Web Services-based Provisioning of Connections in GMPLS Optical Networks

Fábio Verdi[1], **Rafael Duarte**[1], **Fabrízzio C. de Lacerda**[1], **Maurício Magalhães**[1], **Eleri Cardozo**[1] , **Edmundo Madeira**[2]

[1]Department of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (Unicamp)
13083-970, Campinas-SP, Brasil.

[2]Institute of Computing (IC)
State University of Campinas (Unicamp)
13084-971, Campinas-SP, Brasil.

Abstract. *In this work we present an architecture for provisioning of connections in GMPLS optical networks. Such architecture is based on Web services and allows the establishment of two kinds of connections. The first one is known as Soft Permanent Connection (SPC) and is triggered by the manager of the optical domain. The SPC connects two network elements belonging to the same domain. The second one is an end-to-end connection by which a given client can send data across multiple domains. In this last case the client needs to negotiate with each domain to verify the availability of resources. We describe the modules of the architecture and the implementation of the modules that are responsible for provisioning of connections. Simulations using the GLASS simulator have been done in order to test our approach.*

Key Words: Management of Optical Networks, GMPLS, Web Services.

1. Introduction

The optical network technology has appeared as a solution for problems like low bandwidth and bottlenecks typically found in today's networks. The international community and organizations of network designers such as Internet Engineering Task Force (IETF), International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) and the Optical Internet Working Forum (OIF) are creating specifications in order to define standards to support the development of new solutions related to optical networks. All these organizations agree that future transmission networks are supposed to have ten to thousands of Gb of available bandwidth to attend all kinds of applications requiring faster transmission rates. These networks will likely consist of elements such as routers, switches, Dense Wavelength Division Multiplexing (DWDM) systems, Add-Drop Multiplexors (ADMs), photonic cross-connects (PXCs) and optical cross-connects (OXCs) [Mannie, 2003].

The Generalized Multiprotocol Label Switching (GMPLS) [Mannie, 2003] architecture is an evolution of MPLS and as such, the protocols defined for the MPLS model are reused in the GMPLS world by extending them. These extensions provide routing protocols such as Open Shortest Path First (OSPF) and signaling protocols such as Resource Reservation Protocol(RSVP), the capability to support not only switching in the

packet domain (e.g. IP) but also in the time division networks (using Time Division Multiplexing-TDM), where the switching decision is based on time slots, and optical networks where the switching decision is based on wavelengths or physical ports. Looking to the optical domain, the lightpaths are seen as LSPs (Label Switched Paths) or optical LSPs (from now on optical LSP and lightpath will be used interchangeably) and because of technologies like DWDM it is now possible to have a very large number of parallel links between two directly adjacent nodes (hundreds of wavelengths, or even thousands of wavelengths if multiple fibers are used).

Although there is a common sense to dynamically provide resources and perform the routing and signaling functions in the control plane in order to speed the provisioning of new connections, so far such job has been made typically by the management plane. An example of that is the ATM network that despite of having Private Network to Node Interface (PNNI) as its signaling protocol, it is almost unused. So, the question is: will the GMPLS succeed? We believe that GMPLS will likely be used inside the domains since some tests have been done and proved its feasibility [Cavazzoni et al., 2003]. Connections between domains are being deeply discussed through the international community and they have not reached a consensus yet. The External Network-to-Network Interface [E-NNI, 2004] is being defined by the OIF and its purpose is to allow the interactions between different domains by creating an standard interface. However, there is still missing a routing protocol that satisfies the requirements for routing across multiple optical domains [Truong et al., 2004].

While international organizations try to find a way towards the standardization, a group in Canada known as CANARIE is using another alternative for standards [Boutaba et al., 2004]. They assume that in the future the user will be responsible for controlling his lightpath and by means of concatenation of lightpaths such user will be able to reach any destination. They have developed a testbed network in order to validate their approach [CANARIE Project, 2004] and showed how simple their idea is. Basically, they have adopted XML and Web services as the core technologies.

The main objective of Web services is to allow applications to communicate with other applications outside corporate Intranets in an automatic fashion and without the human intervention. This is called application-to-application communication. To achieve this, Web services make use of the Service-Oriented Architecture (SOA). This architecture is the next step of distributed computing, which enables software components including application functions, objects and processes from different systems to be exposed as services.

Although some people argue that we already have some technologies for distributed computing like the Common Object Request Broker Architecture (CORBA), Microsoft Distributed Component Object Model (DCOM), Sun Java RMI (Remote Method Invocation) and others, those technologies adopted a tightly coupled mechanism of a synchronous communication model (request/response) having low interoperability among them. Furthermore, they are very complex and need heavy investment in infrastructure and at the same time they do not pass by firewalls.

On the other hand, by using XML standards and Internet protocols - such HTTP, FTP, SMTP - Web services are naturally loosely coupled asynchronous communication model and avoid those problems. It makes interoperability easy and allows the development of applications with any programming language, any protocol, or any platform by using such open standards.

In this project we are taking into account the use of Web services to provide interoperability between domains and also to provide a high level of flexibility to access the

management system. Inside the domains the GMPLS protocols are running to support signaling, routing and recovery and protection mechanisms. On the other hand, the relations between domains are performed by adopting almost the same idea of the CANARIE project in which each domain advertises its connectivity with neighboring domains.

The goal of this paper is to describe an architecture that allows a manager of a given domain to create, manage and tear down a Soft Permanent Connection (SPC) [ASON, 2001]. Also, this paper depicts the details on how a given client leases lightpaths from multiple domains in order to create an end-to-end connection. These two functions are seen as the first functions to be implemented in an optical network considering their importance and the demand for automatic provisioning. The architecture we are proposing intends to manage an optical network as a whole and as such, it is composed of modules responsible for fault management, admission control, policy-based management, grooming and Optical Virtual Private Network (OVPN) management. Specifically for this paper we shortly introduce the architecture and focus on the provisioning of connections giving details on how the provisioning is performed using our approach. The architecture has been developed taking into account the implementation of the modules responsible for the provisioning of the two connections. Simulations are being done using the GLASS simulator [GLASS Simulator, 2003] which provides an environment with IP over DWDM optical networks running GMPLS protocols. Such scenario represents a real GMPLS optical network and it is enough to validate our approach.

The paper is organized as follows. Next section presents some works related to our project. Section 3 details the architecture and its modules. Section 4 some aspects about the implementation are briefly described. Finally, Section 5 concludes the paper and presents some future works.

2. Related Works

The work presented in [Boutaba et al., 2004] considers that users can lease and manage their own resources (lightpaths). By means of a management system users can create bandwidth-guaranteed tunnels across multiple domains. The bandwidth resources are referred as lightpath objects (LPOs) and the management system can concatenate LPOs, partition, advertise/lease and establish end-to-end LPOs. The architecture presented in that work is based on the web services technology and is being tested on the Canadian research network CA*Net4 [CANARIE Project, 2004]. Although the approach is very promising it does not consider constraints imposed by each domain. The establishment of end-to-end lightpaths does not take into account domain policies. The work presented in [Truong et al., 2004] discusses a solution for that problem in order to ensure that management rules of each domain are enforced. The authors showed a web services-based architecture to establish end-to-end lightpaths considering policy utilization in admission and resource reservation control. Our approach is similar to both except that we are using GMPLS protocols inside the domain and web services between domains. The integration of the management plane with a GMPLS-based control plane is seen as a very difficult task. In [Verdi et al., 2004] we presented a policy-based Admission Control responsible for managing the installation and aggregation of packet-based LSPs within lightpaths (optical LSPs). In that work we defined some policies for performing packet-based grooming in the border of the optical network domain. Such policies are intended to minimize the number of LSPs rerouting in the optical domain.

3. Detailing the Architecture

The concepts of management, control and transport planes are defined in the context of the Automatically Switched Optical Network (ASON) [ASON, 2001] and are the same for both GMPLS and ASON architectures. Management plane is in charge of performing management functions for the system as a whole. It also coordinates all the planes taking into account the Fault, Configuration, Accounting, Performance and Security (FCAPS) management functional areas. The control plane is responsible for the call control and connection functions [ASON, 2001]. By means of signaling, calls are negotiated, connections are established/released and faults are notified and may be restored. Finally, the transport plane performs the bidirectional or unidirectional transfer of user data from one point to another.

Our architecture takes into account the three layers mentioned above and tries to keep the independence of each one. In this work we are interested in supporting two functions (related to the Configuration management area) and evaluate the interactions between the different planes. The first function is the creation of lightpaths performed by the domain's administrator (a.k.a SPC). The second function is performed by clients that need to cross multiple optical domains. Such client needs to access the Admission Control and asks for an end-to-end tunnel that satisfies the user traffic requirements. The Inter-domain Service (IDS) is responsible for contacting other domains in order to verify the availability of resources in each optical domain. This is necessary because domains have their own policies and the administrative rules of each domain need to be followed. Fig. 1 shows the architecture and its modules. In the following, each module and their interactions are described in details. At the end of this section we show the steps to perform the two functions mentioned above.

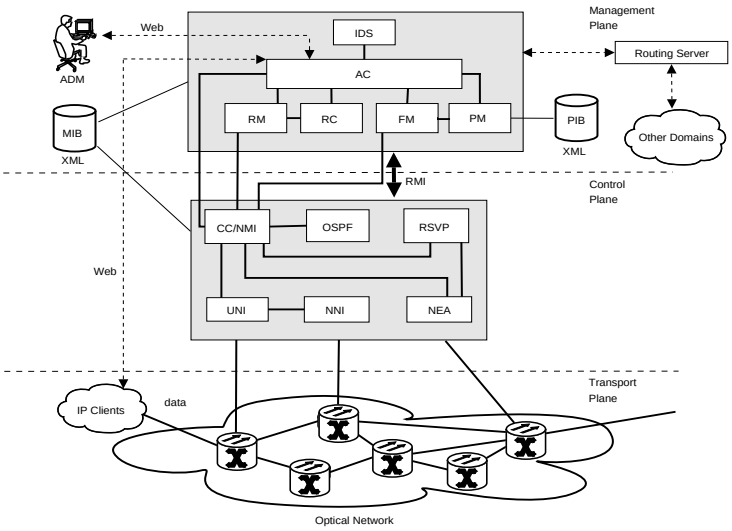

Figure 1: The Proposed Architecture.

Control Plane Modules:

- **RSVP - Resource Reservation Protocol**: RSVP is the signaling protocol used in this project. It is responsible for setting up/tearing down LSPs as required and carrying faults information to the Connection Controller/Network Management Interface (CC/NMI). RSVP is notified by some link management protocol (e.g. LMP) about node and link failures in the transport network. These notifications are sent to RSVP in the form of events;
- **OSPF - Open Shortest Path First**: OSPF is a well known link state protocol used in IP networks and was extended to support GMPLS capabilities. It is responsible

for propagating link state information necessary for routing algorithms to find TE paths. Usually, OSPF protocol sends link state information to the ingress node that is in charge of calculating TE routes. In our case, the OSPF module sends such information to the Resource Manager module located in the management plane;

- **NEA - Network Element Agent**: This module is located in the network elements in order to apply policies, configure the OXC taking any action related to the network element. Specifically for this work, the NEA is responsible for performing the crossconnection in each OXC located in the border of the optical domain;
- **UNI and NNI (User-to-Network Interface and Network-to-Network Interface**: UNI (UNI-C and UNI-N) [UNI, 2001] is the interface used to carry signaling messages between the edge node (client side) and the network ingress node (server side) in an overlay scenario. The NNI interface is used to carry the same signaling messages inside the transport network. RSVP has a strong relation with UNI and NNI since its messages are transfered through such interfaces (in Fig. 1 the lines between RSVP and UNI/NNI are omitted to make it clearer);
- **CC/NMI - Connection Controller/Network Management Interface**: It is used by the management plane to access the modules of the control plane. Also, it is used by the control plane to send events (e.g. fault notifications) to the management plane. The NMI interface defines the methods that the management plane can invoke. Currently, the CC/NMI component is a Remote Method Invocation (RMI) object offering a simple interface for the management plane.

The modules RSVP, OSPF, UNI and NNI belong to the GLASS Simulator and we just use them to perform the simulations. The NEA and the CC/NMI are located in the control plane performing tasks on behalf of the management plane. Note that RSVP and OSPF modules represent the RSVP-TE and OSPF-TE with extensions to support GMPLS capabilities.

In the following, the management plane modules are described.

Management Plane Modules:

- **Admission Control - AC**: It represents the interface whereby administrators and client networks have access to the management system. It is responsible for receiving connection requests, verifying pre-defined Service Level Agreements (SLAs), receiving notifications from the CC/NMI and invoking the Policy Manager, Resource Manager and Fault Manager to perform specific management tasks;
- **Policy Manager - PM**: PM is also known as the Policy Decision Point (PDP) and is responsible for applying the policies defined to the domain being managed. Policies are stored in a Policy Repository described using XML. In case of IP/MPLS LSPs grooming, the PM always tries to minimize the quantity of removals that are needed to accommodate an LSP. Considering that LSP removals are necessary in case of having different classes of flows, the AC is in charge of interacting with the control plane to send a tear down message for each LP LSP that needs to be removed. Details of this as well as scenarios and results of testing the defined policies can be found in [Verdi et al., 2004];
- **Fault Manager - FM**: It is responsible for performing the fault management. In case that the NEA is not able or does not have enough information to take actions when a given event happens in the transport network, the FM is notified to act and correct the problem. These corrective actions are based on some policies previously defined by the administrator and try to minimize the impact of a negative event. FM also keeps a fault information base which stores all the faults that

occurred in a period of time. This is useful to detect points of faults and avoid dangerous paths;

- **Resource Manager - RM**: It is responsible for managing the transport network resources (e.g. lambdas, available bandwidth of lightpaths, points of grooming, etc.). Constraint-based routing done by the Routing Controller uses the information provided by the Resource Manager to find TE paths. Specifically for this work, the RM is responsible for sending advertisements to the Routing Server (see below) about the connectivity of the domain with other domains;
- **Routing Controller - RC**: Basically, the tasks of this module are the same from the ones described in [ASON, 2001]. It is responsible for finding a TE route for the establishment of connections inside the optical domain and running Routing and Wavelength Assignment (RWA) algorithms. It also provides basic information about the transport network topology for management purposes;
- **Inter-domain Service - IDS**: This module performs the end-to-end negotiation across multiple domains. It contacts other domains in order to ask for a resource (lightpath) and verify the availability for using such resource. This negotiation can be based on a previously agreed SLA between the domains.

Although each module has its own group of specific functions, they work together to provide some management functionalities. The information model stored in the Management Information Base (MIB) has information about connections (optical LSPs), available resources (e.g. lambdas, bandwidth in each lightpath), points of grooming, clients, backup lightpaths and optical elements. The policy information model stored in the Policy Repository describes the policies defined to manage the domain. The information is used for policy-based admission control, policy-based traffic aggregation (grooming) and policy-based fault treatment.

The Routing Server (RS) does not belong to the management plane as we can see in Fig. 1. It acts as a centralized routing for all the domains and as such, each domain sends information about its connectivity with other domains to the Routing Server. It collects this information and creates a routing table. Note that such routing table refers to the routing information among domains. Clients that need to create a connection across multiple domains ask the RS for a route.

The level of information that is published depends on what kind of data the routing algorithm needs. Information about the internal topology of the domain is not published since each domain does not want to advertise how its infrastructure is organized. Basically, information about link cost is enough considering that it is desirable to preserve confidentiality of each domain [Farrel et al., 2004b]. In this work we are advertising only the information about connectivity between domains and no internal information is currently being published.

Below (in Fig. 2), we show the sequence diagram for the SPC connection (performed by the administrator). The information that is necessary to create an optical LSP is:

- Explicit Route: The IDs of each node belonging to the route;
- Ingress Node and Interface: The MPLS ingress source node and its interface ID;
- Egress Node and Interface: The MPLS egress source node and its interface ID;
- Rate: The bandwidth of the lightpath;
- Level of Protection: 1+1, 1:1, 1:N, no protection. For more details about fault terminology see [Mannie and Papadimitriou, 2004].

SPC Setup:

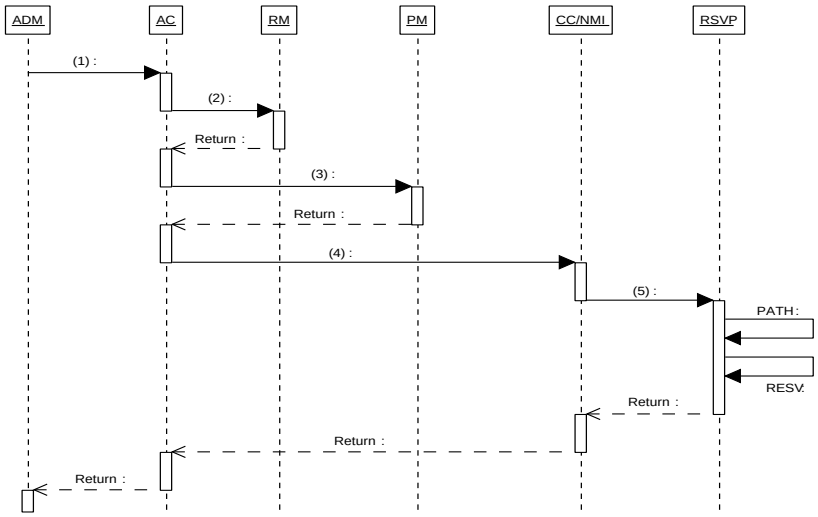

Figure 2: Soft Permanent Connection Setup.

1. ADM sends an SPC request to the AC by using a web interface. The Administrator gives the request all the necessary information to set up the SPC;
2. Upon receiving the SPC request, AC asks RM to verify if there are available resources (lambdas) in the given route;
3. If there are resources, AC notifies the PM in order to apply the defined policies on the request using the explicit route;
4. Considering that the request is accepted, AC requests CC/NMI to start setting up the lightpath;
5. CC/NMI sends the parameters to the RSVP which in turn is responsible for signaling the optical LSP. This is done by sending PATHs and RESVs messages.

After finishing the connection setup, a notification is sent to AC which is in charge of updating the MIB. Note that the Routing Controller is not being used since the explicit route is given by the manager. Further on we will be considering the use of the RC to calculate the route taking into account Traffic Engineering constraints.

On the other hand, to create an end-to-end connection for a client it is necessary to interact with other domains. There has been an intensive discussion on how to create an end-to-end connection in a multi-domain scenario. A Path Computation Element (PCE), defined under the IETF umbrella, is an entity that is capable of computing a network path or route based on a network graph, and applying computational constraints [Farrel et al., 2004a]. In our work, the Routing Server (RS) acts as a centralized PCE providing the routes across multiple domains. Within the domains the Routing Controller (RC) acts as a local PCE providing local routes from an ingress node to an egress node in the same domain. This modularization gives more flexibility to independently extend each module. In the present architecture, each domain advertises its connectivity to the Routing Server (steps 1 and 2 in Fig. 3). Afterwards, each client that needs to create an end-to-end connection queries the Routing Server to find the route (steps 3 and 4 in Fig. 3). The Xindice is responsible for storing the XML files (see Section 4).

In this work we are considering that the domains belong to the same administrative instance. In this case, the centralized Routing Server makes sense since it is a trustful entity inside the administrative domain. However, in case of having different administrative domains, the centralized solution is not feasible and a distributed approach has to be used. It consists in flooding the resumed routing information of each domain to other neighbor domains. The PCE located in each domain is responsible for propagating

and receiving information to and from other domains. A client that needs to create an end-to-end connection asks the local PCE for finding the end-to-end route.

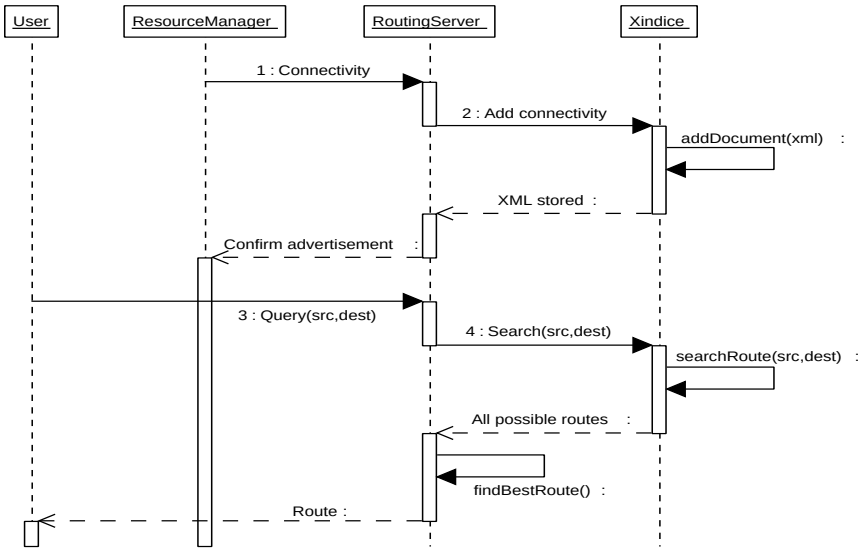

Figure 3: Advertisement and Query.

After the client queries for a route in the Routing Server (RS) and lets say that the route is domain 1 and domain 2, the client needs to send a requisition to AC in order to ask for the end-to-end connection. Fig. 4 depicts step-by-step how to create an end-to-end connection across two domains considering that the lightpaths inside the domains are already created and as such, one of those lightpaths is chosen by applying some policies.

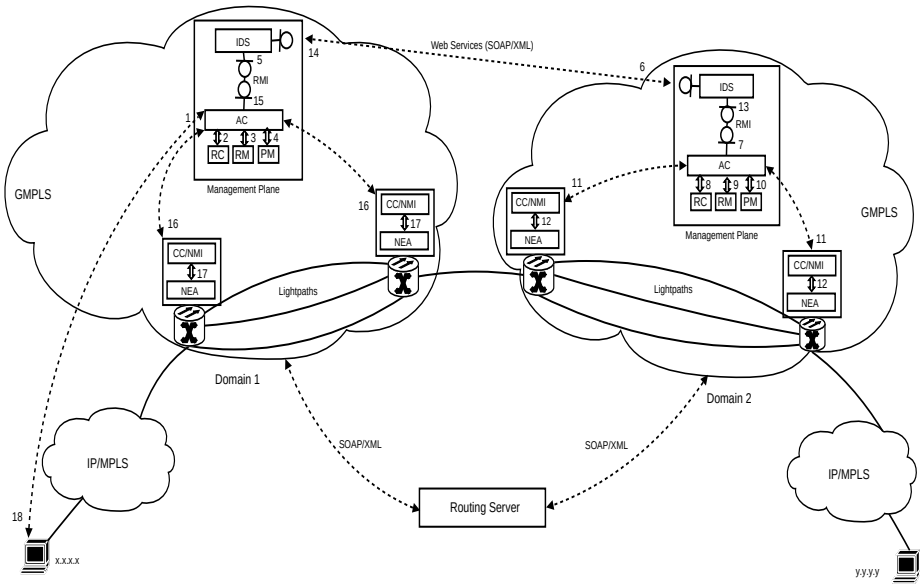

Figure 4: Inter-domain Connection Setup.

Basically, the client gives the route obtained from the RS and other constraints like bandwidth, level of protection to the AC (step 1 in Fig. 4). The AC validates the request and asks the RC to find the egress node(s) (in domain 1) that connect to domain 2 (step 2). After getting the egress node (s), the RM is invoked to find all the lightpaths between the ingress node and egress node(s) in domain 1 (step 3). In step 4 of Fig. 4 the PM applies the defined policies in order to find the most appropriated lightpath to be used (or, eventually, to create a new one). By doing this, the management system concludes that the resource in domain 1 is pre-reserved and now it is necessary to forward the request to domain 2. The AC sends the requisition to the local IDS (step 5) which in turn communicates with the

next IDS (step 6). The domain 2's IDS forwards the request to AC (step 7). Steps 8, 9 and 10 are the same as 2, 3 and 4. After verifying that there is a resource available and since domain 2 is the last domain in the route, the crossconnection is done in each boundary OXC (steps 11 and 12). It means that a given lightpath connecting domain 1 to domain 2 is crossconnected with an internal lightpath of domain 2. There can be a negotiation phase between IDSs in order to choose the best lightpath to be used between the two domains. After performing all the necessary configurations in domain 2, the domain's 2 IDS replies to the upstream IDS (steps 13, 14 and 15). Steps 16 and 17 are the same as steps 11 and 12 (local boundary crossconnection). Finally, the end-to-end connection is ready and an identifier of that connection is sent back to the client to be used to send data.

By adopting this model, each domain can have its own policies and the negotiation between domains can be done through the IDS. The way such negotiation is to be done is a matter of protocol specification. Domains can freely agree on what kind of attributes can be negotiated. On the other hand, E-NNI [E-NNI, 2004] signaling depends on the standardization procedures that takes time for companies to adopt the solution. As a consequence, we believe that for the mid-time, inter-domain establishment/negotiation of connections will be based on different solutions. One of these solutions is likely to be through the Web services technology.

4. Implementation

The proposed architecture has been implemented and tested in order to validate our approach. Currently the Admission Control, the Policy Manager, the Resource Manager (partially) and the Routing Server are implemented. Also, the interaction of the AC with the CC/NMI to set up lightpaths is ready. For this article, we assumed that the inter-domain negotiation is done since the Inter-domain Service is being currently implemented. The modules are being developed in Java and some free tools are being used to facilitate the implementation.

4.1. Detailing the Management Plane Implementation

The web interface is based on the JSP technology running on the Apache Tomcat container 5.0.18 [Apache Tomcat, 2004] and the web service module is created using the Apache AXIS 1.1 [Apache AXIS, 2004]. The management plane modules as well as the CC/NMI are remote objects developed using the Java RMI technology.

The AC module is composed of small components as can be seen in Fig. 5. The client application shown in the right side of Fig. 5 represents all the web services applications made up using the Web Services Description Language (WSDL) provided by the AC. The JSP module offers a web interface to human users and also translates incoming requests into XML-based Simple Object Access Protocol (SOAP) messages that are sent to the Web Service module. Upon receiving the SOAP message, the Web Service module forwards the requisition to the AC Engine by using the Sockets API or RMI.

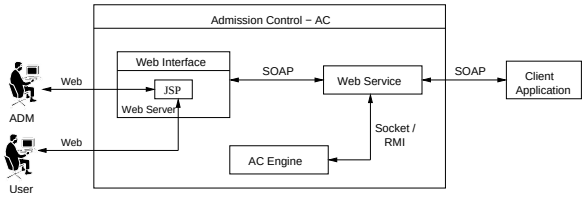

Figure 5: Admission Control Internal Architecture.

The communication between the Resource Manager and the Routing Server is done through the XML-based SOAP protocol. A general XML example of the informa-

tion sent from the Resource Manager to the Routing Server to advertise a given connectivity is shown in Fig. 6.

```
<?xml version="1.0" encoding="UTF-8"?>
<connection> <!-- Description of connectivity from torres domain to iracema, mosqueiro and riviera -->
   <srcDomain>torres</srcDomain>
   <destDomain>iracema</destDomain>
   <destDomain>mosqueiro</destDomain>
   <destDomain>riviera</destDomain>
</connection>
```

Figure 6: Advertisement of Connectivities in a Domain.

The XML above was advertised by the torres domain in order to publish its connectivity with other three domains: iracema, mosqueiro and riviera. Each domain advertises its connectivity in the same way.

It is important to point out that the XML of Fig. 6 is very simple and is useful only to show the connectivity among domains. More information about the links connecting the domains can be added to the XML. In case of using TE algorithms, some metrics like link cost and level of protection can be published in order to improve the routing and better balance the load among domains. Furthermore, some filtered information corresponding to the connections within the domains can also be published. For example, the information may be limited to Forwarding Adjacencies (FAs) across other domains, or the information may be aggregated or abstracted to preserve confidentiality [Farrel et al., 2004b].

The Xindice [Xindice Project, 2004] is used as the XML database to store the XML data both in the Routing Server and locally in each domain. The access to the MIB as well as the XML handling is done by using SAX/DOM/XPath tools. To simulate an optical network with IP/MPLS clients we have used the GLASS Simulator from NIST [GLASS Simulator, 2003]. GLASS is a discrete network event simulator for GMPLS-based Optical Internet and by using it, we created typical scenarios with IP over DWDM (see Section 4.2 for details).

The AC can be accessed either by a JSP web page or by a web service client application. Below (Fig. 7), an excerpt of the WSDL of the AC Web service is showed.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:http://www.dca.fee.unicamp.br">
   <wsdl:message name="createLightpathResponse">
      <wsdl:part name="createLightpathReturn" type="xsd:string"/> <!-- Return Confirmation -->
   </wsdl:message>
   <wsdl:message name="createLightpathRequest">
      <wsdl:part name="in0" type="tns1:ArrayOf_xsd_string"/> <!-- Explicit Route -->
      <wsdl:part name="in1" type="xsd:string"/> <!-- Ingress Node -->
      <wsdl:part name="in2" type="xsd:int"/> <!-- Ingress Interface -->
      <wsdl:part name="in3" type="xsd:string"/> <!-- Egress Node -->
      <wsdl:part name="in4" type="xsd:int"/> <!-- Egress Interface -->
      <wsdl:part name="in5" type="xsd:int"/> <!-- Bandwidth -->
      <wsdl:part name="in6" type="xsd:int"/> <!-- Level of Protection-->
   </wsdl:message>
   . . .
   <wsdl:service name="ACService">
      <wsdl:port binding="impl:ACSoapBinding" name="AC">
         <wsdlsoap:address location="http://riviera:8080/axis/services/AC"/>
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

Figure 7: WSDL for the AC Web Service.

Fig. 8 shows the web page by which the manager creates an SPC. Note that all the necessary attributes listed in Section 3 are collected through this interface. The data

collected in the web page showed in Fig. 8 are based on the topology depicted in Fig. 11. The explicit route is represented by the node identifiers belonging to the connection. The bandwidth is 1 Gb/s. The client ingress and egress nodes as well as the client ingress and egress interfaces are necessary to create the tunnel by which data is to be sent. The level of protection in this example is 0 that means no protection.

The web page to create an end-to-end connection is quite similar to the one showed in Fig. 8 and will not be depicted for sake of space.

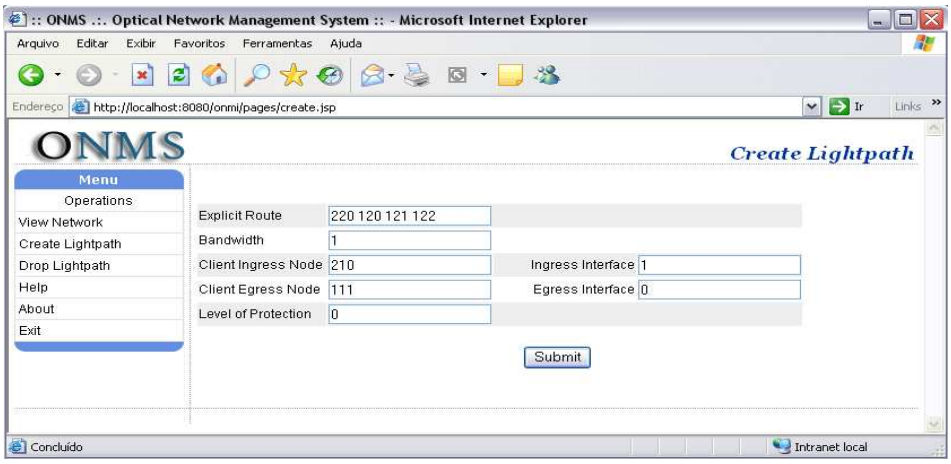

Figure 8: Web Page to create an SPC.

We are currently working on the Inter-domain Service. The IDS is being implemented as a Web service and will extend the work developed in [Pedatella et al., 2003]. In that work the authors created an end-to-end Bandwidth Broker for MPLS networks. We are defining the protocol based on the mentioned work and adapting it to support optical attributes. The IDS and the Routing Server are developed using the Web services technology. Since they are accessed from different domains, the XML-human readable standard and the SOAP over HTTP are seen as a very promising combination for current and future distributed system applications.

4.2. GLASS Simulator Details

GLASS (GMPLS Lightwave Agile Switching Simulator) is a discrete event simulator for GMPLS-optical Internet developed in Java and runs over the Scalable Simulation Framework Network (SSFNet) [SSFNet Simulator, 2003]. An important point of GLASS is that its source code is open and free allowing a very deep research on how the simulation and the protocols work. Furthermore, changes and adaptations in the simulator can be done taking into account specific requirements of our project.

The simulator implements the main GMPLS protocols such as RSVP and CR-LDP responsible for signaling LSPs, and OSPF responsible for routing. Note that all these protocols have the appropriate extensions to support Generalized MPLS. Since the simulator is a starting point towards new generation networks (NGN), currently it does not implement the Link Management Protocol (LMP) responsible for link correlation, fault detection and isolation. GLASS also has diverse example implementations of failure propagation and recovery protocols that are not IP based (pure optical signaling). Moreover, it has RWA support with specific algorithms such as the ShortestPathDistance for routing and BestFit for wavelength assignment.

The multiple-domain scenario is simulated by running a GLASS simulator instance in different machines (PC Linux running Slackware distribution). Each instance of it represents a different domain. The communication between domains is implemented

using the Sockets API and to fully validate the scenario we are sending data across the domains after installing and end-to-end connection.

It is important to highlight that the GLASS simulator was developed to run in a single machine without interacting with other machines running other instances of GLASS. This kind of interaction is not supported by the normal distribution of GLASS from NIST. The communication between two different instances of the simulator done through sockets was added by our group to support the multiple-domain scenario. First of all, we had to give the notion of end-to-end normal LSP signaling. For this paper, we assumed that there is a permanent connection between the domains. Such connection is represented by lightpaths that interconnect two domains. There can be many lightpaths between any two domains. Fig. 9 shows a scenario with one lightpath between domains 1 and 2.

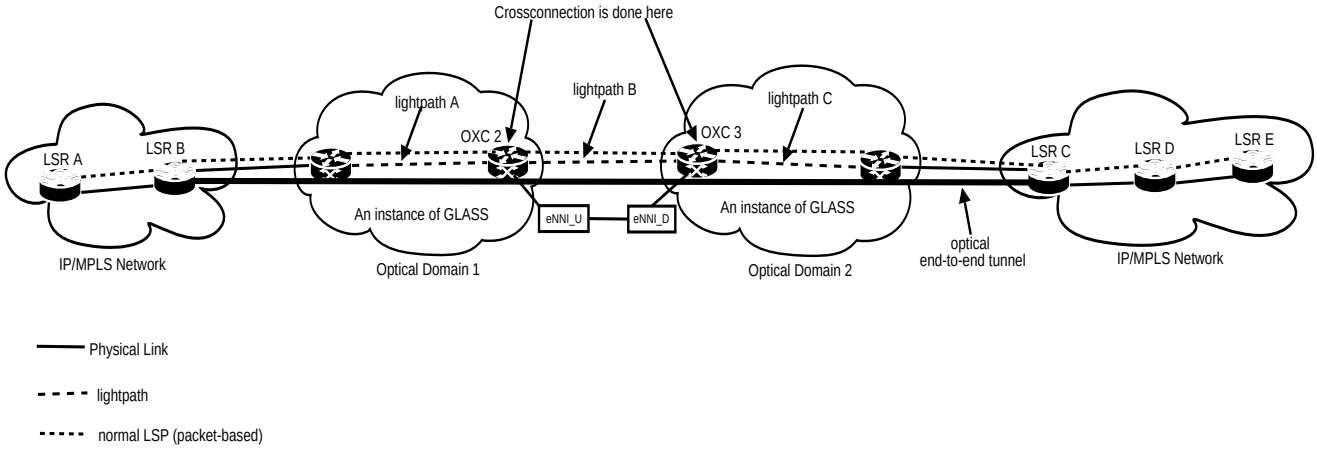

Figure 9: Multi-domain scenario with GLASS.

We can see in Fig. 9 that there are two optical domains (1 and 2). Furthermore, in each domain there is a lightpath connecting an ingress OXC to an egress OXC (lightpaths A and C). These two lightpaths are SPCs and were set up by the manager of the optical domain. The lightpath B connects domain 1 to domain 2. In order to signal an end-to-end normal LSP from LSR A to LSR E (for example), the PATH message transparently needs to go through domains 1 and 2 as if LSR B and C were peers. To implement this end-to-end signaling, two new modules were added to GLASS (see Fig. 9), eNNI_U (E-NNI Upstream) and eNNI_D (E-NNI Downstream), both defined in the context of the OIForum [E-NNI, 2004]. However, in our project theses two modules do not implement the abstract messages defined in the OIForum. Such messages are being specified to support the signaling between domains (Switched Connection) and it is a working in progress in the Forum.

In our project, the eNNI_U module is responsible for receiving the PATH message from the RSVP module (step 1 in Fig. 10) and forwarding it (through sockets) to the eNNI_D module of the next downstream domain (step 3 in Fig. 10). Before forwarding the message, the eNNI_U module opens the explicit route to find out the next domain to send the PATH message (step 2). Then, the parameters belonging to the PATH message that are necessary to continue the signaling in the next domain are sent to the eNNI_D module, which in turn is responsible for gathering all the parameters and mounting a new PATH message using the received parameters (step 4). Finally, the message is sent to the RSVP module to be forwarded across the domain (step 5).

The TrafficParam object of the RSVP protocol was modified to carry the explicit route representing the end-to-end connection [1]. Also, the TrafficParam object carries the

[1]The explicit route is formed by an identifier of each domain.

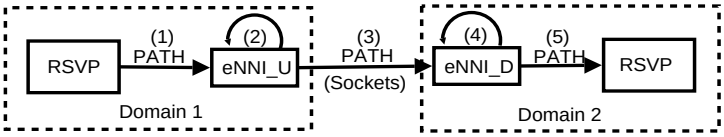

Figure 10: End-to-End Signaling implemented in the GLASS Simulator.

identifier of the tunnel to be used in each domain. This tunnel identifier is used by the RSVP protocol in order for it to know in which lightpath the PATH message is to be sent.

Below (in Fig. 11), we show the topology currently used in the simulations. The topology is the same for all the domains. Note that the topology is quite simple since the purpose here is to validate our management architecture taking into account the provisioning of connections. Scenarios that require more nodes with more routes are being tested, mainly the ones related to failures and recovery.

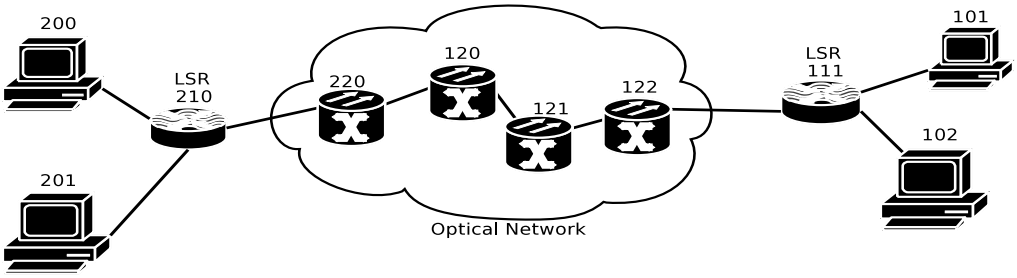

Figure 11: Topology of a single domain.

5. Conclusion and Future Works

The architecture presented in this paper is a starting point of a project whose main objective is to develop an integrated infrastructure to manage GMPLS-based optical networks. We describe some basic modules of the control and management planes illustrating some interactions among the modules when the administrator sets up an SPC and when an user asks for an end-to-end connection. Although we have adopted the same idea of a PCE for end-to-end path computation, in our project there is a centralized module (Routing Server) responsible for end-to-end path computation among multiple domains and a local module (Routing Controller) in each domain responsible for the local routing.

The implementation of the architecture has showed that the modules we have been defining are feasible and the integration of different technologies seems to be very useful in the context of GMPLS optical networks management. By using the Web Services technology to provide the administrator and the user the access to the management system and also to integrate the interactions among domains, we are going towards new ways of developing web-based management solutions. Not only are the research community willing to adopt XML-based standards but also many consortiums and companies are driving the development for the convergence and adoption of XML-based integration.

In the context of this project there are some issues to be taken into account further. The fault notification, fault handling and the utilization of policies for creating different levels of protection/recovery are being defined and tested. The E-NNI abstract messages (the ones related to provisioning of connections) as defined by the OIForum are to be implemented in order to compare the solution presented in this paper with the signaling using the E-NNI interfaces. The distributed approach considering different administrative domains is being implemented. Finally, some performance issues in using Web services are being analized by our group.

Acknowledgments: The authors would like to thank CNPq and CAPES for their support. This work was also supported by the Research and Development Centre, Ericsson Telecomunicações S.A., Brazil.

References

Apache AXIS (2004). http://ws.apache.org/axis/.

Apache Tomcat (2004). http://jakarta.apache.org/.

ASON (2001). ITU-T: Architecture for the Automatically Switched Optical Network (ASON), G.8080/Y.1304.

Boutaba, R., Golab, W., Iraqui, Y., and Arnaud, B. S. (2004). Lightpaths on Demand: A Web-Services-Based Management System. *IEEE Communications Magazine*, Vol. 42, No. 7:101–107.

CANARIE Project (2004). http://www.canarie.ca/.

Cavazzoni, C., Barosco, V., Alessandro, A., Manzalini, A., Milani, S., Ricucci, G., Morro, R., Geerdsen, R., Hartmer, U., Lehr, G., Pauluhn, U., Wevering, S., Pendarakis, D., Wauters, N., Gigantino, R., Vasseur, J., Shimano, K., Monari, G., and Salvioni, A. (2003). The IP/MPLS Over ASON/GMPLS Test Bed of the IST Project LION. *IEEE Journal of Lightwave Technology*, Vol. 21, No. 11:2791–2803.

E-NNI (2004). OIF Intra-Carrier E-NNI 01.0 Signaling Specification.

Farrel, A., Vasseur, J.-F., and Ash, J. (2004a). Path Computation Element (PCE) Architecture. *draft-ash-pce-architecture-00.txt*.

Farrel, A., Vasseur, J.-F., and Ayyangar, A. (2004b). A Framework for Inter-Domain MPLS Traffic Engineering. *draft-farrel-ccamp-inter-domain-framework-01.txt*.

GLASS Simulator (2003). http://dns.antd.nist.gov/glass/.

Mannie, E. (2003). Generalized Multi-Protocol Label Switching Architecture. *draft-ietf-ccamp-gmpls-architecture-07.txt*.

Mannie, E. and Papadimitriou, D. (2004). Recovery (Protection and Restoration) Terminology for Generalized Multi-Protocol Label Switching (GMPLS). *draft-ietf-ccamp-gmpls-recovery-terminology-05.txt*.

Pedatella, R., Madeira, E., and Magalhães, M. (2003). Um Modelo de Negociação de Serviços com QoS Fim-a-Fim na Internet. *Revista da Sociedade Brasileira de Telecomunicações (RevSBrT)*, Vol. 18, No. 2:153–164.

SSFNet Simulator (2003). http://www.ssfnet.org/.

Truong, D. L., Cherkaoui, O., Elbiaze, H., Rico, N., and Aboulhamia, M. (2004). A Policy-based approach for user controlled Lightpath Provisioning. *IFIP/IEEE NOMS, Seoul, Korea*, pages 859–872.

UNI (2001). OIF User Network Interface (UNI) 1.0 Signaling Specification.

Verdi, F. L., Madeira, E., and Magalhães, M. (2004). Policy-based Admission Control in GMPLS Optical Networks. *First IEEE International Conference on Broadband Networks - Broadnets'04 (formerly OptiComm), San Jose, USA*, pages 337–339.

Xindice Project (2004). http://xml.apache.org/xindice/.