УДК 681.3:621.39:51

Andriy Luntovskyy, Dr.-Ing. habil., Marius Feldmann, Dipl.-Inf., and Alexander Schill, Prof.Dr.rer.nat.habil.

Web services based Network Management

This paper examines arguments for Web service based integration of network management and design tools being developed in the context of the CANDY Framework. The elaborated network design routines are provided with collected project data about real operating networks. The offered approach combines powerful network management and continuous reengineering workflow aimed at improving the already created infrastructure. SNMP integration into hierarchical and layered management tools increases the overall quality of design and reengineering solutions essentially. The innovativeness of the examined integration method based on Web services lies in utilization of modern technologies SOAP and REST. The created software-technical architecture possesses indispensable advantages in aspects like simplicity, flexibility and extendibility.

В данной статье исследована актуальная задача создания интегрированных систем проектирования и сетевого менеджмента, основанных на применении Web services. Разработанные в рамках автоматизированной системы CANDY Framework процедуры проектирования локальных сетей получают возможность использования реальных данных, аккумулирумых с использованием протокола SNMP. Предложенный подход сочетает эффективный сетевой менеджмент со встроенным маршрутом проектирования и пригоден для улушения ранее созданной инфрастуктуры сети. Применение SNMP в составе иерархических, многоуровневых средств сетевого менеджмента в ходе процесса реинжиниринга существенно повышает технико-экономические показатели проектных решений. Новизна описанного метода интеграции программных средств на основе Web services состоит в использовании технологий SOAP and REST. Разработанная программно-техническая архитектура несет в себе безусловные практические преимущества в части простоты, гибкости и расширяемости.

1. Introduction

The World Wide Web has reached a status far away from just being a system of interconnected Hypertext documents. Due to their simplicity, their ability of integration and the availability of various development tools, Web technologies are nowadays used for various application areas for which they were not intended originally. Network management (NWM) is one of these areas.

The term network management describes the set of means, activities, methods, procedures and tools that concern with the operation, maintenance, administration and provisioning of computer networks. Choosing the right elements for this set and integrate them into the network is one important aspect of today's approaches used during the overall network design workflow. This includes particularly the choosing of useful NWM protocols, the placement of NWM systems and software and the planning of the infrastructure for monitoring devices. Being relevant for the network design workflow, research on NWM has been done for extending the network design Utility (CANDY) Framework [1] by the possibility to offer design decisions for NWM. CANDY has been introduced to try new and innovative ways and ideas in network design. It claims to be a complete integration platform covering all aspects of managing a networks lifecycle which is possible due to its flexible structure depicted in Figure 1. This figure gives an impression of the various tasks that have to be supported by a utility covering network design and reengineering. Beside multiple technological aspects such as topological or capacity calculations and analysis it has to concern with economical aspects that are not negligible. As pointed out in [2] in comparison to other approaches, CANDY has evident advantages like tool simplicity, open source and freeware tools, the use of XML as main integrating component, openness (use as Framework) and extensibility for new tools. For the purposes of data representation, of communication and of configuration the XML-based Network Design Markup Language (NDML) [3] is used.

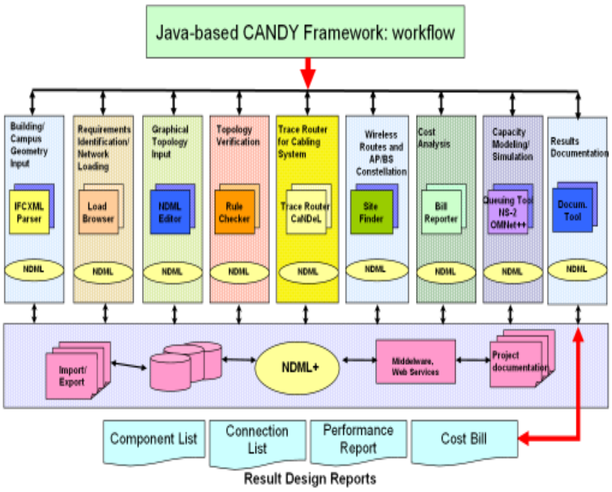

Fig.1. The CANDY Framework

One of the strategies that is used by CANDY to improve the design of a network is to continuously collect information generated by network management and monitoring tools after the network has been deployed. This information is used during a network reengineering workflow that takes place to restructure parts of the network or to exchange single components if they have been identified as bottlenecks. The reengineering workflow leads to a cyclic network design process that is depicted in Figure 2. It shows that it is a central goal to offer management information by Web services. These Web services are accessed by the CANDY Framework to collect different information and summarize this information in a NDML-based fault report (summary of device and service availability, network load, general errors and performance parameters) that is used as fundamental input for redesigning parts of the network.

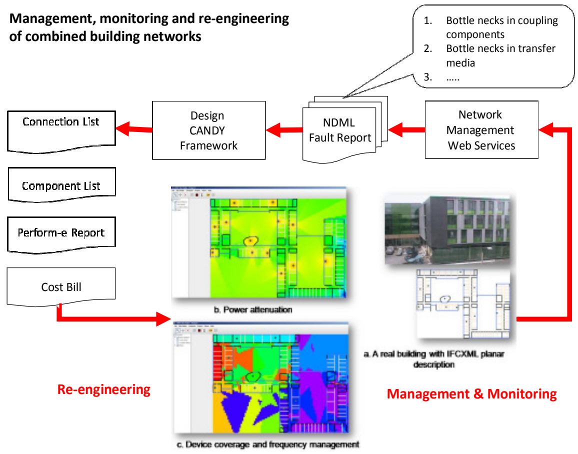
Management, monitoring and re-engineering of combined building networks
1. Bottle necks in coupling components
2. Bottle necks in transfer media
3. .....
Connection List
Design CANDY Framework
NDML Fault Report
Network Management Web Services
Component List
Perform-e Report
Cost Bill
b. Power attenuation
a. A real building with IFCXML planar description
Re-engineering
Management & Monitoring
c. Device coverage and frequency management

Fig.2. Network Design Workflow

The central question that has arisen was how to provide a proper NWM infrastructure that on one hand represents a powerful and efficient administration instrument during the regular network deployment and maintenance and on the other hand makes it possible to integrate management and monitoring information in an easy manner into the reengineering workflow before making them available via Web services. Due to the fact that NWM is a very old and important concern, a number of different technologies and protocols connected with this topic have been developed recently making it necessary to choose an adequate one. For this reason the first part of this paper provides an in-depth argumentation why the decision has been made to use a Web-based solution. After this the second part gives an overview about possible Web-based architectures and describes a concrete Web-based solution that has been developed as a system used within the CANDY project and that introduces major improvements in comparison to former approaches.

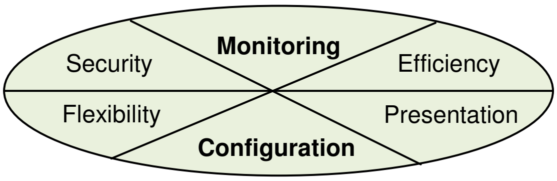

Fig.3. Requirements for NWM tools

The remainder of this paper is structured as follows. Section 2 defines important general requirements for modern NWM tools. After this Section 3 focuses on the exceptionally wide-spread Simple Network Management Protocol. It is analyzed if this protocol fulfils the requirements defined before. This analysis gives the motivation to improve NWM by using Web principles. Section 4 gives an overview about Web technologies that are used in modern Web applications. Based on this knowledge a survey of Web-based NWM is provided in Section 5. Due to some shortcomings of many current Web-based NWM systems and tools, an improvement using a lightweight application model is provided in Section 6. The paper finally closes with a conclusion and a short view on future work summarized in Section 7.

2. Requirements for Network Management tools

Up-to-date NWM tools often come with a very wide range of functionality covering different demands and usage scenarios. Having a look at available functionalities, categories of recurrent requirements for well-functioning NWM tools can be defined. These six categories or dimensions are summarized in Figure 3.

The two most obvious requirements are support for monitoring and for configuration. First of all, monitoring describes the verification if a device or a service is available. In most simple cases this information is announced by the binary representations “is available” and “is not available”. Beside simple status information, monitoring includes collection of information regarding working loads of network components and regarding appeared alarms. Monitoring the working load of devices should make available more detailed information than a simple binary status. It should provide exact values of a special attribute, and may present averaging of values during a defined space in time. An alarm always describes some sort of alteration in the network. An alteration may be the change of the status of a network device like the change from the state “is available” to “is not available”. Furthermore it is desired to be informed about the exceedance of predefined boundaries such as a predefined maximum of working load of a router or of a switch. Alarms should include information like the source, the reason and the exact time of occurrence. Additional to the named monitoring aspects in some cases rule-based monitoring is necessary. For example, it enables grouping of network device attributes and definition of in-depth constraints regarding such defined groups. The support for configuration of network entities enables the manipulation of settings of network devices and services. Owing to the complexity of modern computer networks this task should not be done manually but should be mainly automated. Sometimes even script-based configuration is desired, which leads to a further increase of automation and makes mass operations possible.

In addition to monitoring and configuration, security, flexibility, efficiency and presentation are common requirements for NWM tools. Security does not only include a secure exchange of information over a network by means of encryption, but also user or group dependent definition of access rights. Some security guidelines in a computer network result in the need for a high flexibility of a NWM tool. This counts particularly for the necessity to deal with firewalls. Beside avoiding problems caused by general network restrictions, flexibility stands for aspects like ability of integration, modularization and extendibility. These abilities are necessary for using a NWM tool in an arbitrary application context. It may be the case that different sorts of 3rd party data stores have to be accessed during configuration or that the NWM tool operates on different information models, such as the Management Information Base or the Common Information Model. In some cases data export functionality is desired that support various common data formats to increase the ease of integration of other applications. Furthermore flexibility demands the coverage of a huge set of platforms or even platform independence. By offering open and well-defined interfaces for configuration and monitoring tasks extendibility can be achieved. Moreover modularization results from a separation of concerns such as introducing a set of specialized commands. All these criteria are summarized by the desired flexibility. The category of efficiency covers the demand for a very small overhead of transmitted management information. Protocols that are used for monitoring and configuration should not increase the network load in an unacceptable manner. High efficiency and a small overhead of a NWM solution results in enhancement of scalability. The last identifiable category is one that contains everything connected with presentation. It is evident that an easily and efficiently usable user interface is typically required to meet administrator needs. Beside a simple access to management information it should also support meaningful and potential various different output formats. Particularly monitoring information, such as alarms, should be communicated to an administrator in a straightforward way. Furthermore users do not want to be confronted with too much technical details that may take place at low management protocol levels if they are not necessary thus leading to demanding abstraction and transparency regarding the information presentation and data output of the NWM tool.

Though the named categories have some non empty intersections and interact with each other, it is obvious that an entire achievement of all elements of the six dimensions of requirements is not possible as some of them contradict each other. This counts, for example, for the aspects of security and of efficiency. In practice, not all elements of the six dimensions are achievable by one NWM tool. In fact it is often even not necessary to achieve them all. In some cases the presentation can be rudimentary, in others there is no need for very complex security support.

It is evident that a NWM tool realizes the different categories of functionalities on different technical levels. While configuration and monitoring may be directly supported by a low-level network management protocol, presentation aspects are introduced on a higher application level. Furthermore complex network management operations may be introduced on a higher level by combining simple operations available at a low-level protocol. This points out that NWM tools fulfill the different requirements by a hierarchically arranged technological structure. Before having a look at which aspects of the defined categories are directly supported by or can be improved by using Web service based NWM, a focus is directed on the low-level Simple Network Management Protocol. After giving a short overview about this protocol, it will be pointed out, which of the above mentioned categories it covers to a higher extend and at which points it lacks of support.

3. Simple Network Management Protocol

The Simple Network Management Protocol is a solution for remote network management used by most of the industry since the early 1990s. After its first version [4] has been published in 1988, it has been extended by further functionalities and concepts. Today three main versions, plus additional subversions of SNMP are available. The elementary principle of the protocol has stayed the same. There exist two roles, agents and managers who interact by exchanging messages encoded by using the Basic Encoding Rules (BER). Agents are network entities like routers, switches or services that are managed and monitored by the manager. The communication is predicated on two different interaction schemes. The first scheme is a bidirectional one based on a request-response mechanism. A manager formulates a request and sends it to the agent. On one hand a request can be used to get the current value of an attribute defined by an agent such as the identification and description of a network entity or the current processor load of a router. The information the manager is interested in is included into the agents response. On the other hand a request enables the configuration and thus the manipulation of attributes of an agent. In this case the response contains a confirmation that the status change of an attribute has been realized successfully. A manager might send a request to set the packet counter of the eth0 interface of a router to zero and the response contains the value ‘0’. Returning the new value of a changed attribute is often done in practice. Beside the bidirectional communication a second, unidirectional way of interaction is offered. It is used for sending notifications from the agent to the manager. These traps might be used to indicate errors or to inform the manager about other sudden status changes.

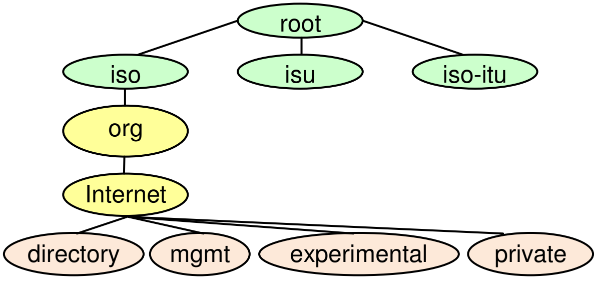

Fig.4. MIB excerpt

The values that can be read and modified by a manager - the so called managed objects - are described in the Management Information Base (MIB). The MIB is organized in a tree-like manner. A small except of this structure is given in Figure 4. The nodes of this tree can be addressed by using an alphanumerical or a numerical identifier. These Object Identifiers (OIDs) are specified in the requests by the manager. For example the name of a system has got the identifiers

- *iso.org.dod.Internet.mgmt.mib.system.sysName*
- and *1.3.6.1.2.1.1.5*

The first version of SNMP, SNMPv1 defines five different Protocol Data Units (PDUs) used for the communication between manager and agent. Transmitted SNMPv1 messages mainly contain the used SNMP version, a string called the community name and a data block that includes the PDU. SNMPv1 messages are transmitted via the unreliable UDP. The five available PDUs are *get-request*, *get-next-request*, *get-response*, *set-request* and *trap*. The *get-request* PDU is sent by the manager to the agent and results in the delivery of the value of a given OID. Besides error related information this PDU contains a request identifier and a variable binding that specifies the OID to which the request relates. The response of the agent is done by using the *get-response* PDU. Values of an OID can also be fetched by using the get-next-request PDU though in this case the agent responses with the value of the OID that succeeds the specified one. The *set-request* PDU makes it possible to set the value of an included OID to the one transmitted as an argument. For the named unidirectional communication the *trap* PDU is used by the agent. The content of such a PDU differs from the one of the others. Beside further information it contains the OID of the object that has caused the trap, the network address of the associated agent and a timestamp defining the time passed since the last initialization of the agent. The overall structure of SNMPv1 communication is presented in Figure 5.

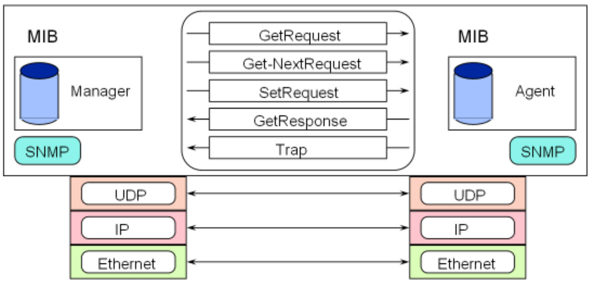

Fig.5. Simple Network Management Protocol

The mentioned SNMPv1 community string was introduced as a possibility to achieve a basic authentication mechanism. An interconnected sub-tree of the MIB managed by one SNMP agent and an arbitrary number of SNMP management stations can be summarized as a community. Each manager can join various communities. If communication between manager and agent occurs, the membership in a special community is indicated by including the community string as an identifier into the exchanged message. This identifier is transmitted in plain text. Though there exists an extension defined in [4] for encrypted authentication, this mechanism is not well-defined and thus is not used in practice. SNMPv1 does not support strong authentication and encryption algorithms. This criticism was one argument for developing a new version of SNMP – SNMPv2. In fact SNMPv2 is not one specification but consists out of different subversions. The reference version SNMPv2 introduced a so-called party based mechanism that has been viewed as overly complex and inefficient and has not found broad acceptance. Instead the de-facto standard of the second generation of SNMP is the community version SNMPv2c [5]. Though it is still based on the described community concept and thus does not really increase security, it brings more efficiency, flexibility and is more powerful. One improvement is the removal of transport dependence that existed in SNMPv1 with its focus on UDP. For example now even the Connectionless Network Service (CLNS) or the Internetwork Packet Exchange (IPX) were mentioned as potential candidates. A further change occurred in the area of PDUs. The trap PDU does not differ from the other PDUs any longer. Beside this further PDUs were introduced. The *get-bulk-request* PDU makes it possible to request more than one object from an agent. This increases efficiency because an iterative transmission of get-next-request PDUs to receive objects in series is not necessary any more. The *inform-request* PDU has been introduced to enable a manager-to-manager communication. For example this can be used to for forward SNMP messages that have been received by one manager to a further one. The use of the new *report* PDU has not been defined in detail. A further change is the extension of the MIB definition by introducing the MIB-II that comes with a larger set of objects.

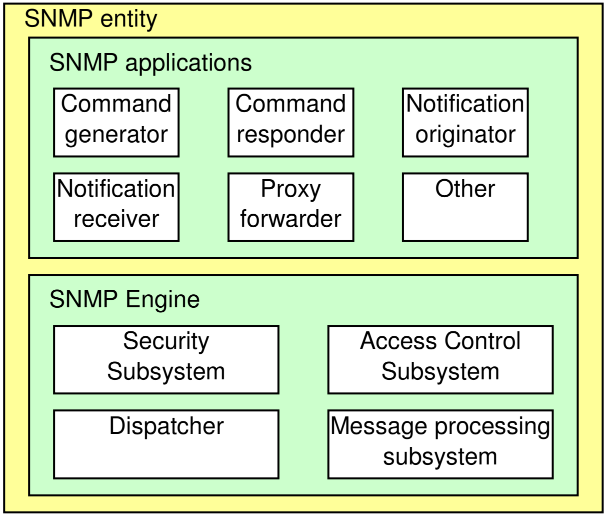

Fig.6. SNMPv3 entity architecture

A central goal during the development of the latest version of SNMP - SNMPv3 - was to create a unified security concept and by this bringing improvement to the area of authentication and encryption. The initial specification of SNMPv3 [6] defines a complete new Framework and introduces even different terms. Managers and agents are now named *SNMP entities* in general although in this paper the differentiation between manager and agent will be retained. The structure of such a SNMP entity is presented in a summarized manner in Figure 6. The architecture consists of various modules that are used on demand. A manager normally includes the *command generator* and *notification receiver* on application layer while an agent should contain a command responder and a notification originator. These modules on the application layer make use of the underlying SNMP engine. The *security subsystem* realizes the authentication and encryption of messages. It has to be implemented by all interacting instances. Potentially different security models may be used. At the moment only the user-base security model (USM) [7] is well defined. It uses regular algorithms such as the Data Encryption Standard and the Secure Hash Algorithm 1 to offer confidentiality and authentication. The *access control subsystem* has to be implemented on the agent side. It checks if a request that targets a special area of a MIB is valid due to defined access rights of the originator. The *message processing subsystem* offers compatibility to former SNMP versions. It enables transformation of version specific formats to a universal format. Finally, the *dispatcher* builds the glue between the message processing subsystem and the underlying network protocol layer. It forwards messages to this underlying layer which are then transmitted to the communication partner. SNMPv3 offers further transport protocols. For example [8] defines the use of TCP.

Following this introduction to SNMP, the appropriateness of SNMP for modern network management will be pointed out and a short evaluation of this protocol will be provided. It is checked if SNMP is appropriate to build a low-level basis for a hierarchically structured network management system that offers Web-based access on a higher level.

As mentioned above, the six dimensions monitoring, configuration, security, flexibility, efficiency and presentation have been identified as requirements for network management tools. The aspect of *presentation* can be ignored at the moment, as SNMP definitely has no support for presentation. This has to be realized by higher level functionality of a NWM tool. By its functional range SNMP supports *monitoring* an agent and notifying a manager about errors or other non-regular conditions in the agents by using the trap PDU. Therefore it avoids a need for polling these devices. However, no complex rule-based monitoring engine is provided that enables a way to define extensive conditions leading to generating of a trap. The task of *configuration* is fulfilled in a simple, nevertheless adequate manner. Via the request/response mechanism it is possible to manipulate the settings of a network device or a network service. In SNMPv1 and SNMPv2c the available *security* aspects are not enough to meet modern needs. Although security is massively increased by using SNMPv3, there are some voices criticizing the low *efficiency* of this protocol version. Research in this area has shown that this does not count as a general argument. If SNMP applications are well-designed the efficiency is acceptable as has been shown in [9]. Due to its simplicity, its little overhead and the existing abbreviations like the bulk operations, SNMPv2c is efficient. The most interesting requirement for network management tools is the one of *flexibility* and it leads to a central statement about SNMP. Out of one superficial point of view flexibility is definitely covered due to the fact that SNMP only offers basis functionalities and makes it possible to combine them to more complex ones on a higher level if needed. Thereby it achieves one main design goal of the protocol which was simplicity of the agents. They have to support only a non-complex functional range. This simplicity results in a broad support by many network entities. In comparison to other protocols SNMP has the widest spreading in regular computer network management. Out of another, closer point of view SNMP is not flexible. The very central aspect is that it lacks an easy possibility to integrate SNMP information and functionality into other applications. The mentioned BER encoding is not useful for achieving or importing such messages by other applications. A more universal used format has to be introduced. Furthermore it is considerably difficult to manage the bypassing of firewalls with SNMP. For this purpose complex firewall rules have to be defined.

To summarize our results of analyzing SNMP and its advantages as well as its disadvantages, it can be stated that SNMP is suitable regarding lower-level functionality for configuration and monitoring network devices and services. However, it lacks flexibility on higher levels. The central shortcoming is the missing possibility for an easy integration into arbitrary other applications. For exactly this central goal Web technologies are very useful and have been taken by us to implement our NWM solution. The general idea behind this approach is to build a gateway that offers Web service based access to SNMP infrastructure. But before getting into more technical details of these gateways, a short survey of Web technologies will be offered to provide a fundamental understanding about this technological field and to name all important terms used beneath.

4. Basic Web Technologies

Web technologies describe a set of technologies based on the Hypertext Transfer Protocol (HTTP), on the principle of Uniform Resource Identifiers (URIs) and on a content description format which nowadays is the eXtensible Markup Language (XML). HTTP is a protocol based on a request-response mechanism. A Web client opens a TCP connection to a Web server and requests a resource located on this server with a HTTP *GET* request. The server answers with a response that includes (if available) the resource. Beside fetching resources with a GET request, HTTP supports further operations like uploading a representation of a resource to the server with the *PUT* request, deleting a resource with a *DELETE* request or send data to be processed to the server via the *POST* request. Resources that are associated with each request are addressed by an URI. Often resources and data available in the Web is represented in a XML format. XML is a general-purpose markup language that offers a way to exchange data in a tree-structured way. XML nodes are called elements and can be enriched by attributes. Associated with these technologies supporting ones have been evolved. The Document Object Model (DOM) [17] is a special API that enables easy navigation through XML trees. For example, it is possible to access XML elements by providing their name or access child and parent nodes or the attributes of an element. The DOM is extended by a event specification thus making it possible for elements to be source of events. The API is offered by multiple JavaScript engine (an engine to execute logic on client-side) implementations available in current Web browser. To address parts of an XML document, XPath can be used. It was introduced in order to specify a special entity such as a node or an attribute. It provides a powerful filtering mechanism to e.g. pick out all elements with a special name. The eXtensible Stylesheet Language Tranformation (XSLT) has been defined to provide a unified way of realizing transformations between different XML formats. An XSLT processor uses an XML document and a stylesheet as input. The stylesheet contains rules that define the mapping of XML elements to an other format. HTTP has been enriched with security aspects. HTTP Secure [10] is a protocol that provides encryption and authentication and uses the Secure Socket Layer (SSL). Owing to the facts that HTTP is a simple request-response protocol and that communication has to be initiated by the Web client, transmitting information asynchronously to the client is not natively supported. For realizing pseudo-asynchronous communication, a further Web API has been defined – the XMLHttpRequest API [18]. It makes it possible to poll the Web server in an efficient manner to request up-to-date information. Furthermore a real asynchronous form of communication is introduced by HTTP Streaming. This is realized by opening a connection from client to server over which new and up-to-date date can be sent from server to client, if necessary. HTTP Streaming is also called HTTP push mechanism. Based on the API extensions a real bidirectional pull- and push- based communication is supported.

Out of the Web the concept of Web services has been developed. In general, a service is a encapsulated functionality which is accessible via a well-defined interface. Such a service becomes a Web service if it is available in the Web. Accessing services through HTTP solves many problems with firewalls due to the fact that they are often configured in a way to pass HTTP traffic. To invoke Web services in a unified way the SOAP [16] protocol has been defined. It is a quite complex and powerful XML protocol. There exist a number of standards that are associated with SOAP and that extends its functionality. An example is the WS-Security specification, which defines means for encryption and authentication. Due to the fact that the SOAP-based sort of Web services may be inefficient and that the powerful tools they provide are often not needed, a lightweight Web service architecture is often used particularly in the area of regular Web applications, the Representational State Transfer (REST) [11]. Instead of defining an extra layer like the SOAP protocol does in order to invoke functionality on server-side, a REST architecture uses the methods offered by HTTP. The central principle is that it only operates on resources by using a CRUD (Create, Read, Update, Delete) principle. There are no method calls or something comparable to it on server-side. The only thing that is done is that representations of resources addressed via URI are created, exchanged, deleted and updated by using the named HTTP methods. This way an application architecture with a generic interface is achieved, that is ,firstly, lightweight and, secondly, enables easy integration into further applications.

5. Web-based Network Management

As described before, the SNMP-based NWM within the CANDY Framework is used as a lot of network devices and services nowadays support SNMP and as SNMP covers the main requirements on lower levels. To make a total integration of the NWM solution possible, Web gateways are used. However, this idea is not brand-new but was never included in that overall context, as it has been included within the CANDY Framework. Furthermore an improvement to existing solutions will be presented below. Before this the general approaches of Web service based NWM are described. Different application areas of Web technologies within NWM are provided. A first possibility to include Web technologies to network management is to embed a Web server into a network device for element management. This approach is called *Embedded Web Server* (EWS). As a data representation format used on top of HTTP XML often occurs. An advanced implementation is presented in [12]. The main criticism regarding this approach focuses on the fact that there exists no accepted standard that is implemented by the majority of manageable entities and due to its complexity this standardization is unlikely. Furthermore, it introduces a new data model that is as well not wide-spread. Beside this, many regular and wide-spread NWM tools are not able to access a EWS. A further disadvantage is that a trap based communication from devices to the manager makes it necessary in the given implementation to deploy a Web server inside the manager thus making it accessible via HTTP. As pointed out in [13] there even exists the approach to introduce a gateway for a EWS that translates its functionality to SNMP, thus making it accessible by a SNMP manager. Even if this enables the use of already available SNMP tools, the above mentioned criticism of a new underlying data model and non-standardization stays the same. Furthermore the provision of SNMP as a interface to the world outside is no good choice due to the lack of its ability for integration as has been described before. A third possibility for a gateway is to translate SNMP NWM to a HTTP-based interface. The three named architectures are summarized in Figure 7.

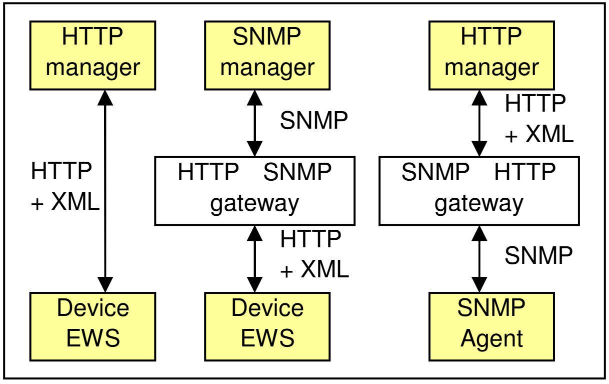

Fig.7. HTTP usage in NWM

Due to the advantages of the last form of architecture now a description of existing approaches to translate SNMP functionalities to a HTTP-based interface is summarized. There exist many technical variations used to realize the gateway application. Three common ones are pointed out in [13]. At first an approach is described that maps the MIB to an XML representation, and by this making it possible to use the DOM interface to access the XML format. Each DOM call is then translated to a SNMP request that is directed to the proper OID. Even though this approach focuses on common Web standards, the way to access the SNMP agent leads to potential complex sequences of DOM API calls. A further possibility is to use a direct translation of HTTP methods to SNMP operations. As pointed out in [13] one way of realizing this is to extend URIs with particularly XPath to indicate a target object. Again the operations are directed to an XML document and are than mapped to the underlying SNMP data structures. This approach is exceptionally easy to implement and mainly relies on HTTP methods though the implementation can be made more simple and thus more efficient as shown below. The last implementation presented in [13] uses the SOAP protocol. Though it is not as lightweight as the first two approaches are and results in protocol overhead, it is much more powerful. The high level of standardization and of acceptance in the distributed system community makes

SOAP a perfect candidate for accessing SNMP-based and accordingly legacy infrastructure. With its extensions, particularly its security extensions, it makes it possible to access this infrastructure in an encrypted and authenticated way through the Internet. It does not only enable the integration of regular NWM infrastructure but also the integration of regular SNMP tools which offer higher level functionality such as graphical representations and summarizing reports of the network status. SNMP tools, like for example OPENNMS [14] or RRDTool [15] that offer various graphical representations and visualizations of the SNMP managed network, can be accessed by using SOAP-based Web services. As a unifying format for this data NDML is used. This possibility has been analyzed within CANDY. A possible architecture for translation is given in Figure 8. By this presented architecture integrating available functionality of many existing NWM tools is possible, leading to the avoidance of reinventing the wheel. Within CANDY a similar architecture to offer the initially mentioned fault report for improving the management structure during the redesign workflow is used. Instead of a legacy tool, a network analysis process that itself uses different NWM tools to get various status information of the network has been introduced. This data is analogically made available via transformation to NDML and the SOAP-based interface for integrating it into the redesign workflow. The given solution for a powerful Web service based network management enables a wide-range of integration thus bringing the desired flexibility to the underlying SNMP infrastructure.

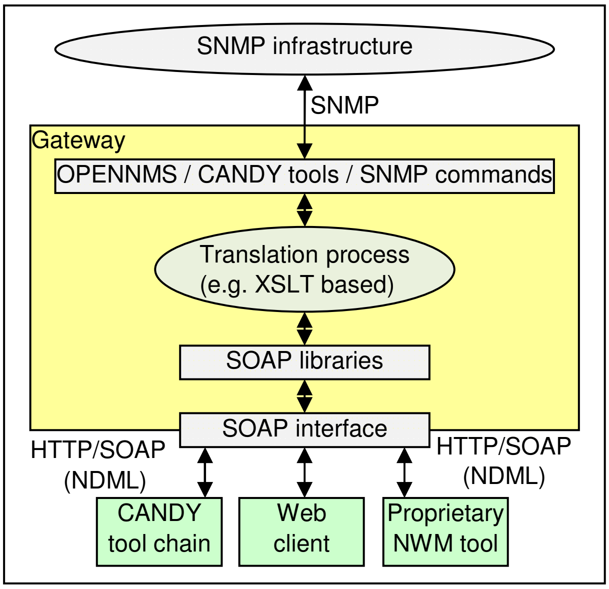

Fig.8. NWM tool functionality access via SOAP

By the achieved possibility of integrating different tools a very flexible and platform-independent solution has been produced. Instead of monolithic systems that are platform dependent in fields like NWM database usage or even in fields like hardware usage, the CANDY project offers a loosely coupled and extendible architecture. This is possible due to the application of open standards and the ability of integration offered by NDML. Although the architecture achieves our goals, a very central shortcoming has to be pointed out: It may result in inefficiency and in some cases is over dimensioned. For example, if a Web client only wants to change the values of some OIDs it is obvious that the indirection through SOAP is not necessary. In general a simpler approach is desired. As mentioned above the two further and non-complex given possibilities – the DOM- and directly HTTP-based ones – have disadvantages that have to be eliminated by this new implementation. In the next section our solution that is offered as an alternative to the SOAP-based approach is described.

6. REST-based SNMP Gateway

Though the idea that leads to the new implementation is simple, it is innovative in the area of Web service based NWM. To summarize the concept, nothing else has been done than applying the REST principle to the SNMP gateway. The MIB structure is seen as a set of logical documents that put together form a further document. Additional technologies like XPath etc. do not have to be applied in this case. URIs are used to access OIDs in a SNMP agent. The URIs are accessed with the regular HTTP methods that have the following semantics:

- HTTP GET: Retrieve the current representation of the value addressed by a given URL formulation of an OID
- HTTP PUT: Update a value addressed by an OID thus enabling the possibility for configuration
- HTTP DELETE: Set a value addressed by an OID to zero or null

The Web server of the gateway is configured in a way that it takes requests that are addressed to an arbitrary resource located on it and maps it to the right OID. The URI representation uses a slash instead of a dot to separate the single elements of the OID. *Simple URIs* are formed after the scheme

```
http://[gateway_address]/[agent_address]/
[community_string]/[SNMP_version]/
[slash_separated_OID]
```

If a Web client wants to receive the value of a description of the device with the IP 176.20.30.21 based on SNMPv2c with the community string “com” it sends a HTTP GET request to the URI *http://gateway.com/176.20.30.21/com/v2c/1/3/6/1/2/1/1/5*

There result three questions:

1. How is the exchanged data represented?
2. How are more complex operations (e.g. BULK operations) realized?
3. How are traps transmitted to Web clients?

For representing values requested by a Web client, a simple XML representation has been introduced. It builds a conceptional extension of the NDML and was formulated to create as little overhead as possible. The gateway encapsulates responses into this XML format and extracts data (e.g. new value of a managed object) that is transmitted inside the request and uses it as input for the regular SNMP tool chain used to access the SNMP infrastructure. Complex operations are realized in the same URI-based manner as simple ones. To fulfill the demands of the REST principle, the URIs are enriched with a subcomponent indicating a complex resource to which the request is directed. In our terminology this extends the *simple URIs* to *complex* ones. A complex resource just means that more than one OID is addressed by the operation. For example, by including the String “range” into the URI behind the SNMP version value two OIDs can be specified – a start OID and a end OID separated by a special character. If a GET request is sent to such a range all values of the information between these two OIDs are responded. By this concept further higher level semantics can be introduced, such as bulk transfer. In the field of transmission of traps our implementation also brings an improvement of the above discussed architectures. Modern HTTP Streaming mechanisms to communicate traps to Web clients that have registered for these requests are used. For this the Web server located at the gateway is extended with a HTTP Streaming component. If a trap occurs, it is detected as an event and transmitted to all currently registered Web clients. As a sample client a “Web 2.0” (modern Java-Script-/XMLHttpRequest-based Web application) has been created to enable a straight forward configuration and monitoring of network devices and services. By this Web application all goals regarding the dimension of *presentation* were fulfilled. The administrator interface has been designed well arranged and easy to use. As a design alternative instead of HTTP HTTPS can be used to realize encryption and authentication of the communication between Web client and gateway. By this the functionality can be offered in the Internet in a secure manner.

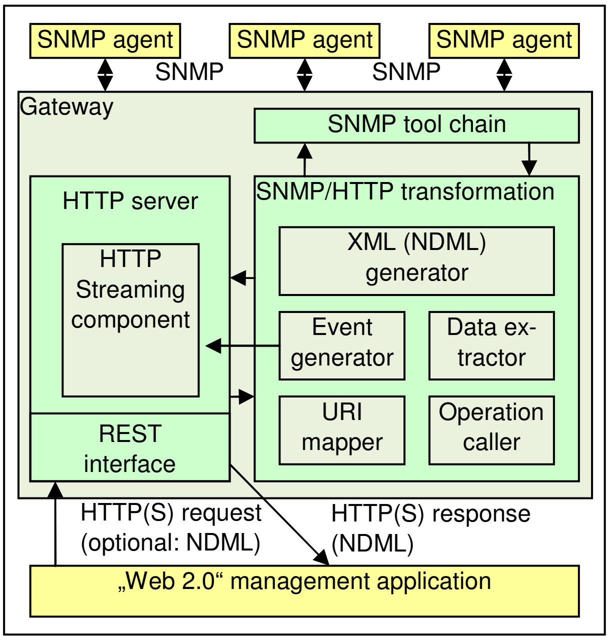

Fig.9. REST-based NWM architecture

The evaluation of the basically described architecture has shown that it is very efficient. To give a meaningful example for this efficiency, a complete SNMP walk through a given MIB (device: LaserJet 4100) has been realized on the basis of command line SNMP tools and based on our implementation. This experiment has shown that, although the results of the SNMP link between SNMP agent and gateway are evaluated and encapsulated into an XML representation and access is based on HTTP, the end-to-end communication is only slowed down by a factor of 1.17. If the walkthrough is realized by SNMP command line tools without an intermediate gateway, it takes 5.367 seconds, in the REST-based implementation it takes 6.289 seconds. Figure 9 summarized schematically the REST-based architecture.

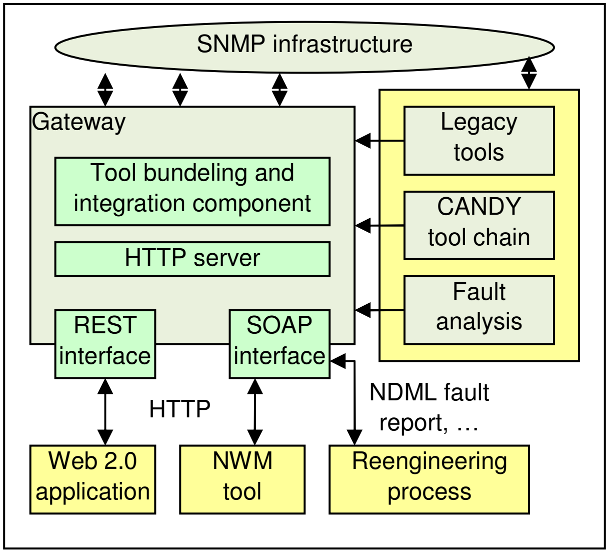

Fig.10. Web service based management solution

7. Conclusion and Future Work

This paper has presented an argumentation to use a hierarchical and layered structured NWM solution to meet the needs of modern NWM tools. The suggested system is based on SNMP as a low-level protocol which on one hand offers all necessary basic functionality and on the other hand is very wide-spread in practice with a broad device and tool support. The two central disadvantages of SNMP have been identified as missing security concepts in version 1 and 2 of the protocol and as a lacking of easy integration into other tools and into 3rd party software. These shortcomings have been eliminated by our Web-based approach that uses a gateway to offer a SOAP-based and a REST-based interface to NWM functionality. SOAP was chosen due to its powerful functional range and its security extensions that make access from the Internet to a local network possible in a secure way. Furthermore, an integration of (open source) legacy tools and there functionality is enabled by the presented architecture. By this the reinvention of the wheel is avoided. A very flexible, non-monolithic and extendible architecture has been achieved that fits the philosophy behind the CANDY Framework. Due to the fact that in some cases the SOAP-based approach is oversized and too powerful, a lightweight, REST-based architecture has been provided as an alternative. In both subsystems NDML has been introduced as a glue component. It is used for representing the exchanged information. The overall network management solution that is used when designing networks with the help of the CANDY Framework is presented in Figure 10. It shows that on the one hand the two offered interfaces can be used during regular network management, but on the other hand that the SOAP-based interface is used to support the above mentioned reengineering workflow. In this paper it has been proven that loose integration of tools is possible by using modern Web services and “Web 2.0” paradigms. This approach is applied to integrate tools that analyze network management information or network traffic and generate a fault report formulated in NDML. This functionality is deployed for improvements of overall quality parameters of designed networks.

It is planned for the future to research the profitableness provided by directory services that for example may be used to detect the location of Web gateways. These directories may keep information about the functionalities offered by SNMP tools that are bundled by a gateway. So far first experiments have been made with directory service (an UDDI-based service) support, but this research area offers much more tasks and possibilities. Furthermore, the REST-based application will be extended by further abbreviations to achieve an increase of efficiency. Beside this, our goal is to extend our solution by a complex-rule monitoring engine. For this goal different ways of implementing this engine will be analyzed in the next weeks.

References

1. Luntovskyy, A., Gütter, D., Schill, A., Winkler, U.: “Concept of an integrated environment for network design”, IEEE CriMiCo Conference, Sevastopol, Sept. 2005, pp. 959-961; ISBN966-7968-79-0
2. Luntovskyy A., Guetter, D., Pfeifer, G., Schill, A.: "Network Design Methodology and Workflow within the CANDY Framework", 14th International Multi-Conference ACS-AISBIS 2007, Szczecin, Poland, 17-19 Oct. 2007 (published in Polish Journal of Environmental Studies)
3. Luntovskyy, A., Trofimova, T., Trofimova N., Guetter, D., Schill, A.: “To a Proposal towards Standardization of Network Design Markup Language”, International Network Optimization Conference 2007 (INOC 2007), Spa, Belgium, 22-25 April 2007, p.54
4. Case, J., Fedor, M., Schoffstall, M., Davin, J.: "A Simple Network Management Protocol (SNMP)", Request for Comments 1157, May 1990
5. Case, J., McCloghrie, K., Rose, M., Waldbusser, S.: "Introduction to Community-based SNMPv2", Request for Comments 1901, January 1996
6. Harrington, D., Presuhn, R., Wijnen, B.: "Simple Network Management Protocol (SNMP) Management Frameworks", Request for Comments 3411, December 2002
7. Blumenthal, U., Wijnen, B.: "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", Request for Comments 3414, December 2002
8. Schoenwaelder, J.: "Simple Network Management Protocol (SNMP) over Transmission Control Protocol (TCP) Transport Mapping", Request for Comments 3430, December 2002
9. Hia, H., Midkiff, S.: “Deploying Secure SNMP in Low Data Rate Networks”, Office of Naval Research, Navy Collaborative Integrated Information Technology Initiative (NAVCIITI), July 2000
10. Rescorla, E.: "HTTP Over TLS", Request for Comments 2818, May 2000
11. Fielding, R.: "Architectural Styles and the Design of Network-based Software Architectures", Dissertation, University of California, Irvine, 2000
12. Ju, H., Choi, M., Han, et. al.: "An Embedded Web Server Architecture for XMLbased Network Management", Proc. IEEE/IFIP Network Operations and Management Symp., Florence, Apr. 2002, pp.1-14.
13. Oh, Y., Ju, H., Choi, M., Hong, J.: "Interaction Translation Methods for XML/SNMP Gateway", Proc. DSOM 2002, Montreal, October 2002, pp. 5465
14. OPENNMS, http://www.opennms.org
15. RRDTool, http://oss.oetiker.ch/rrdtool/
16. W3C SOAP Recommendation, April 2007, http://www.w3.org/TR/soap/
17. W3C DOM Level 2 Core Recommendation, November 2000, http://www.w3.org/TR/DOM-Level-2-Core/
18. W3C XMLHttpRequest Object Working Draft, October 2007, http://www.w3.org/TR/XMLHttpRequest/

Glossary

BER	Basic Encoding Rules, rules for encoding data that is described using the ASN.1 format
CANDY	Computer-Aided Network Design utility, Java and XML-based integrated network design environment, developers: CANDY@TUD initiative
CLNS	Connectionless Network Service
CRUD	Operation principle based on methods “Create, Read, Update, Delete”
DOM	Document Object Model API
EWS	Embedded Web Server
NWM	Network Management
HTTPS	Secure variant of HTTP (encryption + authentication support)
IPX	Internetwork Packet Exchange
OID	Object Identifier for referring MIB
MIB	Management Information Base
NDML	XML-based Network Design Markup Language
Overall network quality	A set of technical and economical characteristics of networks (QoS)
PDU	SNMP Protocol Data Unit
REST	Application architecture based on a generic interface and a stateless communication protocol
SNMP	Describes three versions and further sub-versions of the Simple Network Management Protocol
SOAP	XML-based format for Web service communication
UDDI	Universal Description Discovery and Integration, Web service standard for service directories
URI	Uniform Resource Identifier, addresses Web resources
XMLHttpRequest	Object for enabling asynchronous calls from browser to server (available via JavaScript engine)
XSLT	eXtensible Stylesheet Language for tranforming XML documents