# USER-MANAGED END-TO-END LIGHTPATH PROVISIONING OVER CA*NET 4

**Jing Wu, Scott Campbell, J. Michel Savoie, Hanxi Zhang**

Communications Research Centre Canada, 3701 Carling Avenue, Ottawa, Ontario, Canada K2H 8S2, {jing.wu, scott.campbell, michel.savoie, hanxi.zhang}@crc.ca

**Gregor v. Bochmann**

School of Information Technology & Engineering, University of Ottawa, bochmann@site.uottawa.ca

**Bill St. Arnaud**

Canarie Inc., bill.st.arnaud@canarie.ca

Abstract: Customer owned and managed optical networks bring new cost saving benefits. Their technical challenges and the design for a customer's management tool are presented.

## 1. Introduction

There are basically two types of customer owned and managed optical networks: metro dark fibre networks and long-haul wavelength networks. Schools, hospitals and government departments are acquiring their own metro dark fibres. They participate in so-called "condominium" dark fibre networks to better manage their connectivity and bandwidth. In the long-haul area, many providers are selling or leasing point-to-point wavelengths. Some providers are offering "condominium" wavelength solutions, where a number of customers share the capital costs of deploying long-haul optical networks. In return, each customer in the condominium consortium owns a set of wavelengths.

In this paper we discuss the benefits and technical challenges presented by customer owned and managed optical networks. We then propose an architectural framework, communication protocols and software engineering technologies that are useful for building management tools for this context. In the second part of the paper we describe the system and software architecture of a management system developed for user-managed lightpath provisioning in the context of Canarie's CA*net 4 [1].

## 2. Benefits and Challenges of Customer Owned Optical Networks

### 2.1 Benefits

Through arrangements such as condominium dark fibre and condominium wavelength networks, enterprise and university research networks can substantially reduce the cost of bandwidth, as it now largely becomes a capital cost rather than an ongoing monthly service charge. This is particularly important when the demand for bandwidth increases significantly over time.

In customer-managed optical networks, the customer is able to optimize the overall resource consumption. The customer purchases dark fibres and/or wavelengths from a number of independent suppliers, as well as participates in a condominium wavelength built for some portions of their network. Therefore, the customer has more flexibility in network planning and deployment and is able to negotiate the best deal from different suppliers. The customer can fine-tune the usage of each resource from each independent supplier.

Customer-managed networks provide indirect cost savings through reduced Internet costs via remote peering and transit. Large enterprise or research networks can use customer-managed connections to directly peer with each other and more importantly set up bandwidth guaranteed connections to no-cost peering exchanges. The customer management of the cross connect allows the customer to change the peering relationship without having to contact a central management body or pay expensive Internet transit fees.

Since the customer directly owns and manages an optical network, the bandwidth and quality of service are guaranteed. The complexity of service management at the IP layer is removed. A new opportunity for potential cost savings is introduced by eliminating expensive high-end routers in the core and replacing them with optical switches. However, there is a cost in terms of network efficiency as the functionality of IP packet multiplexing is lost. The trade-off needs to be explored regarding the bandwidth efficiency and the cost of wavelengths versus the cost of routers.

### 2.2 Technical Challenges

The first technical challenge is the management of networks with resources from different sources. Only the customer has total visibility of its own network and no single provider can see all the network elements. The traditional centrally managed hierarchical networking technologies, e.g. Generalized Multi-Protocol Label Switching (GMPLS) and Automatic Switched Optical/Transport Network (ASON/ASTN), assume that the provider has total visibility of all network elements and a common management system for all optical equipment. Virtual private network (VPN) technology allows provisioning of customer networks within a single provider's domain. Clearly this type of architecture is not practical with customer-managed networks. For the protection and restoration, the customer, rather than any provider, is in a better position to decide the optimal solution. How to co-ordinate the protection and restoration involving multiple providers is an open issue.

The second challenge is the collaboration among multiple independent customers without co-ordination through centralized management. Customer-managed networks adopt the peer-to-peer architecture, in which customers peer with each other. Each customer domain not only receives transport services from other customer domains but also contributes new transport services to other customer domains. A link between two customer domains is equally controlled by both of them as opposed to being controlled as an access link, in which a provider plays an active role while a customer domain plays a passive role. During the establishment of an end-to-end connection, each segment of the connection between domains is set up on a peer-to-peer basis. Central guiding intelligence and arbitration of conflicts may be necessary, but day-to-day management and per connection control should be decentralized. An end-to-end connection from one customer to another involves at least two different customer domains, and if transit is required, one or more intermediate transit domains may participate. So the collaboration among multiple independent customers is critical for end-to-end connection provisioning. How to search and take control of resources in collaborative domains has to be addressed. Policy enforcement, authorization and authentication have to be applied. The organization of customer federations is a new issue.

The third challenge is the dynamic partitioning of a provider's resource to customers. VPN technology allows partitioning of a provider's resource to customers. However, VPNs are not as dynamic as some emerging applications require, e.g. Grid computing, etc. Some customers prefer significant control and management capabilities in the provider's domain. They want a fine-grained resource allocation, which enables further optimization of the overall resource consumption. Deploying and upgrading customer's services is difficult and time-consuming in today's networks due to the closed, integrated architecture of network nodes. How to manage a provider's network element in a condominium fashion is challenging.

## 3. Architectural Considerations for Customer-Managed Networks

### 3.1 Network Architecture and Management Domains

We assume a general mesh-type of network architecture where the nodes consists of optical cross-connects and the edges consist of optical fibers over which multiplexing is provided through wavelength division multiplexing and/or SONET time division multiplexing. The optical lightpaths that are established through the optical network are mostly used as links between packet switches and IP routers that are connected at the end-points of these lightpaths.

In analogy to the architectural structure of the Internet, we assume that the optical network is partitioned into a number of administrative domains. Within each domain, we assume that complete management information would be available which could be consulted when a new lightpath is to be established. However, between different domains, only partial information would be exchanged.

The optical links within the network may belong to different parties, even within the same management domain. We note that this also implies shared ownership of the switches. In fact, it is usually assumed that for a link between two switches that belongs to party A, the two end-points, that is, the respective ports in the two switches, also belong to the same party.

### 3.2 Advertising Networking Resources

In order to establish a new end-to-end connection, it is sometimes necessary to concatenate links that belong to different parties. The following two cases may be considered.

- Peering: Two lightpath spans from switch A to B, and from B to C are to be interconnected at switch B in order to create a new lightpath span from A to C; the lightpath spans from A to B and from B to C may belong to different parties.
- Leasing: In order to establish a lightpath span from node A to node C, a given party P1 may own a free lightpath span from node A to node B, and needs another free lightpath span from B to C to be interconnected with the former; another party P2 may own such a lightpath span and be willing to sublease it to P1.

In order to find free lightpath spans that belong to other parties, it has been proposed that free resources could be publicly advertised. The concept of service directories has been proposed in distributed computing and several versions of this concept have been realized as Common Object Request Broker Architecture (CORBA), Jini Lookup Services and JavaSpaces [2, 3, 4], or Web Services Directories [5]. Each resource or service that is to be made available to be used by other parties must be registered in the service directory. Potential users of these resources or services may query the directory to find a resource that fits their requirements. In order to support meaningful queries, the object-oriented paradigm of object instances and classes is used. Each registered object instance belongs to a class that defines the properties of the object instances. In addition, the class defines a certain number of attributes, and each instance is characterized by the values of its attributes. A user searching for a service will therefore indicate the class of service desired and possibly some attribute values. For instance, the party P1, in the example above, will search the directory for an object of class *lightpath span* with the attributes *source=B* and *destination=C*.

In this context, it is also conceivable that a given party, say P1, has leased a lightpath span from party P2 in order to build a longer lightpath span from, say A to C. Now P1 may subdivide the bandwidth of the lightpath span and create N low-bandwidth lightpath spans from A to C. P1 may use a few of these lightpath spans and may re-advertise the others as available in one of the service directories. In this way, the leasing mechanism may be used as a basis for establishing an optical lightpath market. The advertised leases could be associated with a price to be paid.

### 3.3 Distributed Resource Management

Even if we have access to all the information about available lightpath spans and the possibility of leases, it is important to note that the actual resource reservations necessary for the establishment of a new lightpath will generally involve several parties in several management domains and therefore several databases in which the status information about the different resources will be stored. In order to avoid inconsistencies due to concurrent access to these resources by several users, it is important to foresee appropriate mechanisms for mutual exclusion of access. Since one also needs persistent storage in the presence of occasional crashes of the computers that contain the databases, the transaction concept developed for centralized and distributed databases appears to be useful here. A transaction is a sequence of actions, such as reading the status of resources and requesting resource reservations, that are all executed as specified (the transaction commits) or not executed at all (the transaction aborts). This applies even in the case where one of the computers managing certain resources crashes during the operations (in this case the transaction aborts).

It is interesting to note that JavaSpaces represent a service which provides persistent storage of object instances, retrieval of object instances, and transaction management involving actions taking place in different JavaSpaces possibly residing on different computers. This technology is therefore an interesting platform for implementing distributed resource management tools.

### 3.4 Why Use Jini and JavaSpaces for the Implementation of Our Management Tool

Jini and JavaSpaces provide a number of advantages for developing distributed applications. Jini runs on top of Java and uses Remote Method Invocation (RMI) to access remote services, which are concepts that are very clear cut and well known. Jini also provides a set of application programming interfaces that hide the underlying complexity of distributed computing from the user. The Jini Lookup Service (JLS), a distributed service registry allows users to find services without having to know anything about where they are located. Through the federation of JLS's, a user can find any service in any domain. Services that are registered in the JLS are also persistent and will be maintained as long as the service is alive. Jini also provides mechanisms for distributed events, distributed leasing and transactions, which will ensure that any request will either complete fully, or not at all. An infrastructure for the management of dynamic service networks was presented by using Jini technology [6].

JavaSpaces provides a distributed data store for Java objects. Objects stored in a JavaSpaces are loosely coupled; anyone can take an object from a space without knowing (or caring) the details about the person who put it there. Operations on JavaSpaces are transactionally secure. All the service calls in a transaction are committed, or none at all. Transactions are supported for a single operation on a single space as well as multiple operations over many spaces. Like the JLS, JavaSpaces are also persistent; an object will remain in a space until it is explicitly removed. It also includes the search facilities of Jini, and its mechanisms for distributed leases.

The main reason why Jini and JavaSpaces technology was chosen over Extensible Markup Language (XML) Web services is because of the functionality that JavaSpaces provide for storing Lightpath Objects (LPOs) in a distributed fashion. Also because the JLS is more powerful and more mature than the XML based service registries. Since Jini passes Java objects via RMI, there is no need to have XML schema definitions for all remote service calls. Although using Jini/JavaSpaces limits us to the Java programming language (while XML is language independent), the internal Jini service calls are transparent to the user and to other applications that may use the Open Grid Service Interface (OGSI) Grid service interface provided by our Grid Service Access Point (SAP).

# 4. Design of a Management Tool Based on Jini and JavaSpaces

## 4.1 Organizing Management Domains as Federations

CA*net 4 is a shared network used by all the provincial Optical Regional Advanced Networks (ORANs) across Canada. Its purpose is to interconnect each provincial ORAN and provide a set of wavelengths that can be shared among them. Even though CA*net 4 is one single autonomous system, the goal is to design a management and control system that acts as if CA*net 4 does not exist, and instead, thinks each ORAN is a management domain and provides the resources to interconnect each other. This is because the lifetime of CA*net 4 is only 5 years. After this time, it is expected that each ORAN will continue to share resources and operate without the help of CA*net 4. Because of this, we must associate each switch in CA*net 4 with the ORAN it is directly connected to as if it is actually owned and operated by that ORAN and not by CA*net 4.

It is for this reason that we are using the concept of "Federations". Each ORAN is a member of the same federation as the crossconnect switch that it uses to access CA*net 4. If more then one ORAN connects to CA*net 4 via the same switch, they are members of the same federation. It is also possible to have more then one switch in a single federation. Because each ORAN operates independently of the other, so does each federation.

Federations are not necessarily limited to one per ORAN. A federation could also be associated with an autonomous system. Since each ORAN has many different AS's within it, there is a possibility of having any multitude of federations.

## 4.2 System Architecture

The architecture of the lightpath management system, we call it a User-Controlled Lightpath Provisioning (UCLP) system, is shown in Figure 1. The figure shows the generic architecture and does not show the replication of the different system components in the different parts of the network. Typically, one instance of each of the components shown would exist in each Federation, however, they may also be shared. The JLS, JavaSpaces, Switch Communication Service (which interfaces to a single switch or a cloud of tightly coupled switches) and Grid SAP may run on different computers, while the Jini SAP and the LPO Service are downloaded to the process using them, in this case the Grid SAP.

This architecture uses the concept of service directories at two levels. First, internally, the JLS is used to find the different instances of switches and JavaSpaces in the network. Secondly, Grid SAP advertises its service instantiation through a well-known process described by OGSI implementation such as a Web Services Inspection Language (WSIL) pointer or Universal Description, Discovery and Integration (UDDI) database. The client communicates the user requests to the Grid SAP using the Simple Object Access Protocol (SOAP) protocol adopted for Web Services, and these requests are converted into Java procedure calls within the Grid SAP which then performs these calls on its local Jini SAP which executes these commands with the help of the other components within the system.

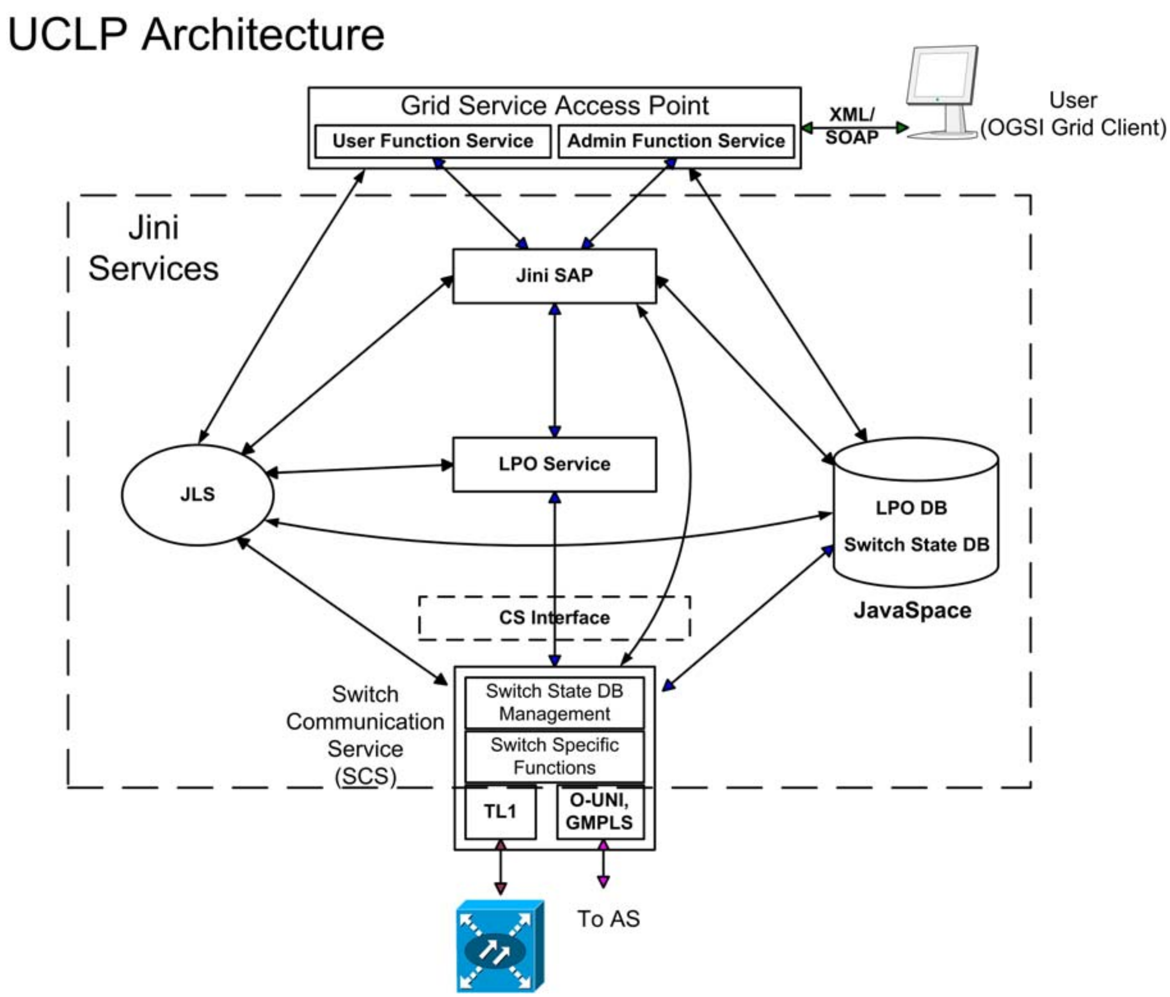


Figure 1. Overview of UCLP Architecture for CA*net 4

In a typical system configuration, each federation has its own set of services supported by the UCLP system (for short, we call them UCLP services), including its own JavaSpaces and JLS. Even though many services are accessible across federations, they are maintained independently of those in other federations. Even though we are promoting the sharing of resources among different domains, it is still important to maintain the administrative boundaries between each domain to avoid confusion about the ownership of assets and administrative privileges.

Jini and JavaSpaces address most networking and distributed system issues on top of the existing Java networking capability (sockets, remote method invocations, etc.). Firstly, Jini offers a mechanism for a customer to locate registered services. Customers interact with such services through Java's dynamic class loading. The lookup service returns a proxy object that corresponds to the service the customer was looking for. The whole network provides a lookup service. Multiple lookup services may be introduced for redundancy and fault tolerance. In the start-up process, customers obtain the lookup service either by pre-configuration or through a discovery protocol that uses multicast throughout the network. They look up registered JavaSpaces to locate the resources other customers offer. Access to JavaSpaces is controlled by user authentication and authorization. Thus, the customer can use a JavaSpace as if it was local, thereby freeing the system designer from considering the complex underlying networking protocols.

Secondly, Jini enhances network robustness and reliability. When a service is registered to the lookup service, the owner of the service sets a leasing period. The owner will keep renewing its leasing period for as long it is willing to continue providing the service. Should the service suffer failures and be unable to renew its lease, the service will expire. The lookup service will notify all customers having downloaded the proxy of the failed service, if the customers have registered for such events.

Thirdly, Jini and JavaSpaces provide the transaction facility mentioned above. For example, when a connection request needs multiple resources, this facility ensures all the required resources will be successfully taken or none of them will be taken if the request cannot be fulfilled. The transaction properties simplify the usage of advertised resources.

## 4.3 Lightpath Management Services

Figure 2 shows the system architecture in more detail and indicates for each of the components the main service methods provided by the component interface. At the user access level, we distinguish between the normal user who may invoke the ConnectionRequest method which creates a new end-to-end connection or the other methods indicated. The administrative users may in addition perform the administrative functions, such as displaying the complete map of the switch and network resources or the creation of a new basic optical link (provisioning) between two switches.

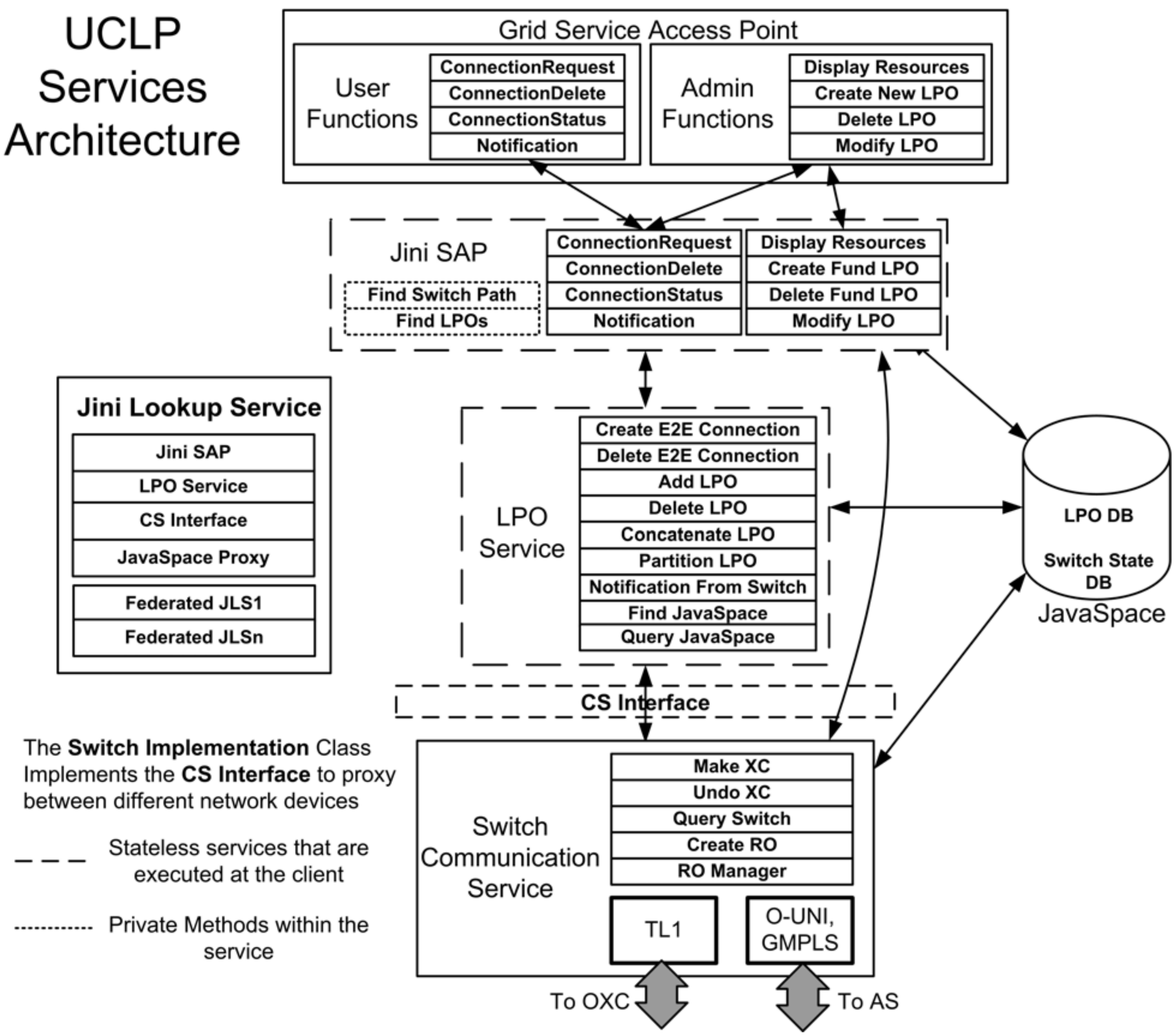

Figure 2. Detailed UCLP services architecture

In our design, LPOs are objects stored in JavaSpaces. An LPO is an abstraction of one or more lightpath segments. It is associated with a set of attributes and methods that enable possible peering to other LPOs at a switch to create an end-to-end connection or a longer lightpath segment. Supported customer operations on LPOs include: concatenating two LPOs, partitioning one LPO into many LPOs sharing common start and end points but with smaller bandwidth allocations, and reserving/using/releasing LPOs. The administrative operations include adding new LPOs and deleting LPOs corresponding to changes in the physical layer and the allocation of new resources.

For the execution of the connectRequest, the Jini SAP uses the internal methods FindSwitchPath and FindLPOs. The latter searches through the pertinent JavaSpaces to find LPOs with attributes suitable for the end-to-end connection to be built. It also uses the functions provided by the methods of the LPO Service.

## 4.4 Service Access Points

The management services provided by our system are classified into two groups: those only available to administrative users and those available to all users. The latter include in particular ConnectionRequest by which a user can request the establishment of an end-to-end connection from a given entry port of a given switch to a given exit port on another given switch, possibly belonging to a different federation. One of the functions reserved to administrative users is the addition of a new physical lightpath to the available optical network.

We have implemented these user services as methods of the so-called SAP-Service class which is registered in the JLS, as shown in Figure 2. Any Java application or user interface implementation could obtain a copy of this class from the Lookup service and execute it within its local Java Virtual Machine. However, in order to provide an interface for applications not using the Java environment, we have also provided an access point to these management services in the context of the evolving Web Services standards [7] which are based on XML and SOAP. In fact, we have used this SOAP interface to construct an interactive user interface which runs within the Web environment. It is organized as follows.

In general, the Web Services environment foresees concepts quite similar to those found in the Jini environment. The XML-based SOAP protocol is used for remote method invocation. There is also a service directory where server applications can register themselves by supplying the list of their services specified in terms of an XML schema thus defining the interface provided by the methods of the application. A client application would normally search this directory to find a corresponding server application and then access it for obtaining the desired service.

For our UCLP services, we have built two server applications providing the UCLP services for normal and administrative users, respectively. They run within a Web server which we call the Grid SAP. They are registered in the Web Services directory and accessible at a given URL. Our UCLP user interface client is implemented as a Java application that can be downloaded by users via Java Web Start from the same Web server that is the effective access point to our UCLP system. The downloaded user interface will communicate with an instance of the SAP-Services class in the Grid SAP in order to execute the commands received from the user. We note that the UCLP server applications perform the user authentication which will then be passed down to the lower layers of the UCLP system.

For the implementation of our UCLP Grid server and interface client, we have used the OGSI environment [8], which provides a convenient implementation environment for Grid services. In fact, given the interface definition of the SAP-Services in Java, the OGSI implementation tools automatically provide the corresponding XML schema defining the applicable SOAP protocol and the protocol encoding and decoding routines that are to be included in the code of the server and client side.

## 4.5 Path Search Strategies

As mentioned in Section 2.2, the visibility of lightpath resources belonging to different owners within a context where different parts of the network belong to different autonomous systems is one of the challenges of customer-managed end-to-end connections. In the traditional networking environment, routing protocols assemble routing tables that can be used to find available resources for routing a new connection through the given network. Standards for inter-domain routing for optical networks do not yet exist and full knowledge of network topology as normally used for intra-domain routing is not appropriate for customer-managed networking. In our current prototype, we have taken a quite simple and straightforward approach, as explained below.

We have adopted a two-step path search strategy. In the first step, for a given end-to-end connection request, we obtain from a static database the set of possible sequences of intermediate switches that are possible according to the physical interconnection structure of the entire network. We assume that this information is given in a form similar to the sequences of intermediate AS's in the BGP routing tables. In the second step, we try to find a lightpath along one of the switch sequences identified in the first step.

The second step of our search strategy is in general difficult to realize in larger networks, since we need access to up-to-date availability information for various resources throughout the network. One of the main issues is to find this up-to-date information which, in general, may be stored in various components of the distributed multi-domain network management system. In our design, we have decided to have one resource database per federation (autonomous domain) and the information about a lightpath is always stored in the database of the federation in which the source port of the lightpath resides. This convention allows us to determine in which JavaSpaces we have to search if we look for a lightpath from a given switch (belonging to a given federation) to some other switch. Based on this database partitioning scheme, we have developed path search algorithms with three levels of sophistication:

1) At the first level, we assume that a concatenated lightpath, which was created for establishing a given end-to-end connection, will be decomposed into its constituting fundamental (physical) lightpath when the end-to-end connection is deleted. This means that for establishing a new end-to-end connection, we only have available fundamental lightpaths to establish a new end-to-end lightpath.

2) At the second level, we assume that concatenated lighpaths may be available for use with new end-to-end connections. For instance, the temporary owner of a given lightpath may advertise it (in the JLS) as available for lease if it is not used any more. In this case, the path searching algorithm becomes more complex because it has to take the availability of such longer lightpath into account.

3) At the third level, the path searching algorithm may decompose an available longer lightpath into its components if this is helpful for finding the required end-to-end lightpath.

In our prototype, we have implemented a brute force algorithm for level (1) that searches along a sequence of switches obtained in the first step for lightpaths of suitable bandwidth. If such a sequence of fundamental lightpaths is found, they are concatenated to form a single end-to-end lightpath and the end-switches are configured to cross-connect the source port of the end-to-end connection to the source port of the end-to-end lightpath and the exit port of the connection to the destination port of the lightpath.

## 5. Conclusions

Customer-managed optical networks are useful in today's market place, where dynamic configuration of the network according to changing market is essential for business profits. It also finds significant opportunities in non-profit research and educational networks, where reduced operational cost is observed. Their technical challenges were discussed and the design of a system for user-controlled lightpath provisioning based on the software technology of Jini and JavaSpaces was presented.

## 6. References


1. J. Wu, J.M. Savoie, A. Vukovic and H. Hua, "Architecture, Routing and Signalling for Peer-To-Peer Inter-Domain Optical Networking", Proceedings of 2002 National Fiber Optic Engineers Conference (NFOEC), Dallas, Texas, USA, September 2002, pp.671-679
2. E. Freeman, et al. JavaSpaces Principles, Patterns, and Practice, Addison-Wesley Pub Co, 1999
3. S. Li, et al. Professional Jini, Wrox Press Inc, 2000
4. Jini Technology Core Platform Specification (Version 2.0 Beta), Sun Microsystems, Februray 2003
5. www.uddi.org
6. P. Hasselmeyer, "An Infrastructure for the Management of Dynamic Service Networks", IEEE Communications Magazine, Vol.41, No.4, April 2003, pp.120-126
7. www.w3c.org/ws
8. www.globus.org/ogsa