleware consisted in an *OpenPegasus* provider based on a development platform named CIMPLE [33] and was developed in the C language. The development followed a modular architecture, having been created logic to decode WBEM messages, to encode XML components and to implement a NETCONF manager. The XML messages encoding is performed using a *Xerxes-C* library and the implementation of the NETCONF manager has been carried out using a NETCONF over SSH implementation [27].

A. *Data model*

Policies that are used by the system define the behavior of PCRF, determining how to make admission control and how to manage network resources. A system policy is formally defined as an aggregation of policy rules. Each policy rule is composed of a set of conditions and a corresponding set of actions. The condition defines when the policy rule is applicable. Once a policy rule is activated, one or more actions contained by that policy rule may be executed. In our middleware development of reused some previous work [34] as well as the policy representation data model.

The data model associated with NETCONF component was derived from the CIM component data model. CIM classes were converted into *leaflists* and association classes were converted as *leafrefs*. CIM class properties were implemented as leafs of a YANG equivalent built-in data type and were placed as leaf elements inside the *leaflists*. CIM class methods were declared as NETCONF operations with YANG module scope, since the language does not allow the method declaration with a scope associated with YANG constructs.

B. *Operation encoding*

Technology adaptation implies, besides the data model conversion, the matching between the operations of both technologies. TABLE 1 identifies the immediate matching between the WBEM and NETCONF operations.

TABLE 1. WBEM and NETCONF operations match

WBEM	NETCONF
GetInstance	<get-config>/<get>
EnumerateInstances	<get-config>
CreateInstance	<edit-config>
ModifyInstance	<edit-config>
DeleteInstance	<delete-config>
ExportIndication	<notification>

WBEM operations have a granularity of the object, given the nature of its object oriented data model. When some WBEM operation is performed, it affects an object. In the case of NETCONF, when some operation is performed it affects a complete document or a part of it, if a filter expression is provided to the operation. As formerly described, policies are represented as the aggregation of objects from several classes, and they are transferred in WBEM technology in a per-object basis.

The message match mechanism described in the TABLE 1 is extremely inefficient, since it would execute a NETCONF operation sending a small policy component, when it could receive the complete WBEM policy description and then send the full policy to NETCONF agent with a single operation. For instance in the *read policy* use case, the system needs several *GetInstances* to visualize the entire policy but only one <*get-config*> to obtain the total policy from the managed device. On other hand, depending on the NETCONF agent capabilities, the <*edit-config*> should be preceded by the <*lock*> operation and succeeded by the <*commit*> and <*unlock*> operation, to atomize the submit operation. Most cases follow this trend: the direct operation match is inadequate. For example the policy creation process requires several WBEM *CreateInstance* operations and a single atomic <*edit-config*> operation in the NETCONF technology. Figure 2 illustrates the create policy use case where our middleware receives the complete list of policy components from WBEM component, and after recoding it, performs an atomic policy transfer to the NETCONF agent.

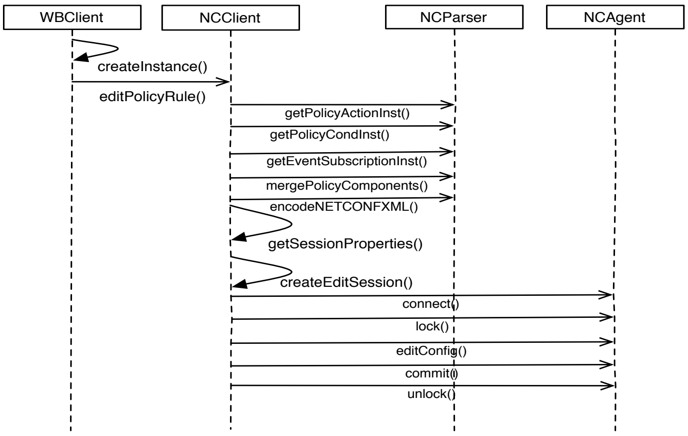

Figure 2. Policy creation sequence diagram

IV. SYSTEM ANALYSIS

This section presents the scenario used to evaluate the management system, defines the evaluation uses cases and evaluates the results.

A. *Test scenario*

The test scenario included three machines running the different applications: a machine that ran the CIM client, another that ran the CIM server with the developed provider and a third machine with the NETCONF agent. The network segment was isolated from the local LAN in order to avoid the network traffic to bias the evaluation results. The WBEM server was installed in a 2.0 GHz Intel Centrino with 1GB RAM, and the client was installed in a 2.6 GHz Pentium 4 core with 512 MB RAM, both running 2.6.34 Linux Kernel.

The following use cases, typical for a PCRF, were considered for system analysis: (i) read the running policy from the Managed Device; (ii) edit the property "*Caption*" from the "*UA_ActionNull*" class and submit the changes; (iii) create a policy from scratch. For each use case the tests were performed, the traffic was captured and analyzed on the WBEM and on NETCONF interfaces of the middleware, and the memory consumption for each measured.

B. *Traffic analysis*

Upon execution of the three test cases, the WBEM network traffic captures were filtered, analyzed and quantified. TABLE 2 contains for each test, the amount of traffic discriminated by each of the layers of the protocol stack, as well as their relative weight. To simplify the traffic analysis, the Secure Sockets Layer (SSL) support was separated of the rest: the session establishment process includes the connection and the authentication of the client in the WBEM server.

TABLE 2. WBEM traffic

	Session Establish.		Read Policy		Edit Policy		Create Policy	
Packets		8		417		10		601
Messages		1		102		2		45
Comp.	Kb	%	Kb	%	Kb	%	Kb	%
CIM-XML	0	0	206,3	69.68	4,2	73.07	103,9	61.33
HTTP	0,8	60.51	62,9	21.24	0,9	15.59	26,8	15.8
TCP	0,3	19.15	13	4.4	0,3	5.5	18,8	11.09
IP	0,2	11.97	8,1	2.75	0,2	3.43	11,7	6.93
Ethernet	0,1	8.38	5,7	1.93	0,1	2.4	8,2	4.85
Total	1,3	100	296	100	5,7	100	169,4	100

A *read policy* operation requires that 417 packets and 102 messages are exchanged between CIM entities until the complete policy could be presented in the CIM client. A deeper analysis in the traffic exchanged shows that before showing any class, the CIM client performs at least three requests to the CIM server: *GetClass*, *EnumerateInstanceNames* and *GetInstance*. Additionally it was observed that the CIM client repeated those operations for each instance it receives, thus requiring a high number of operations. Apart from the messages exchanged between the CIM client and server, the generated traffic is further increased by the HTTP transport overhead, which exceeds 20% of generated traffic.

By its turn, the *edit policy* is less demanding on traffic, just 10 packets and 2 exchanged messages. In this case the high fragmentation of the policy in separated classes may be an advantage, because we only need to retrieve the policy class that contains the property that is intended to edit. For the policy creation, it took 601 packets and 45 messages. Paradoxically this test case requires more packets and less information (more than half on the read case) despite being the same policy. In this case, there are less messages exchanged, 45 versus 102. Editing a class requires two requests to the CIM server: *GetClass* and *CreateInstance*.

Upon execution of the three test cases, the NETCONF network traffic captures were filtered, analyzed and quantified. TABLE 3 contains for each test, the amount of traffic discriminated by each component as well as their relative weight. We further indicate the load of session establishment. As before the session establishment corresponds to the establishment of an SSH connection with its key exchange between the NETCONF Manager and NETCONF Agent.

The *read policy* use case from the NETCONF component, involve obtaining the policy from the running repository through the operation <*get-config*>. This operation is quite simple, requiring only one message. The policy itself represents more than 80% of the traffic exchanged. For the edit policy use case, the changes are submitted by the NETCONF Manager, sending the policy component containing the change. The system policy follows the conceptual approach, events, conditions and actions: *<UA_EventSubscription>*, *<UA_PolicyCondition>* and *<UA_PolicyAction>* respectively. In this case exists a bigger granularity in the submitted changes comparing with WBEM, and by other way the NETCONF agent supports the candidate capability which means that four requests must be made to submit the changes in an atomic way: *<lock>, <edit-config>*, *<commit>* and *<unlock>*. Finally the create policy case, is the operation that spends more traffic, for the same reasons pointed before in the edit policy case. Entire policy plus the *<lock>*, *<edit-config>*, *<commit>* and *<unlock>* requests.

TABLE 3. NETCONF traffic

	Session Establish.		Read Policy		Edit Policy		Create Policy	
Packets		41		23		19		27
Messages		4		1		4		4
Comp.	Kb	%	Kb	%	Kb	%	Kb	%
XML	0	0	15,8	80.17	8,8	66.78	18,3	76.97
RPC	0	0	0,2	1.2	0,9	7.14	0,9	3.99
SSH	7,3	73.92	2,2	11.12	2,2	16.82	2,8	11.7
TCP	1,3	12.93	0,7	3.65	0,6	4.49	0,8	3.56
IP	0,8	8.08	0,4	2.28	0,4	2.8	0,5	2.22
Ethernet	0,6	5.66	0,3	1.6	0,3	1.96	0,4	1.56
Total	9,9	100	19,7	100	13,2	100	23,7	100

C. Memory requirements

An analysis was performed to the memory consumption in the WBEM server, in the SSH daemon and in the NETCONF agent during an entire edit operation, which comprehends the session establishment, the read and the edit policy procedures. The SSH daemon and the NETCONF agent showed a constant behavior during the entire process: SSH daemon occupied 580kB and the NETCONF agent 1212kB. The memory used by the WBEM server grew from operation to operation: 8924kB when the service was started, 9456kB when the session was established, 11076kB when *NetConfSession* instance was created, 14152kB when the policy information was read and 15052kB when the policy information was edited.

The software was subjected to profiling tools in order to detect problems with the resource usage (memory, processing). It analyzed memory usage; the time spent executing each method and the sequence of the application methods invocation. A framework named *Valgrind* was used to detect errors resulting from incorrect use of dynamic memory and a helper application named *Callgrind* recorded the call history among functions in a program's run, as a call-graph. The profiling analysis made to the system showed that there are no problems in memory usage and confirmed that results are coherent, since both emphasize *Xerces-C* methods as the most time and memory consuming. This is easily justified, since *Xerces-C* is used for parsing XML, and this corresponds to the major part of the processing.

D. Load analysis and profiling analysis

They were some load tests run in order to verify system stability and scalability. Tests were done using a command line CIM client (*cimcli*), invoked by some bash scripts that perform read policy requests to the system. Two types of tests were performed: a test where the WBEM server made several calls to different agents conducting a NETCONF request, and another test with the WBEM server making several requests to a same agent.

Figure 3 illustrates the memory behavior of the WBEM server as result of the multi agent scenario. Tests show an increase in memory consumption, which means that the limitation for the reception of new requests from the CIM server is the memory of the machine where the process is running. Indeed this is what happens, because of 3500 applications, the CIM server was only able to process 2961, reaching the limit of available memory (2GB), and the server process was killed by the O.S.

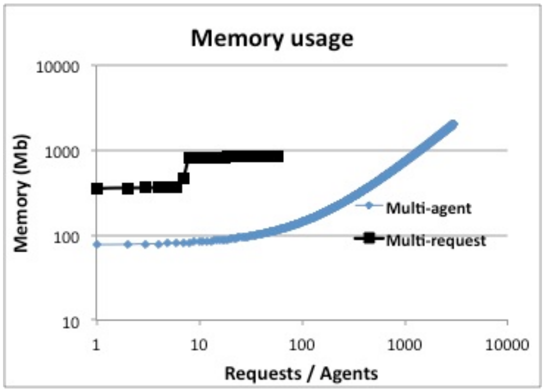

Figure 3. Memory behavior in load tests

In the single agent scenario, tests performed various requests to a same agent; the requests were executed in parallel, imposing a much higher load to the server. Figure 3 illustrates the behavior of memory consumption in this situation. In this case the memory behaves differently due the simultaneous requests and does not constitute the limitation because the limit has not been reached. However the processing power in server capacity CIM to answer the multiple simultaneous requests in a timely manner. The system was able to process 58 requests, rejecting the others by a timeout.

V. CONCLUSION AND FINAL DISCUSSION

This paper presents a NGN management solution based on ICT-world technologies. It consists in a CIM server middleware that integrates the two management technologies from the enterprise and the network management. WBEM is a widely used technology in the system management area, with plenty of implementations and commercial products. NETCONF is a newer network management technology that addresses SNMP shortcomings and has been receiving lots of attention.

A management solution consisting of a technology integrating middleware allows by on hand to implement a management system capable of managing a vast diversity equipment, as the functional entities that exist within the IMS platform; and by the other hand it allows to adapt the technologies, using the most appropriate management technology for each equipment type. Additionally, and using an integration technique that integrates the data models used in the telecom and the enterprise management [35], the proposed solution could cope with upper layers of the telecom management technology.

Although a broader architecture was proposed, the discussion was centered on the implementation of the policy provisioning process, and the developed middleware was applied to the management of an IMS element named PCRF. The middleware was functionally evaluated and stress tests conducted in order assess its scalability and its applicability to a production scenario.

The results show that these technologies scale in a promising way to its usage in a production network and, especially the need for NETCONF was evident considerably more efficient than WBEM closer to network elements. WBEM technology offers greater flexibility, allowing the management of a greater equipment range, given the vast richness of its data model.

REFERENCES

[1] TM Forum, http://www.tmforum.org/browse.aspx, 2012-04-09.

[2] The Internet Engineering Task Force (IETF), http://www.ietf.org/, 2012-04-08.

[3] DMTF, Desktop Management Task Force Home, http://www.dmtf.org/home, 2012-04-09.

[4] DMTF, Desktop Management Interface Specification - v 2.0.1, v 2.0.1, 2003.

[5] J. P. Thompson, "Web-based enterprise management architecture", Communications Magazine, IEEE, vol. 36, pp. 80-86, 1998.

[6] DMTF, Web Services for Management (WS-Management), 2005.

[7] J. Case, M. Fedor, M. Schoffstall, and J. Davin, Simple Network Management Protocol (SNMP), RFC 1157, 1990.

[8] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry, The COPS (Common Open Policy Service) Protocol, RFC 2748, 2000.

[9] R. Enns, NETCONF Configuration Protocol, RFC 4741, 2006.

[10] J. J. P. Balbas, S. Rommer, and J. Stenfelt, "Policy and charging control in the evolved packet system", Communications Magazine, IEEE, vol. 47, pp. 68-74, 2009.

[11] DMTF, Specification for CIM operations over HTTP version 1.2, 2007.

[12] DMTF, "Common Information Model (CIM) Specification - Version 2.28", 2011.

[13] V. G. Ozianyi, R. Good, N. Carrilho, and N. Ventura, "XML-Driven Framework for Policy-Based QoS Management of IMS Networks", Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE, 2008, pp. 1-6.

[14] J.-C. Seo, H.-S. Kim, D.-S. Yun, and Y.-T. Kim, "WBEM-Based SLA Management Across Multi-domain Networks for QoS-Guaranteed DiffServ-over-MPLS Provisioning", in Management of Convergence Networks and Services. vol. 4238, ed: Springer Berlin / Heidelberg, 2006, pp. 312-321.

[15] B. W. Yoon, S. Ahn, and J. W. Chung, "Web-Based Home Gateway Management System using SNMP", 17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management Large Scale Management, Dublin, Ireland, 2006.

[16] J.-h. Yoon, H.-t. Ju, and J. Hong, "Development of SNMP-XML translator and gateway for XML-based integrated network management", Int. J. Netw. Manag., vol. 13, pp. 259-276, 2003.

[17] S.-J. Lee, M.-J. Choi, S.-M. Yoo, J. W. Hong, H.-N. Cho, C.-W. Ahn, and S.-I. Jung, "Design of a WBEM-based Management System for Ubiquitous Computing Servers"

[18] Y.-J. Oh, H.-T. Ju, M.-J. Choi, and J. W.-K. Hong, "Interaction Translation Methods for XML/SNMP Gateway", Proceedings of the 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management: Management Technologies for E-Commerce and E-Business Applications, 2002.

[19] M. Wasserman and T. Goddard, Using the NETCONF Configuration Protocol over Secure Shell (SSH), RFC 4742, 2006.

[20] T. Goddard, Using NETCONF over the Simple Object Access Protocol (SOAP), RFC 4743, 2006.

[21] M. Badra, NETCONF over Transport Layer Security (TLS), RFC 5539, 2006.

[22] E. Lear and K. Crozier, Using the NETCONF Protocol over the Blocks Extensible Exchange Protocol (BEEP), RFC 4744, 2006.

[23] S. Chisholm and H. Trevino, NETCONF Event Notifications, RFC 5277, 2008.

[24] M. Bjorklund, YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF), RFC 6020, 2008.

[25] E. Nataf and O. Festor, "End-to-end YANG-based configuration management", Network Operations and Management Symposium (NOMS), 2010 IEEE, 2010, pp. 674-684.

[26] B. Zores, R. State, and O. Festor, YENCA, http://sourceforge.net/projects/yenca/, 2011-03-22.

[27] Netopeer - Remore configuration system using NETCONF protocol, http://code.google.com/p/netopeer/, 2012-04-21.

[28] Yuma - YANG-based Unified Modular Automation Tools, http://sourceforge.net/projects/yuma/, 2012-04-21.

[29] P. Tavares, P. Gonçalves, and J. L. Oliveira, "An IDE for NETCONF management applications", 7th Latin American Network Operations and Management Symposium, Quito, Equador, 2011.

[30] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, Network Configuration Protocol (NETCONF), RFC 6241, 2011.

[31] V. Perelman, J. Schoenwaelder, and M. Ersue, Network Configuration Protocol for Constrained Devices (NETCONF Light), draft-schoenw-netconf-light-00.txt, 2011.

[32] P. Gonçalves, J. L. Oliveira, and R. L. Aguiar, "An evaluation of network management protocols", 11th IFIP/IEEE International Symposium on Integrated Network Management (IM 2009), New York, USA, 2009.

[33] M. E. Brasher and K. Schopmeyer, "CIMPLE: An Embeddable CIM Provider Engine", March 2006.

[34] P. Gonçalves, R. Azevedo, J. L. Oliveira, and R. Aguiar, "Managing QoS in a NGN using a PBM approach", The Fourth International Conference on Systems and Networks Communications- ICSNC 2009 Porto, Portugal, 2009.

[35] TMForum, CIM-SID Solution Suite Release 6.0, 2006.