Semantics for Hybrid Networks Using the Network Description Language

Jeroen van der Ham* Paola Grosso
Cees de Laat
System- and Network Engineering Research Group,
Universiteit van Amsterdam

May 3, 2006

Abstract

Hybrid networks offer end users a mix of traditional connections and new optical services in the form of dedicated lightpaths. These must be requested in advance and are currently configured on demand by the operators. Because lightpaths are circuit switched, the user must be aware of the topology and of the techniques involved in the provisioning. Once connected, they offer a high-speed, low-level connection to the requested destination.

We present here an ontology based on RDF, called the Network Description Language (NDL). It aims to provide an abstraction layer for networks, specifically hybrid networks. NDL defines a common semantics for the exchange of information between the application, the service provider, and the domains that are part of a global hybrid network. NDL can be used to create inter-domain network graphs at various abstraction levels, to provide an information model for service discovery, and to facilitate lightpath provisioning.

1 Introduction

Currently many research communities show the need for dedicated end-to-end connections, see for example [1, 2, 3]. Private network paths allow the scientists to quickly move large data sets, or have guaranteed and pre-determined network service parameters to transport (a constant) data flow. *Hybrid networks* are a way to solve this need: the regular internet IP-based traffic is separated from these specialised applications. We describe this new kind of infrastructures further in section 1.1.

In hybrid networks it is typically the end user who requests the dedicated circuit. This creates the complication that the end-users must know about the techniques involved and be able to make an informed decision about the network path in order to get the required behaviour. The provisioning of this path requires a common understanding of the underlying topology by the requesting

*Corresponding author: vdham@science.uva.nl

application and the provider of the service. A common semantic forms the basis for automated information exchange.

We developed NDL, the Network Description Language, to define this vocabulary. NDL is based on Resource Description Framework and therefore relies on the idea of the Semantic Web to make information easily exchangeable among interested parties, with a web of traversable information. The Semantic Web and NDL are explained in sections 2 and 4 respectively.

This article is organised as follows: the rest of section 1 introduces hybrid networks and describes GLIF, the Global Lambda Integrated Facility, where most of our work sees its current application. Section 2 describes the general concept of the semantic web and the RDF language. Section 3 discusses related work. In section 4 we introduce NDL, and in section 5 we describe applications of NDL. Section 6 summarises the article.

1.1 Hybrid networks

A *hybrid network* consists of two logical components: an IP-based part and an optical part. These components coexist in the same administrative domain and often share the same physical infrastructure. The IP component provides the traditional routed services to end-users. The optical component targets applications that require guaranteed services over dedicated circuits.

In the IP-based part the routing protocols adapt their understanding of the topology to the changes caused by equipment and link failures. When the communication begins, packets follow a path from source to destination that is valid at that specific time. In most cases, the user does not have any guarantee that the conditions of the communications remain the same over time. Delays and bandwidth can change at any moment, caused by concurrent traffic sharing (parts of) the same link, route changes, or other factors.

In the optical part of the hybrid network, automatic configuration of links is currently not possible. The link must be requested well before it can be used. Once provisioned, the path between the end-points is fixed. The advantage of an optical link is a guaranteed service for each communication, with predetermined and reproducible characteristics, such as delay or bandwidth. These characteristics are determined by the components that form the path through the optical network.

Dedicated optical circuits (*lightpaths*) can run over wavelengths (*lambdas*) available in the optical part of a hybrid network. Lightpaths are assigned to applications or users for their data traffic, creating a lambda network, where the lightpaths can not interfere with each other.

Lightpaths are ideal to suit the needs of many modern data-heavy scientific applications. The scientific applications that most benefit from dedicated optical circuits are high bandwidth or delay sensitive applications, such as the ones that transfer large data sets between high energy physics laboratories or astronomical instruments, or applications that perform remote visualisation of medical or biological data.

In the past years academic research networks have started to build hybrid infrastructure. There are several examples of such networks at various stages of maturity, from test-bed to production quality. One such example is SURFnet6[4], the research network in The Netherlands, which consists of a backbone based on the hybrid paradigm: the routed Internet traffic uses one of the lambdas, while the other lambdas are available for on-demand circuits for scientists and applications.

1.2 Global Lambda Integrated Facility

The Global Lambda Integrated Facility[5] (GLIF) is an international collaboration, whose goal is to promote co-operation in the field of lambda networking. In GLIF, national research and education networks, institutions and consortia provide their technical expertise and share their research results. The goal of the co-operation is to integrate the individual (hybrid) networks into an international facility for end-users all over the world. Figure 1 shows the current network map of GLIF with all wavelengths currently available, spanning the entire globe.

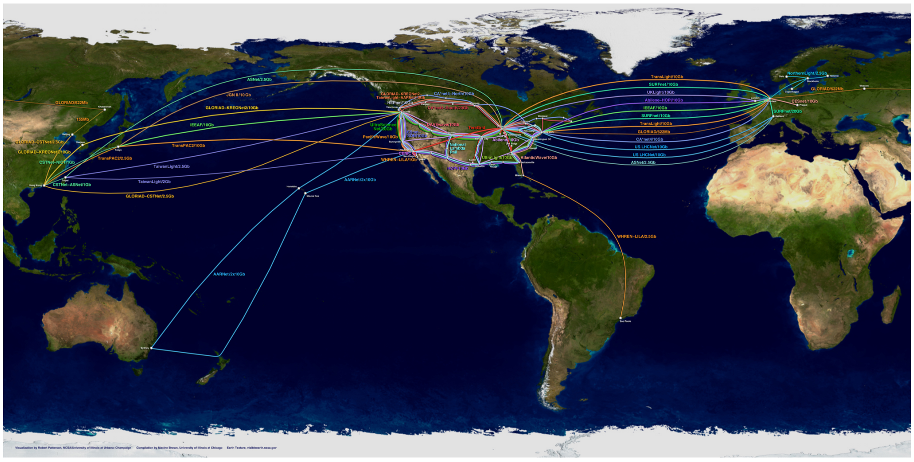

Figure 1: GLIF global infrastructure

The biggest challenge in GLIF, and in general in any hybrid network, is to create the basis for an effective communication between the end user application and the service provider. Ultimately an end-user should be able to request and control the lightpath setup, without direct intervention of the service provider. This kind of automated lightpath setup requires tight cooperation of the different management and control planes. To attain this level of cooperation, a common vocabulary is required. The GLIF community has identified two basic areas in need of a common vocabulary: service and topology definitions:

- A *service* is verifiable, reproducible, repeatable and end-to-end. A request from an application should be both simple and clear, so that both at the intra-domain and inter-domain level all interested parties can validate whether they can or cannot fulfil the request.

- *Topology information* is public, has various abstraction levels, and can be navigated. This information is available to end-users and other domains, at varying degrees of abstraction depending on the usage scenario and the authorisation policies. Any time an application makes a request it provides the necessary input parameters based on knowledge of the topology. The provider can then easily validate the feasibility of the requested lightpath, because it shares the same view of the underlying network.

The control plane group in GLIF is currently working on defining services using the Common Service Definition[6]. For the topology description within GLIF we propose NDL. It satisfies the above topology requirements by using varying abstraction levels, and leverages the software tools available for RDF to make the parsing, validation and integration of the information straightforward.

2 Introduction to Semantic Web and RDF

In this section we explain a way of defining vocabularies using the Semantic Web and RDF. In section 4 we explain how we apply RDF to define a topology for networks.

In order for two computer programs to communicate there must be a common understanding about the vocabulary being used. Currently most communication by computer programs is fixed by protocols. The form of the interaction is fixed, but the meaning of the data being exchanged is not.

Take a web-browser for example, when a user types in a URL, the web-browser starts communicating with the designated server, asking for the URL ('GET'). The server then answers the browser with the designated file. This interaction is strictly defined in RFC 2616[7]. But neither the web-browser nor the server know what kind of data is being exchanged, it could be about the weather, traffic information, etc.

Because the applications do not grasp the meaning of the data being presented, it limits the possibilities to mere presentation. The Semantic Web idea originated as a solution to this problem; it tries to make it easier for computers to understand the meaning of the content they present, so that they can navigate autonomously through this information to find what they are looking for.

To really define terms the Resource Description Framework was created. It allows for the creation of schemas, defining terms using URIs (URIs is a superset of URLs) in an XML syntax. These are used to make the terminology unique, so that when used, the sender implicitly signals that it understands the meaning of the terminology.

An example of using semantics with data is the Friend of a Friend project[8]. Participants of this project describe themselves, giving their name, homepage, place of work, etc. The properties are predefined to make sure there is no conflict with e.g. using *'last name'* versus *'surname'*. But definitions of terms is not enough for computers: is *'surname'* the same as *'Surname'*? An example of a FOAF description is given below.

```
<foaf:Person rdf:nodeID="me">
    <foaf:firstName>Jeroen</foaf:firstName>
    <foaf:surname>van der Ham</foaf:surname>
    <foaf:workplaceHomepage
        rdf:resource="http://www.science.uva.nl/~vdham/"/>
</foaf:Person>
```

The document contains information about the first author of this article. All the tags use namespaces for readability. They are either in the RDF namespace (basic types) or in the FOAF namespace, to identify the meaning of these properties. The `#` is a shorthand for the local namespace and evaluates to the URL of the document.

Each statement of the document above in RDF is in essence a triplet of the form *(subject, property, object)*. The subject in each case is the first author, the property is the name of the tag, `firstName`, `surname`, et-cetera. Using these triplets, the document essentially describes a graph, with subjects and objects as vertices, and properties as edges. This graph form is the underlying format of RDF descriptions, meaning that sequence or syntax does not matter.

Another feature of RDF is to point to other descriptions. Using these kinds of links, people can describe themselves and point to their friends' description. Traversing these links, a whole network of friends appears, hence the name 'Friend of a Friend'. Applications are available to browse this network visually, see for example the Foafnaut[9].

The Resource Description Framework is a W3C recommendation[10, 11] and is a lightweight implementation of semantic web capabilities. There are numerous libraries available for all major programming languages, making it very easy to develop new applications for it. All implemented libraries make simple data access straightforward. For more complicated operations on the data, there is SPARQL[12], an SQL-like query language for RDF, which is also implemented in most RDF libraries.

3 Related Work

In this section we discuss related work regarding the description networks and their elements. One of the first standard ways to gather information from network elements was by using SNMP[13]. Diagnostic, performance and configuration information of network devices can be retrieved from the MIB (Management Information Base). The MIB is a tree of name, value pairs, which can be requested and changed. The model provided by SNMP is centred on the device itself and does not contain much references to other devices. And although a large part of the MIB is standardised, most vendors use their proprietary space to store the relevant configuration data of their devices. SNMP is provided by nearly all networking devices.

The IETF is currently working to replace SNMP with a new standard, NetConf[14]. While SNMP only allows for three data-types (integer, string, sequence) and uses its own protocol, NetConf utilises other standards. It allows the use of XML for the description of data, allowing for any data-type. Netconf defines the transportation of the data and change requests over standard protocols, SSH, BEEP or SOAP. NetConf has not yet been implemented in networking devices.

Another important network device information model is the Common Information Model, CIM[15]. It is being developed by the Distributed Management Task Force (DMTF)[16] and it is an object-oriented information model described using the Unified Modelling Language. The model captures information regarding computer systems, operating systems, networks and other related diagnostic information. CIM is mainly implemented in consumer-oriented computing equipment, and operating systems such as Windows and Solaris.

The Data Center Markup Language (DCML)[17] is an information model being developed by OASIS[18]. As the name states, it is aimed at describing

Data Centers, including computing equipment, networks and other hardware at these sites. DCML is also using RDF for their model. The DCML model has not been published yet, but appears to be similar in complexity to the CIM model.

All the information models related to networking equipment aim at diagnostic information and information about the device itself. Because the models are aimed at diagnostics, the schemas aim to be specific and complete. Getting a rough overview of the network is only available through gathering information from different devices (possibly using different information models).

The Network Description Language described in the next section complements the above information models. It defines a simple schema to describe the network and the relation between the different devices. This kind of description can be used by any application that needs only an overview of the network and not all diagnostic information.

4 Network Description Language

In this section we introduce the Network Description Language (NDL), which is based on RDF. We explain the different components of the schema and show examples of NDL in use.

NDL provides a description of a network, just like the FOAF project, however, NDL is aimed at hybrid networks, and computer networks in general. While computer networks and social networks are very different kinds of networks, defining the objects and the relations between them can be done in a very similar fashion. NDL provides just that, a description of the elements in networks and the links between them.

Initial development of NDL focused on defining the semantic needed for a hybrid network overview and its physical setup[19]. Further work has focused on the information needed specifically for the lightpath provisioning at the inter-domain level. The conclusion of our current work is that lightpath provisioning can be facilitated by a light-weight vocabulary of just four classes and eight properties. This simple semantics is sufficient in our opinion to allow users and service providers to evaluate the feasibility of the setup at the both the intra and the inter-domain level. Figure 2 shows the new model.

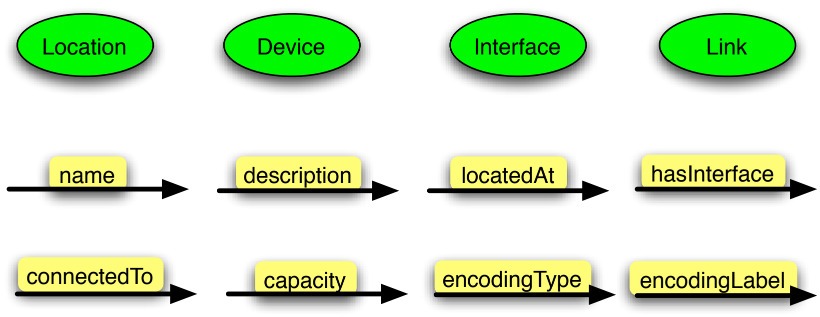

Figure 2: Overview of the Network Description Language Schema.

The classes in the figure, shown as ovals, are explained below:

- **Device** Any kind device that is connected to the network.
- **Interface** The connection between a *device* and the network.
- **Location** A place where *devices* are located.
- **Link** A complex or abstracted connection between two *interfaces*.

NDL also contains several properties, shown as arrows in figure 2:

- **hasInterface** The connection between a *device* and its *interface*.
- **locatedAt** The relationship between a *device* and the *location*.
- **name** Provides the human readable name for *devices* and *interfaces*.
- **description** A property to define extra human readable information for a device.
- **connectedTo** Physical connections between two *interfaces*, or between a *link* and an *interface*.
- **capacity** Defines the bandwidth capacity of an *interface* or *Link*.
- **encodingType** Defines the encoding type used on an *interface* or *Link*.
- **transportType** Defines the transport types supported by the *interface* or *Link*

The above classes and properties allow a fine grained description of the network, including cables, capacity and transport type, and on the other hand, they allow for more abstraction of the network, using the *Link* class. Connections between interfaces can either be defined directly using the *connectedTo* property, or by defining the intermediate *Link* object. The latter can be used to express information about the link itself, or for abstracting parts of the network that can not or will not be changed, for example if the line is leased for an extended period of time.

The values of the *capacity*, *encodingType*, and *transportType* properties are defined in a separate namespace, using the GMPLS terminology[20] (the 'LSP encoding type' and 'General Protocol ID').

Current NDL implementation addresses inter-domain path provisioning because it covers both the issue of defining the necessary abstractions and of helping the exchange of information between domains. The *Link* class allows domains to present an abstracted view of their network to the outside. Using abstracted views of domains makes the planning of inter-domain connections easier: this is an important issue in global networks such as GLIF, and other production environments. The standard RDF *seeAlso* property allows domains to easily point to descriptions of other domains and

Configuring lightpaths by hand does not scale, and in the GLIF community there currently is a lot of effort going into creating tools to automatically configure lightpaths such as DRAC[21], UCLP[22], or DRAGON[23]. These applications provide (web services) interfaces to end-users and applications to request provisioning of lightpaths. Using policy and authentication, they then try to automatically provision that lightpath. The automatic provisioning is currently limited to intra-domain lightpaths and the topology data is stored ad hoc or in a proprietary manner. This creates interoperability problems and NDL can provide a solution for this problem.

5 NDL Applications

Using the schema described in the previous section, we applied NDL to the description of the network resources in our laboratory and other domains we customarily work with. We also experimented with a few applications that make use of these descriptions. Two of the applications are described below. Section 5.1 describes a simple network resource broker and section 5.2 shows an application for visualising NDL-based descriptions.

5.1 NDL-based Resource Brokering

Seeking the general adoption of NDL within the GLIF community we started using it within our own hybrid network to show the effectiveness for automated lightpath setup. We have access to the resources in NetherLight [24], the optical exchange at Amsterdam, The Netherlands, where several international optical circuits converge. NetherLight is one of the many domains that participate in GLIF, and it is a typical example of a lightpath exchange point where lambdas from several external locations can be ‘patched’ together to form lightpaths.

The University of Amsterdam together with SARA, the Dutch National Academic Computing Center, have built a laboratory for optical network research named LightHouse[25]. The LightHouse hosts computing clusters and network equipment and is directly connected to NetherLight. NetherLight and LightHouse together form a testbed for our experimentation, with on one side the network resources and the lightpaths, and on the other the computing nodes that constitute the terminating points of the end-to-end path.

To manage the lightpath setup in NetherLight/LightHouse we developed a simple lightpath brokering and scheduling system. Figure 3 shows the software architecture we prototyped. At the top is the Network Description Language, which is used to describe the LightHouse and Netherlight. Referring to these descriptions, an external client/application interfaces through web services with the resource broker. This resource broker internally manages:

- a *reservation database* to maintain information on current and upcoming lightpaths requests,
- a *path finding engine* to determine if and how it can satisfy a request for a lightpath,
- and a *path provisioning engine* to make the appropriate changes on the underlying network equipment that satisfy the request.

The NDL-based topology description of the LightHouse and NetherLight provides a centralised and consistent view to the external client, the web services and the resource broker. The client uses NDL to determine the entry and exit points of the desired lightpath; the web services uses NDL to validate the request from the user before passing it on to the resource broker; and the broker uses the information contained in NDL to run the path finding algorithm and determine if a suitable end-to-end path can be constructed.

Let’s look at a portion of the actual NDL description for NetherLight:

```
<ndl:Device rdf:about="#tdm4.amsterdam1.netherlight.net">
        <ndl:name>tdm4.amsterdam1.netherlight.net</ndl:name>
```

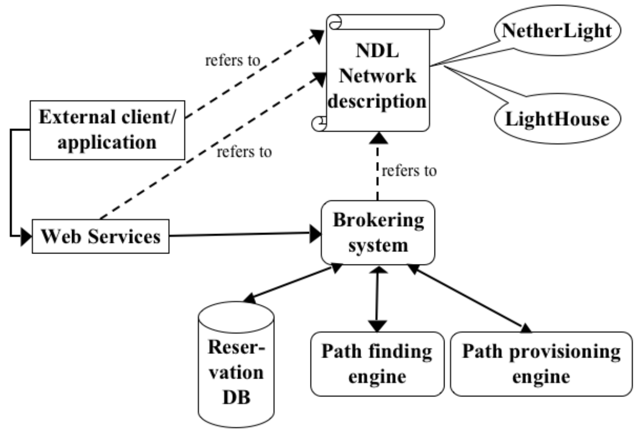

Figure 3: Lightpath brokering and scheduling system architecture

```
        <ndl:locatedAt rdf:resource="#amsterdam1.netherlight.net"/>
        <ndl:hasInterface rdf:resource="#tdm4.amsterdam1.netherlight.net:5/1"/>
</ndl:Device>
<ndl:Interface rdf:about="#tdm4.amsterdam1.netherlight.net:5/1">
        <ndl:name>tdm4.amsterdam1.netherlight.net:POS5/1</ndl:name>
        <ndl:connectedTo rdf:resource="#tdm3.amsterdam1.netherlight.net:501/1"/>
        <capacity rdf:resource="&net;OC192"/>
</ndl:Interface>
```

The RDF description above states that a device called `tdm4.amsterdam1.netherlight.net`, located at NetherLight has one interface, interface `5/1`; this interface has a (SONET) capacity of `OC192` (10 Gbit/s) and connects to interface `501/1` on a device called `tdm3.amsterdam1.netherlight.net`. In the first instance a client parsing the above code can decide to request a lightpath between `tdm3` and `tdm4`, with a guaranteed throughput of 1Gbit/s. It expects the request to be satisfied because it knows from the NDL description that a connection between the two devices exists.

What the client cannot know is whether there is enough capacity to satisfy the bandwidth requirement. The resource broker parses the same NDL and reaches the same conclusion: there is a path between the two resources. The broker verifies in its database if there are other users that have already been assigned a portion of the *OC192* to see if there is enough bandwidth available to satisfy the request.

A more complex scenario where the data integration feature of NDL come into play, is the one in which a client decide to ask for a lightpath between devices that are not directly connected to each other. In this case the resource broker needs to check if it can build a path between them of the appropriate capacity, and potentially has to parse all of the NDL files for the domains the path between the end-points runs through.

5.2 NDL-based Network Maps

A common description of a hybrid network is essential not only for brokering and scheduling of resources, but also for a unified visual representation of the different domains. Comparing infrastructure maps of several hybrid networks requires a common representation format. The information contained in NDL files is suitable to produce maps. Domains that use NDL to describe their topology can build consistent and comparable set of maps.

The GLIF participants are already gathering and publishing information about the exchange points in their respective network. Currently this information is published in hand-made figures and web-pages. Once collected, the descriptions of the exchange points can easily be converted to NDL description, allowing for automatic creation of comparable visualisations. This exercise allows us to benchmark NDL to see whether we have included all the necessary classes and properties to describe the topology of this global network.

To produce network maps we parse the information contained in the NDL files and extract information using SPARQL. We then use GraphViz[26], an open source graph visualisation implementation, to represent structural information as graphs. Figure 4 shows the representation of connections among a few of the optical devices present in NetherLight.

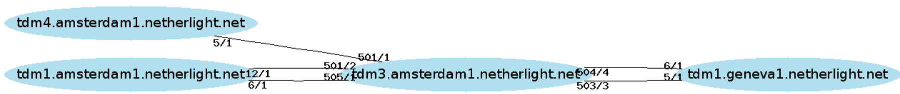

Figure 4: Graphs of NetherLight resources (extracted from NDL file)

The power of these graphs is that they can be automatically generated every time the NDL file for a domain changes. The graphs can integrate multiple domains in a single plot, a feature that is particularly useful when seeking the interconnection between hybrid networks.

6 Conclusion

In this article we have explained that when applications request a path through a hybrid networks, they must be aware of the topology. The awareness of the topology and the communication with the services provider requires a common vocabulary.

We propose the Network Description Language (NDL) as this common vocabulary. NDL is based on the RDF standard, a Semantic Web technique. As such we can leverage a myriad of commercial and open-source tools available for RDF and XML to create, parse and validate NDL.

The NDL vocabulary is small and simple by design, in order to focus on the terms needed for path search and provisioning. While the vocabulary is small, it is already very powerful, and we have shown the deployment in applications that perform resource brokering and network visualisation.

We believe NDL provides a simple way to reduce the inherent complexity of hybrid networks. It can also be applied to other type of networks, allowing applications to easily unlock the networks' possibilities.

In the future we plan to integrate this research in automatic provisioning tools. This will also allow us to test our model in a more practical situation and see if we must change or extend it further. We are also planning to extend NDL to be able to describe higher layers of the network.

7 Acknowledgements

This research was financially supported by SURFnet and TNO. The authors would like to thank Franco Travostino from Nortel Networks, Ronald van der Pol and Andree Toonk from SARA, and Freek Dijkstra and Bert Andree of the System and Network Engineering group of the University of Amsterdam for their constructive input and discussions on this research.

References

[1] LHC Computing Grid Optical Private Network.
URL http://lcg.web.cern.ch/LCG/activities/networking/nw-grp.html

[2] Homepage of the european very long baseline interferometry network.
URL http://www.evlbi.org/

[3] iGrid 2005.
URL http://www.igrid2005.org/

[4] SURFnet, SURFnet6 lightpaths mark start of new Internet era (press release).
URL http://www.surfnet.nl/info/en/artikel_content.jsp?objectnumber=107197

[5] Global Lambda Integrated Facility (GLIF).
URL http://www.glif.is/

[6] J. Sobiesky, T. Lehman, Common service definition, Tech. rep., Mid-Atlantic Crossroads (MAX) (2005).
URL http://dragon.maxgigapop.net/twiki/bin/view/DRAGON/CommonServiceDefinition

[7] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, Hypertext Transfer Protocol – HTTP/1.1, RFC 2616 (Draft Standard), updated by RFC 2817 (Jun. 1999).
URL http://www.ietf.org/rfc/rfc2616.txt

[8] Friend of a Friend (FOAF) Project.
URL http://www.foaf-project.org/

[9] FOAFnaut.
URL http://www.foafnaut.org/

[10] G. Klyne, J. J. Carroll, Resource escription Framework (RDF): Concepts and abstract syntax, W3C recommendation, World Wide Web Consortium (W3C) (2004).
URL http://www.w3.org/TR/rdf-concepts/

[11] Resource Description Framework (RDF).
URL http://www.w3.org/RDF/

[12] E. Prud'hommeaux, A. Seaborne, SPARQL Query Language for RDF (2005).
URL http://www.w3.org/TR/rdf-sparql-query/

[13] J. Case, R. Mundy, D. Partain, B. Stewart, Introduction and Applicability Statements for Internet-Standard Management Framework, RFC 3410 (Informational) (Dec. 2002).
URL http://www.ietf.org/rfc/rfc3410.txt

[14] IETF, Netconf working group.
URL http://www.ops.ietf.org/netconf/

[15] DMTF, Common Information Model (CIM).
URL http://www.dmtf.org/standards/cim/

[16] Distributed Management Task Force (DMTF).
URL http://www.dmtf.org/

[17] OASIS, Data Center Markup Language (DCML).
URL http://www.dcml.org/

[18] Organization for the Advancement of Structured Information Standards (OASIS).
URL http://www.oasis-open.org/

[19] J. van der Ham, F. Dijkstra, F. Travostino, H. Andree, C. de Laat, Using RDF to describe networks, Future Generation Computer Systems.
URL http://www.science.uva.nl/~vdham/research/publications/0510-NetworkDescriptionLanguage.pdf

[20] L. Berger, Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description, RFC 3471 (Proposed Standard) (Jan. 2003).
URL http://www.ietf.org/rfc/rfc3471.txt

[21] Dynamic Resource Allocation Controller (DRAC).
URL http://www.nortel.com/drac/

[22] User-Controlled LightPaths (UCLP).
URL http://www.canarie.ca/canet4/uclp/

[23] Dynamic resource allocation via gmpls optical networks.
URL http://dragon.maxgigapop.net

[24] Netherlight.
URL http://www.netherlight.net

[25] Lighthouse laboratory.
URL http://www.science.uva.nl/research/air/projects/lighthouse/

[26] Graphviz – graph visualization software.
URL http://www.graphviz.org/