Lambda Data Grid:
Communications Architecture in Support of Grid Computing

by

Tal I. Lavian

B.S. Tel Aviv University 1988

M.S. Tel Aviv University 1997

A dissertation submitted in partial satisfaction of the

Requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

The dissertation of Tal I. Lavian is approved:

Chair ____ Date __________.

____ Date __________.

____ Date __________.

____ Date __________.

University of California, Berkeley

Fall 2006

Lambda Data Grid:
Communications Architecture in Support of Grid Computing

Abstract

Lambda Data Grid: Communications Architecture in Support of Grid Computing

By

Tal I. Lavian

Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Randy H. Katz Chair

The practice of science experienced a number of paradigm shifts in the 20th century, including the growth of large geographically dispersed teams and the use of simulations and computational science as a third branch, complementing theory and laboratory experiments. The recent exponential growth in network capacity, brought about by the rapid development of agile optical transport, is resulting in another such shift as the 21st century progresses. Essential to this new branch of e-Science applications is the capability of transferring immense amounts of data: dozens and hundreds of TeraBytes and even PetaBytes.

The invention of the transistor in 1947 at Bell Labs was the triggering event that led to the technology revolution of the 20th century. The completion of the Human Genome Project (HGP) in 2003 was the triggering event for the life science revolution of the 21st century. The understanding of the genome, DNA, proteins, and enzymes is prerequisite to modifying their properties and the advancement of systematic biology. Grid Computing has become the fundamental platform to conduct this e-Science research. Vast increases in data generation by e-Science applications, along with advances in computation, storage and communication, affect the nature of scientific research. During this decade, crossing the “Peta” line is expected: Petabyte in data size, Petaflop in CPU processing, and Petabit/s in network bandwidth.

Numerous challenges arise from a network with a capacity millions of times greater than the public Internet. Currently, the distribution of large amounts of data is restricted by the inherent bottleneck nature of today's public Internet architecture, which employs packet switching technologies. Bandwidth limitations of the Internet inhibit the advancement and utilization of new e-Science applications in Grid Computing. These emerging e-Science applications are evolving in data centers and clusters; however, the potential capability of a globally distributed system over long distances is yet to be realized.

Today's network orchestration of resources and services is done manually via multi-party conference calls, emails, yellow sticky notes, and reminder communications, all of which rely on human interaction to get results. The work in this thesis automates the orchestration of networks with other resources, better utilizing all resources in a time efficient manner. Automation allows for a vastly more comprehensive use of all components and removes human limitations from the process. We demonstrated automatic Lambda setting-up and tearing-down as part of application servers over MEMs testbed in Chicago metro area in a matter of seconds; and across domains, over transatlantic links in around minute.

The main goal of this thesis is to build a new grid-computing paradigm that fully harnesses the available communication infrastructure. An optical network functions as the third leg in orchestration with computation and storage. This tripod architecture becomes the foundation of global distribution of vast amounts of data in emerging e-Science applications.

A key investigation area of this thesis is the fundamental technologies that allow e-Science applications in Grid Virtual Organization (VO) to access abundant optical bandwidth through the new technology of Lambda on demand. This technology provides essential networking fundamentals that are presently missing from the Grid Computing environment. Further, this technology overcomes current bandwidth limitations, making VO a reality and consequentially removing some basic limitations to the growth of this new big science branch.

In this thesis, the Lambda Data Grid provides the knowledge plane that allows e-Science applications to transfer enormous amounts of data over a dedicated Lightpath, resulting in the true viability of global VO. This enhances science research by allowing large distributed teams to work efficiently, utilizing simulations and computational science as a third branch of research.

Professor Randy H. Katz, Chair

To my parent Sara and Rahamim,

My wife Ilana,

and my kids, Ofek, Aviv, and Ella

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, my undying gratitude to my wife, Ilana, for her support in this thesis, as in all aspects of life. Her multiple personal sacrifices and commitment to the management of three busy children with complicated schedules, made it possible for me to complete my Ph.D. studies. It wasn't easy to live with a guy who goes to sleep at 3:00am and does not wake up for the kids in the morning. You have done it all - cared for the kids, organized our social life, and maintained order in our home.

Ilana, thank you for enduring six tough years, it is time for some family fun.

I'd like to recognize my advisor, Professor Randy Katz. He has been outstanding in providing insightful feedback, and creating the perfect balance of encouragement and constructive criticism. By asking the essential research questions, he directed me to the underlying important concepts. Further, he helped me to winnow down the subject matter, zooming-in to address the in-depth challenges. Without his direction, I would still be swimming in an ocean of technical ideas and creative thought. He built structure into the procedural aspect that drove the project to completion. The circumstances of my life presented challenges that were unlike that of a young graduate student. He responded with consideration, taking into account my career, my family of five, and geographic distance. Huge thanks to him for his perseverance regarding my complicated administrative hurdles.

I'm grateful to my co-advisor, Dr. John Strand from AT&T Research. I met him when he was teaching an optical networks class during his Sabbatical at UC Berkeley. Over the years, he has provided the optical network perspective that was absolutely essential to the success of this research.

Special thanks to Professor Doan Hoang from University of Technology, Sydney Australia for his great friendship during the last six years. I have valued his companionship and his friendship that has stretched far beyond the duration of our research. This cross-continental friendship started with Professor Hoang's Sabbatical at UC Berkeley and Nortel Research Lab. Working with Doan was fruitful. It was great fun coming up with radical and often crazy ideas, some of which inspired the concepts presented in this thesis. I thank him for continuing our relation through countless international phone calls at odd hours, which were the vehicle for our productive collaboration. Through his interaction I was able to funnel broad-spectrum ideas into workable, researchable, and provable architecture. He had an unrelenting commitment to the project, including significant contribution to papers and publications, sometime under painful time limitations. In addition, he brought to the research environment a sense of fun, understanding, enthusiasm, inspiration and love.

The thesis began and ended with Nortel's support. As part of working in an industrial research lab, Nortel encouraged building creative, innovative, cutting-edge technologies that were the foundation of this thesis. Moreover, Nortel funded the facility, equipment, testing, and demonstrations. Dr. Dan Pitt, the Director of Nortel's Advanced Technology Lab, encouraged me to work on this Ph.D. As the head of Nortel's research lab, Ryan Stark encouraged me to collaborate with academia, publish results, and specifically work on this thesis. I owe a special debt of gratitude to my boss Dr. Franco Travostino, who was open to working with academia, and personally collaborated on publications. His gift for writing contributed largely to the clarity and eventual acceptance of many of our papers. He was patient in listening to my ideas, even when he believed some were impossible, crazy, science fiction, and gave me the freedom to prove that they were not. Franco let me juggle a lot of different responsibilities, and allowed me to commit myself to a variety of fascinating tasks.

The DWDM-RAM research was made possible by support from DARPA, award No. F30602-98-2-0194. The OMNInet/ODIN research was made possible by NSF funding, award No. ANI-0123399. I'd like to thank Paul Daspit from Nortel for his great support of this research. He was the administrative manager for the DWDM-RAM project at Nortel and conducted all the interaction with DARPA. His presence in the weekly meetings and his implementation of project management, including strict discipline on deadlines, were very productive for the deliverables of demonstrations. Thanks to the rest of the team for the long hours working toward solutions, participating in meetings, and building the demonstrations. I'd like to recognize the team: Dr. Phil Wang, Inder Monga, Ramesh Durairaj, Howard Cohen, Doug Cutrell, Steve Merrill. My appreciation to the Ph.D. students for their internship at Nortel: David Gutierrez, Sumit Naiksatam, Neena Kaushik and Professor Silvia Figueira for her input and review of publications. My gratitude to Professor Cees De Latt and his students for the collaboration on the transatlantic demonstration. Thanks to Beth DeGolia for proofreading this thesis, which helped to significantly improve the presentation.

Special thanks to Professor Joe Membretti and his students at Northwestern University. As the Director of ICARE, Professor Mambretti manages the OMNInet testbed in the Chicago metro area and allowed some parts of the DWDM-RAM project to be run on his testbed. He was instrumental in getting the project off the ground. He continued to be the fundamental link to other fields in science.

1 Introduction and Preview

1.1 Motivation

1.1.1 New e-Science and its distributed architecture limitations

Science is at the early stages of multiple revolutions spawned by the intersection of novice scientific research methods and emerging e-Science computation technologies [1]. Many new and powerful methodologies are evolving within the specialties of biology, health science, genomics, physics, astrophysics, earth science and environmental science. Contemporary science is advancing at an unprecedented rate, much faster than at any other time in history. This acceleration holds the promise of radical breakthroughs in many disciplines.

New types of applications are emerging to accommodate widely distributed research teams that are using computation-based simulations as the third scientific branch, complementing conventional theory and laboratory experiments. These applications require the orchestration of the right data, to the right computation, at the right time. Crucial is the distribution of information and data over time, space, and organizations. For these applications, the transfer of immense amounts of data is becoming increasingly necessary. However, major limitations arise from existing distributed systems architecture, due to the inability to transfer enormous amounts of data.

Our mission was to build an architecture that can orchestrate network resources in conjunction with computation, data, storage, visualization, and unique sensors. In simple terms, it is the creation of an effective network orchestration for e-Science applications, with vastly more capability than the public Internet. To realize this mission, some fundamental problems faced by e-Science research today require a solution.

1.1.2 The Peta Lines

Due to advances in computation, storage, scientific data generation, and communication, we are getting close to crossing, or are crossing the Peta (10^{15}) line in storage size, communication speed and computation rate. Several high level US Department of Energy (DOE) labs have built Petabyte storage systems, and there are some scientific databases that have exceeded one PetaByte. While high-end super-computer centers are presently operating in the range of 0.1-1 Petaflops, they will cross the Petaflop line in a matter of years. Early optical lab transmission experiments are in the range of 0.01-0.1 Petabits/s, and by the end of this decade, they will cross the Petabits/s line [2].

1.1.3 Gilder and Moore – Impact on the Future of Computing

The principles of both Gilder and Moore are important phenomena that must be considered juxtaposed to Grid Computing infrastructure in new e-Science research. Moore's Law [3] predicts doubling silicon density every 18 months. In early 2000, a common misconception held that traffic was doubling every three months. Andrew Odlyzko and Kerry Coffman [4] showed that this was not the case . He demonstrated that traffic has been approximately doubling every 12 months since 1997 Based on progress in optical bandwidth. Gilder's Law [5] predicts that the total capacity of optical transport systems doubles every six months. New developments seem to confirm that optical transport bandwidth availability doubles every nine months.

This difference between Moore and Odlyzko may look insignificant at a glance. However, see figure 1.1, where the calculation over time shows that the gap between computation and traffic growth is x4 in six years, x16 in 12 years, and x32 in 15 years. When comparing to optical transport growth, the difference is even more impressive. The impact of this phenomenon drives us to rethink the fundamentals of computation, storage, and optical transmission in regards to Grid infrastructure for e-Science research. Traditional data-intensive and compute-intensive approaches requires a new balance in the areas of distributed systems, remote storage, moving data to computers, moving computation to the data, storage, and remote data processing. This substantial gap in favor of optical transmission compared to computation inspires one to re-examine traditional computer science assumptions in reference to Grid Computing.

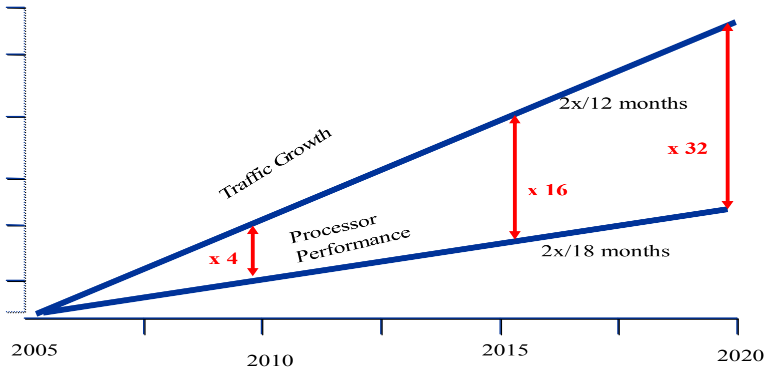


Fig 1.1 Processor vs. Traffic Growth

1.2 Transmission Mismatch

Recent advances in optical transport technologies have created a radical mismatch in networking between the optical transmission world and the electrical forwarding/routing world. Today, a single strand of optical fiber can transmit more traffic than the entire Internet core. However, end-systems with Data Intensive Applications do not have access to this abundant bandwidth. Furthermore, even though disk costs are attractively inexpensive, the feasibility of transmitting huge amounts of data is limited. The encumbrance lies in the limited transmission ability of Layer 3 (L3) architecture. In the OSI model [6], L3 provides switching and routing technologies, mainly as packet switching, creating logical paths known as virtual circuits for transmission of data from node to node. L3 cannot effectively transmit PetaBytes or hundreds of Terabytes, and has impeding limitations in providing service to our targeted e-Science applications. Disk transfer speed is fundamentally slower than the network. For very large data sets, access time is insignificant and remote memory access is faster than local disk access.

Figure 1.2 represents the conceptual limitation that Lambda Data Grid is addressing between the requirements of Data-Intensive applications and the availability of optical bandwidth. The significant imbalance between e-Science applications requirements and the available resources to support them in today's technologies motivates us to build a resource orchestration architecture integrating Grid Computing and optical networks.

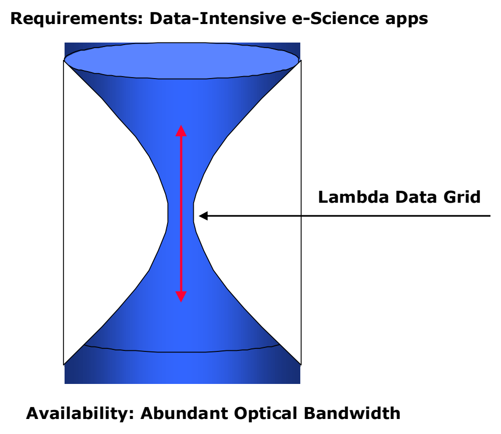

Requirements: Data-Intensive e-Science apps
Lambda Data Grid
Availability: Abundant Optical Bandwidth

Figure 1.2 – Transmission Obstacle for e-Science Applications

1.3 Limitations of L3 and Public Networks for Data Intensive e-Science

There are three fundamental technological choices to address when finding solutions for Data Intensive Applications.

- Packet switching vs. Circuit switching
- Public Internet vs. Private connection (shared vs. dedicated)
- L3 vs. L1 functionalities

The obvious solutions use existing technologies like L3, routing mechanisms, and the public internet for large data sets of e-Science research. However, limitations embedded in these technologies make these solutions less effective. In the age-old question of using packet switching vs. circuit switching, historically packet switching won. Within the context of large data sets, this question must be examined again [7]. In our targeted area, L1 circuit switching to limited address space is more effective than L3 packet switching to large address space. The original Internet Design Principles provides a different set of criteria for low bandwidth supply, and does not perform optimally in e-Science. Routing and L3 works well for small packets and short durations, but lose their effectiveness for large data sets and long durations. In L3 mechanisms, look-ups are performed for large data streams. This is no longer required when the destination is known in advance, saving billions of identical forwarding decisions in large data sets. On the shared public Internet, fairness is important and therefore considered in networking protocols. In dedicated private network, fairness is not an issue. The above ideological differences are also discussed in detail in Chapter 3.

1.4 e-Science

The CyberInfrastructure Council of the National Science Foundation (NSF) is working on "CyberInfrastructure Vision for 21st Century Discovery. [8]" Figure 1.3 is a excerpt from this document. In a 2006 working draft of this document, some important questions are addressed.

How does a protein fold? What happens to space-time when two black holes collide? What impact does species gene flow have on an ecological community? What are the key factors that drive climate change? Did one of the trillions of collisions at the Large Hadron Collider produce a Higgs boson, the dark matter particle or a black hole? Can we create an individualized model of each human being for targeted healthcare delivery? How does major technological change affect human behavior and structure complex social relationships? What answers will we find – to questions we have yet to ask – in the very large datasets that are being produced by telescopes, sensor networks, and other experimental facilities? These questions – and many others – are only now coming within our ability to answer because of advances in computing and related information technology.

Figure 1.3– Excerpt from NSF's CyberInfrastructure draft Vision for the 21st Century Discovery

The overarching infrastructure vision for the 21st Century is hugely complex. This thesis, "Lambda Data Grid: Communication Architecture in Support of Grid Computing," is a solution for one small aspect of this infrastructure. The primary motivation of this thesis originated from the e-Science CyberInfrastructure quandary.

Dictated by its unique requirements, e-Science has massive middleware designed to carry out the distribution of resources and data across the globe, and to facilitate the scientific collaboration. Data Intensive Applications (DIA) and Compute Intensive Applications are expected to grow at a rapid pace in the next decade. The network research community focuses on Internet-sized scaling and has not been pushed to anticipate the massive predicted scale of these e-Science applications. The data generated annually by e-Science experiments is of several magnitudes larger than the entire current Internet traffic, with an expected growth pace that reaches many orders of magnitude beyond. It is difficult to comprehend these sizes because it is so vastly beyond the scope of our networking experience.

To illustrate the scale of data, collaboration and computation being considered, we present below a small sample of e-Science projects, taken from the very large number of projects in progress.

High Energy Physics (HEP) – Finding 'Higgs' particle associated with mass is one of HEP's primary research goals. CERN's Large Hadron Collider (LHC) [9] involves about 5,000 researchers from 150 institutions across the globe. The distributed nature of the research requires collaboration with institutes like Stanford Linear Accelerator Center (SLAC) [10], Collider Detector at Fermilab [11] , (CDF) [11], Ring Imaging Cherenkov Detector (RICH) [12] , and Lawrence Berkeley National Labratory (LBNL) [13]. The BaBar [14] project at SLAC [15] generated over one Petabyte (1PB = 10^{15} Bytes) of data since its inception. The new SLAC collider in construction will generate one Exabyte (1EB = 10^{18} Bytes) during the next decade. CERN's LHC data store consisted of several Petabytes in 2005, with staggering growth expectancy to about 100 Petabytes by 2008. The HEP is the biggest research effort on earth in terms of data and computation sizes. Efficiency in moving and accessing the data associated with this research could be the networking quandary of the millennium. This thesis work attempts to make significant steps toward addressing and overcoming a small portion of the networking challenges for this type of project.

- **Astrophysics**- Of the many experiments in National Virtual Observatories, the NVO [16] project, generated 500 Terabytes of data in 2004. The Laser Interferometer Gravitational Wave Observatory (LIGO) [17] project generated 250 Terabytes, and the VISTA project generated 250 Terabytes. By 2015, the VISTA project alone will generate several Petabytes of data annually.

- **Environment Science** – The European center for Medium Range Weather Forecasting (ECMWF) holds about 330 Terabytes of data. The Eros Data Center (EDC) [18] holds about three Petabytes of data, the Goddard Space Flight Center (GSFC) [19] holds about 1.5 Petabyte of data, and it is estimated that NASA will hold about 15 Petabytes of data by 2008.

- **Life Science** – Protein Data Bank (PDB) [20], protein sequences, Bioinformatics sequence databases, and Gene Expression Databases compose a tiny portion of the many growing databases in life science. Gene expression experiments are conducted by hundreds of institutes and laboratories worldwide. The data range in 2005 was several Petabytes, approximately. The National Institute of Health (NIH) [21] is helping to fund a handful of experimental facilities with online accessibility, where experiments will generate data sets ranging from hundreds of Terabytes to tens of Petabytes. Bioinformatics research requires massive amounts of computation in the order of approximately hundreds of Petaflops per second. The computation required in the gene sequencing of one gene takes the work of about 800 computers for one year.

As one digests the size and scope of these projects, it is obvious that the current networking technologies are inadequate. There are great efforts towards middleware advancements in various research fields. Many middleware projects [22] [23] have adopted Grid technologies, Workflow, and Web Services [24]. These projects require solving tough problems in regards to collaboration, distribution, sharing resources, access to specific data, sharing results, and accessing remote computation or storage. Ultimately, middleware is the key to the success or failure of Grid technologies in Data Intensive Applications and Compute Intensive

Applications. There are substantial efforts aimed at middleware development in this new era of research. Projects like ENLIGHTENED [25] , Tera Path, and Oscars focus on some aspects of Grid network middleware. Our contribution to this effort as presented in this work is as follows: The building of middleware and architecture for the orchestration of network resources that are plugged into the broader scope middleware development effort, interact between application middleware and network middleware, allowing scientific research communities to work efficiently with large data sets.

1.5 Dissertation Overview

This dissertation provides an architecture, design, evaluation, and prototype implementation for wide-area data sharing across optical networks of the 21st century. It provides the middleware services necessary to orchestrate optical network resources in conjunction with other resources. Today, Grid Computing applications and Workflows used by e-Science applications can allocate storage, data, computation, unique sensors, and visualization. This work will enable the orchestration of optical services as an integral part of Scientific Workflows and Grid Computing middleware. As this work progressed, it was prototyped and presented at GlobusWorld [26] in San Francisco, GGF-9 Chicago, Super Computing [27] in Pittsburgh, and GlobusWorld in Boston.

Each demonstration received enthusiasm, comments, and suggestions from the research community, inspiring further work, and resulting in a more advanced prototype for each convention. Each progressive prototype highlighted the expanded capabilities for the technology.

The first prototype in San Francisco demonstrated the basic proof of concept between four nodes, mounted on two separate racks, over a distance of about 10 meters. At GGF-9, we showed the basic architecture with implemented Grid Services, which dynamically allocated 10Gbs Lambdas over four sites in the Chicago metro area, covering a distance of about 10km. This was significant because it showed the core concept on a small scale. We demonstrated how an application expresses a need for service via Grid Service. This prototype of basic network intelligence supported the mechanisms. More important was the novelty of interaction between application and network to accomplish a common goal.

Further and enhanced architecture resulted in the Pittsburgh's prototype, in which we built the Grid middleware for the allocation and recovery of Lambdas between Amsterdam and Chicago, via NY and Canada, over a distance of about 10,000km. Real-time output results and measurements were presented on the floor at Super Computing [27] in Pittsburg. This prototype demonstrated the reservation and allocation of Lambda in about 100 seconds compared to the manual allocation of about 100 days (phone calls, emails, personnel scheduling, manual network design and connection, organizational priorities, legal and management involvement). The computational middleware was able to reduce the allocation time from months to seconds, allowing integration of these interfaces to Grid Computing applications and e-Science Workflows. Shifting from manual allocation to an automated computational reservation system and allocation via the Grid Web Services model opens the door for endless advancements in scientific research.

1.6 Preview: Three Fundamental Challenges

In this section, we will preview three fundamental challenges that this dissertation addresses. The nature of new e-Science research requires middleware, Scientific Workflows, and Grid Computing in a distributed computational environment. This necessitates collaboration between independent research organizations to create a Grid Virtual Organization (VO) [28] [29]

. Each VO addresses organization needs across a large scale geographically dispersed area, and requires the network to function as a fundamental resource. The Grid research community has been addressing many challenges in computation, storage, security, and management, but has failed to successfully address some of the inherent insufficiency of today's public network, as we will show in chapter 3. In this work, three challenges became evident:

1) Limitations in packet switching for Data Intensive Applications and Compute Intensive Applications, over a distant network such as WAN.

2) The Network Resources need for allocation, scheduling and management by Grid Computing middleware rather than statically by network administrators.

3) The transfer management of multi-terabytes of data, in a specific time window, at requested locations.

In this thesis, we analyze these problems in detail and solve some of them by building a new network middleware that is integral to Grid middleware to manage dedicated optical networks. In simple terms, we built a special network just for e-Science. The following is a summary of the problems and a discussion of our solutions.

1.7 Challenge #1: Packet Switching – an Inefficient Solution for Data Intensive Applications

1.7.1 Elephants and Mice

Packets are appropriate for small amounts of data like web pages and email. However, they are far from optimal for e-Science applications similar to Visual Observatories, for example, that will generate Petabytes of data annually in the next decade. Basic Ethernet frame size is 1.5KB or 9KB in the case of Jumbo Frames. L3 data transfer of 1.5TB will require one billion (10^9) identical packet header lookups. Moving a data repository of 100TB is one trillion times greater than moving a web page of 100KB. This would be much like CANARIE's [30] analogy of transferring a herd of elephants compared to a family of mice. It is simply impossible to transfer elephant-sized data on the today's public Internet using L3 packet switching. Such an attempt would vastly destabilize Internet traffic. It is necessary to question the usefulness of current methodologies when dealing with nine-orders of magnitude difference in transfer size.

1.7.2 Lightpath Cut-Through

The evolution of e-Science and its demand for the transfer of bulk data challenge us to examine the scalability of data transfer at the core of the Internet. Effective cut-through methods are relevant. The integrity of end-systems and the edge devices must remain intact, but the underlying optical core demands rethinking. Upper level L3 packet switching infrastructure continues to serve as the main method for small traffic transfers, while a separate underlying optical channel is necessary for scalable transfer. One of the significant challenges is the interoperability of these two systems. The underlying optical infrastructure is based on lightpath, similar to the L0-L1 circuit switching mechanisms, while the functionality at L3 is based on packet switching/routing. In the OSI model [6], Layer 1 or the physical layer, conveys the bit stream through the network at the electrical and mechanical level. Layer 0 represents the photon stream in optical media on a single Lambda. We have built a system that diverts e-Science bulk traffic via a cut-through over an underlying, dedicated, optical network instead of the public Internet.

1.7.3 Statistical Multiplexing

Statistical multiplexing can work for many-to-many small traffic patterns, as found in today's Internet. For few-to-few bulk traffic patterns, as seen in astrophysics research, statistical multiplexing loses its benefits. Distribution of traffic flow is most effective for many small flows. The assumption that the network is always available for the transmission of minute flows, compared to its overall capacity, is reasonable. Roughly, an average of 50%-60% utilization of the network capacity is desirable for network transmission and can maintain peak-time traffic that edges towards 100% capacity. Higher average utilization places the network at risk. In the case of very large data transfers, the network requires 100% of capacity for a period of time. For the traffic pattern represented in large data transfers as required by e-Science, average utilization cannot be more than a few percent. To handle multiplexing of "elephant" sized data flow in an acceptable time, substantial additional capacity is required, where utilization must be very low to handle the traffic.

1.7.4 Availability Expectations

For normal use of the public internet, the users assume constant availability. In most cases, there are no mechanisms to regulate when to use the network, under what conditions, and how much data to transmit. This is efficient because the "normal" transfer data size is a tiny fraction of the available bandwidth at the core. Conversely, for the HEP applications, the requirements are to use 100% of the capacity of the line at a specific time, without sharing the network, and with no multiplexing. This changes the assumption of network availability such that the network will be available and scheduled upon request, and transmission approved only upon permission granted.

1.7.5 Bandwidth and Bottlenecks

In the last 30 years, the research and the industry have held the fundamental design principle that bandwidth is limited and conservation is necessary. Substantial research efforts have looked at ways to optimize bandwidth and conserve traffic, for example, by using data compression. Extensive work has been devoted to applications and protocol stacks to compensate for restrictions in bandwidth. Advances in optical networking and its associated low transmission errors raise a new series of questions: Why do we need so many applications optimized to conserve bandwidth? Can we build these types of Data Intensive Applications differently if bandwidth is free? [31] Is optimization of the network still necessary?

1.7.6 Why not Lightpath (circuit) Switching?

The nearly universal decision to use packet switching rather than circuit switching was arrived at for a variety of good reasons. However in optical switching, utilization, bandwidth optimization, conservation and transmission costs, are not primary goals. With new tradeoffs for large data sets, Lightpath is the optimized solution. Lightpath is actually an implementation of circuit switching in wavelengths. Another approach is Optical Burst Switching (OBS) , which functions under a similar concept of dedicated circuit, but with an extremely fast set-up and for short durations.

1.8 Challenge #2: Grid Computing Managed Network Resources

Typical Grid applications require the management of highly distributed resources within dynamic environments. Basic problems related to these requirements are common to almost all Grid environments, e.g., matching multiple and potentially conflicting application requirements to diverse, distributed resources within a dynamic environment. Other problems are more specific to addressing the complexity of utilizing methods for data provisioning for large-scale data flows [32].

1.8.1 Abstract and Encapsulate

To satisfy various complex patterns of application demands as seen in Section 1.3, it is necessary to abstract and encapsulate the network resources into a set of Grid services that can provide scheduling, monitoring, and fair-shared usage within a service platform. Understanding these application requirements and providing the intelligence to respond to them are all standard issues that can be addressed within the context of Web Services Resource Framework (WSRF) Grid middleware [33]. A network resource service is a key component that we implemented to realize abstraction and encapsulation of network resources.

1.8.2 Grid Networking

Common architectures that underlie traditional data networks do not incorporate all of the capabilities required by Grids. They are generally designed to optimize the relatively small data flow requirements of consumer services and the managed services of enterprises on a common core infrastructure oriented to the requirements of general common communication services. Many Grid applications are data-intensive, requiring specialized services and infrastructure to manage multiple, large-scale data flows of multiple Terabytes and even Petabytes, in an efficient manner. Such capabilities are not effectively possible in traditional routed packet data networks.

1.8.3 Grid Middleware for Dynamic Optical Path Provisioning

It is necessary to provide applications with a direct, flexible access to a wide range of optical infrastructure services, including those for dynamically provisioned optical path channels within an agile optical network. There is a need to design network architectures that can support Grid applications in association with emerging optical networks. The architectures must integrate network requirements with optical control utilizing new techniques for dynamic optical path provisioning. Such networks have the potential to provide instantaneously provisioned, high performance bandwidth with capacity several orders of magnitude beyond that of today's networks.

1.8.4 Virtual Organization As Reality

Of primary importance is the creation of a technology that allows Virtual Organization (VO)[1] to access abundant optical bandwidth using Lambda on demand to Data Intensive Applications and Compute-Intensive Applications. This provides essential networking fundamentals that are presently missing from Grid Computing research and overcome bandwidth limitations to help make VO a reality [2].

1.9 Challenge #3: Manage BIG Data Transfer for e-Science

To function effectively, e-Science researchers must access massive amounts of data in remote locations. From massive, one-of-a-kind, real-time remote sensors, or from immense remote storages, researchers must filter the data, and transfer minuscule portions for their use. The challenge is to get the right data to the right location, at the right time.

Much of science is experimental, with data being gathered using increasingly sophisticated and expensive instruments. These may range from electron microscopes to distributed arrays of seismometers to astronomical interferometric arrays of radio telescopes to high-energy physics particle colliders and their detectors to space-based remote sensing satellites to nuclear fusion facilities, and so on. Heretofore, a scientist wishing to conduct experiments would need to go to a particular facility, directly control the apparatus, gather data, return to the home institution for the analysis phase, decide on the focus for follow on experiments or observations, and repeat the cycle. An added complication to this process is the competition for scarce resources.

If network connectivity were available to access the right data, and to transfer it to the right location at the right time, then the experimental control could be done from the home institution. In addition, the data could be gathered at real time from the facility to the home institution where the emerging Grid paradigm of fast and distributed computation could facilitate early preliminary analysis and visualization. This would guide modifications to the experimental setup, greatly shorten the cycle time, and increase the efficiency of specialized facilities.

In the world of scientific research, collaboration and sharing information is crucial for scientific advances. Limitations in technology and the inability to orchestrate resources prohibit the usability of one-of-a-kind facilities and/or instruments by the wider community of researchers.

Further, non-experimental work could benefit from very high capacity networking. Consider, for example, interlinked models used for climate simulation. There might be an atmospheric model that interacts with an oceanic model as well as with a solar model to address how radiation flux and solar storms affect the upper atmosphere. Econometric models could look at how climate will affect land use patterns, agriculture, etc. and how it might feed back into atmospheric effects. Each simulation would run at its own center of expertise, requiring high-speed data connections to communicate at each time step.

1.9.1 Visualization Example

One of the primary means of image appreciation and integration of data sets is through visualization, with its trend toward higher and higher resolution displays and real-time interaction with multi-dimensional data. These displays may be driven by arrays of special purpose rendering computers working in parallel and sending pixels to a set of display devices also working in concert, to provide a large-scale unified environment, such as a CAVE or wall sized display.

What are the data rates needed for this, even within a single laboratory? Consider a 100 Mega pixel display, which is already available as a prototype[2]. Calculation of bandwidth requirements for this visualization device shows the need for 72 Gbps in the OptIPuter [34] 8 nodes cluster, or 576Gbps for 64 nodes cluster. This is over half a Terabit per second.

This is clearly beyond the range of today's standard 1-10 Gigabit Ethernets but is achievable [35] using display technologies by EVL. These technologies can also allow the user to view all or parts of this display from a remote site, given the local display hardware. That is, the engines that analyze the data and create the visualization can be remote.

1.10 Major Contributions

1.10.1 Promote the Network to a First Class Resource Citizen

- The network is no longer a pipe, but rather a part of the Grid Computing instrumentation. In addition, it is not only an essential component of the Grid computing infrastructure but also an integral part of Grid applications. This is a new design principle for Grid and high-throughput Computing. The proposed design of VO [36] in a Grid Computing environment is accomplished with Lightpath as the vehicle, allowing dynamic Lightpath connectivity while matching multiple and potentially conflicting application requirements, and addressing diverse distributed resources within a dynamic environment.

1.10.2 Abstract and Encapsulate the Network Resources into a Set of Grid Services

- Encapsulation of Lightpath and connection-oriented, end-to-end network resources into a stateful Grid service, while enabling on-demand, advanced reservation, and scheduled network services. This works within a schema where abstractions are progressively and rigorously redefined at each layer. This helps to avoid propagation of non-portable implementation-specific details between layers. The resulting schema of abstractions has general applicability.

1.10.3 Orchestrate End-to-End Resources

- A key innovation is the ability to orchestrate heterogeneous communications resources among applications, computation, and storage, across network technologies and administration domains.

1.10.4 Schedule Network Resources

- The assumption that the network is available at all times, to any destination, is no longer accurate when dealing with big pipes. Statistical multiplexing will not work in cases of few-to-few immense data transfers. We have built and demonstrated a system that allocates the network resources based on availability and scheduling of full pipes.

1.10.5 Design and Implement an Optical Grid Prototype

- We were able to demonstrate dynamic provisioning of 10Gbs in 100 seconds, replacing the standard provisioning of approximately 100 days. This was shown in a connection from Amsterdam to Chicago during Super Computing and on the conference floor in Pittsburg. For technology demonstrations, Cees De Latt [37] described the previous standard process of provisioning 10Gbs from Amsterdam to Chicago in general terms as follows: “It took about 300 emails, 30 conference and phone call and three months to provision the link”. Provisioning has vastly improved thanks to new Lambda service, which takes only a few dozen seconds to create an OC-192 coast-to-coast, compared to the three to six months it takes commercially.

1.11 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2 discusses the background material in the area of Grid Computing, Service Oriented Architecture (SOA), Web Services Resource Framework (WSRF), and related work over middleware for e-Science applications.

Chapter 3 analyzes bulk data transfer and optical networking, in view of limitations for geographically dispersed research collaboration in e-Science. This chapter discusses the limitations of packet switching, routing, Layer 3 technologies and the public Internet for collaborative research, and overviews the need for Lambda Data Grid as part of Cyber-Infrastructure.

Chapter 4 presents a bird's eye view of Lambda Data Grid as a network solution for the scientific research targeted in this thesis. The network needs of some e-Science projects are discussed along with the Lambda Data Grid solution as part of orchestration network resources over GLIF.

Chapter 5 looks at Lambda Data Grid Architecture, and our service orchestration framework. Specifically the Data Transfer Service (DTS) and the Network Resource Service (NRS) is outlined. A correlation is built between Grid Layer Architecture and our layer architecture. Our middleware design is presented as an easy interface between scientific middleware packages, and Scientific Workflows.

Chapter 6 introduces two new concepts: "time-window" and "time-value" for Lightpath scheduling. The data is analyzed, reflecting on the need for Lightpath allocation and reservation for Data Intensive Applications and Compute Intensive Applications.

Chapter 7 conceptualizes time-window and time value through a distributed algorithm for network path allocations. Time-window requests by the applications, the time proposal/response, along with feedback is used to determine a time-value per proposal. Two new entities are presented as part of our middleware: Segment Reservation Authority (SRA) and Path Reservation Authority (PRA). These are designed to fulfill requests within presenting constraints.

Chapter 8 explains our testbeds, public demonstrations, and the experimental networks used in this work. The results of several experiments demonstrating the value of Lightpath allocation is described, along with an evaluation of our results. Also presented is the dynamic allocation of bandwidth on demand, and the radical improvement in trans-Atlantic traffic recovery, from months to minutes. This new capability is analyzed within the context of remote collaboration and federation enabling for e-Science applications.

Finally**, Chapter 9** is a summary of our work, quantitative results, and its contributions. The conclusion of the chapter and this thesis is a discussion of general lessons learned about orchestration of network resources in conjunction with other resources; the interaction between network middleware and Grid middleware as used in Scientific Workflows, and an outline of some directions for future research.

2 Background and Related Work

This chapter introduces the background and related work concerning the investigation areas of this thesis: e-Science applications and resource requirements, Grid computing infrastructure, Grid services orchestration [32], optics-based agile Grid networks, and end-to-end transport mechanisms for data intensive applications.

2.1 Introduction

Important advancements are underway to accommodate scientific research and new e-Science computation technologies. Collaborations among geographically dispersed research groups is possible by new computation technique that allow new types of research applications to be utilized. This chapter summarizes a variety of requirements of e-Science applications that involve data intensive transportations and communications.

The completion of the Human Genome Project (HGP) [38] in 2003 was the triggering event for the life science revolution of the 21st century. Molecular and cellular biologists have been working to uncover how living system works. The understanding of the genome, DNA, proteins, and enzymes is prerequisite to modifying their properties and building Systematic Biology. Computation and information technologies are the major tools for these scientific discoveries. The Mouse BIRN project [39] discussed in this thesis is one example of research in this area.

New service architectures have been proposed and developed for Grid computing to support these new types of e-Science applications. OGSA (Open Grid Service Architecture) [28] under the Globus Grid Forum (GGF) [29] is implemented as a service architecture by the Globus

Project [40], and offers a number of rich features that previous service architectures do not. This chapter provides a brief overview of OGSA and its associated services. These features include standardizing processes such as advertising what capabilities a computer has or the governing of who has permission to use a certain service on the network.

Also discussed are Grid computing limitations, e-Science applications and services, and state of the art developments. We further discuss related work and point out how our research on the DWDM-RAM project is distinguished from previous work. I was the Principal Investigator for this DARPA funded project, and the inspiration for many of the significant innovations contained in this thesis.

2.2 Mouse BIRN – explosive data generation

The Mouse BIRN [39] aims to share and analyze multi-scale structural and functional data and ultimately to integrate them with genomic and gene expression data in the mouse brain. Ongoing collaborations between six institutions in the US are looking at neurological disorders in mice to determine relevancy to schizophrenia, Parkinson's disease, brain cancer, substance abuse, and multiple sclerosis.

Application Scenario	**Current MOP**	**Network Issues**
Point-to-Point data transfer of multi-TB data sets	Copy from remote DB: Takes ~10 days (unpredictable) Store then copy/analyze	Request to receive data from 1 hour up to 1 day Innovation for new bio-science Architecture forces optimization of BW utilization Tradeoff between BW and storage
Access multiple remote DB	N* previous scenario	Simultaneous connectivity to multiple sites Multi-domain Dynamic connectivity hard to manage Unknown sequence of connections
Remote instrument access (Radio-telescope)	Personnel required at remote location	Require fat unidirectional pipes Tight QoS requirements (jitter, delay, data loss)

Table 2.1 – Potential Mouse BIRN requirements

Table 2.1 illustrates some of the challenges in incorporating data across the different geographically disperse institutes. The first application scenario presents point-to-point data transfer of multi-TeraBytes data sets, in unpredictable lengths of time. The main challenge arises from dynamic access of multiple remote databases. This requires synchronization of software modules, information components, data locations, storage management and computation modules, working harmoniously where the network is part of the underlying infrastructure.

Multi-scale collaboration between different scientific disciplines is required. The interaction between the biology images, Vivo Microscopy images, and neuro-imaging present a fundamental data exchange challenge. Incorporation analysis and processing of data must be done across computation models including chemistry and genomics, and across levels: molecular, nuclear, cellular, tissue, and organs. Each discipline uses unique computation models, software, packages, analysis, and tools. It is not enough to simply move the data. The line between computation and communication must be blurred. Dynamic network allocation as part of Grid Workflow allows for new scientific experiments that are not possible with today's static allocation [41].

The Mouse BIRN research is experiencing limitations in its computation/ communication model:

- Tradeoff between computation, communication and data is not always clear
- Delays preclude interactive research: copy, then analyze
- Uncertain transport times force a sequential process
- Must schedule processing after data has arrived
- No cooperation/interaction among storage, computation and network middleware

To address these issues The BIRN research community utilized Service Oriented Architecture (SOA).

2.3 Network, not Computers - Central to e-Science Applications

High Energy Physics (HEP) is a major research effort sponsored by the United States Department of Energy (DoE). The HEP community in general, supports a vast portfolio of applied and basic research in the physical sciences, as well as ancillary research and development activities that support primary scientific endeavors. The planning, execution and analysis of these research programs require the coordination of large, distributed teams of individuals and institutions. The archetypical enterprise is the HEP program, although other DoE sponsored areas [2], such as those listed in Table 2.4, present many of the same issues.

	Estimated 2008 Data Generation
CEBAF (Hadron structure experiments)	<10 PB/year
RHIC (Quark-gluon plasma experiments)	5 PB/year
CERN LHC (Higgs boson search)	10 PB/year
SLAC (BaBar experiments)	1 PB/year
Nuclear physics	3 PB/Year
Plasma physics	5 PB/Year
Magnetic fusion	1 PB/year
Computational fluid dynamics	2 PB/Year

Table 2.3 : DoE Sponsored data intensive research areas [2] (theoretical and experimental).

Consider the Compact Muon Solenoid (CMS) experiment built as part of the Large Hadron Collider (LHC) at CERN. This project involves 31 nations, 150 institutions, and over 1800 scientists. The Atlas detector at the LHC, expected to come on line with the CMS in 2007, should generate 10 petabytes (1 PB = 10^{15} bytes) of data in its first year of operation, and grow to about 1 exabyte per year (1 EB = 1000 PB = 10^{18} bytes) by 2015. These requirements are for highly distributed teams and huge data requirements.

This massive volume of raw data must be buffered at the experimental facility, then shipped to a hierarchy of centers for data analysis. A number of distributed regional computing centers in the U.S, Europe and Asia have intermediate storage and processing facility. These feed large tier-two centers, each of which feeds data to a sizable number of institutes for local processing. From here, the data can be sent to workstations and compute farms of individual researchers. Many different teams may analyze the data sets of the same event. The results of the initial analyses may also generate large derived data that is transported, and ultimately shared, among central locations for statistical and other analyses. In addition to this experimental raw run data, there are large volumes of derived simulation data, estimated to run a factor of three times greater in volume than the former sets. Simulations are crucial for understanding the nature of the Large Hadron Collider (LHC) detectors and modeling the underlying physics, to enable a comparison between what is seen and what is expected to be seen. Experiments search for unknown facts derived from only a few events of interest a year. Original data must be reanalyzed a number of times as the analyzing algorithms improve and new understanding of the emerging physics materializes.

This implies the need to move and manage enormous quantities of data, as well as to manage and coordinate storage and computational resources. Since each LHC detector event is independent of all other events, the analyses can be performed in a highly parallel and distributed fashion, which fits perfectly with the emerging concept and reality of Grid computing. Grids provides a way of marshalling and sharing resources for distributed processing, using "divide-and-conquer" to tackle computational problems that are too large to be dealt with by one single computer or even an institution alone. Grids enable coordinating the use of distributed and heterogeneous resources across multiple organizations by providing a uniform set of middleware in terms of web services to allow authentication, authorization, accounting, resource discovery and scheduling, remote job execution, and feedback to the applications. Table 2.3 represents size estimation of annual data generation of e-Science disciplines [2].

Discipline	**Annual)ata Generation**
Climate studies	>10 PB/year
Bioinformatics(Genomics, proteomics, netabolomics)	>20 PB/year
Astrophysics	8 PB/Year
Sloan Digital Sky Survey	15 TB
Chemistry	4 PB/year
Materials Science (neutrons and photons)	0.35 'B/year

Table 2.3 : Annual Scientific Data generation by 2008 .

Heretofore, the network has been a bottleneck in this scenario. Using a "fast" 100 Mbps network, it would take almost a day to move a 1 TB data set, or 2 1/2 years to move a Petabyte, assuming that the full bandwidth were available for this period. However, the packet routed IP network is shared, has its own overhead, and cannot provide QoS.

2.4 Service Oriented Architecture (SOA)

SOA has been proposed to build cooperation between data and process components within different systems. It supports reusability and interoperability of components using Internet Web infrastructure. Initial Grid Computing models started to embody an SOA that lead to the definition and creation of OGSA. Further work realizes the benefits of the reusing features of Web Services and hence Web Services Resource Framework (WSRF) was born as an amalgamation of OGSA and Web Services.

This leads to an increase in the efficiency of assembly and a decrease in the cost of development. A typical SOA example is WS (Web Services), which provides a set of standards and protocols including the platform independent Web Service Definition Language (WSDL). In recent years, the need for tools for accessing scientific data and executing complex computations and analyses on the data has increased in a variety of disciplines. Such analyses can be modeled as scientific workflows in which one step is connected to another by description in a formal workflow language. The above components of Grid Computing are discussed in more detail below.

2.5 Grid Computing and its Infrastructure

Web Services - Web services use simple XML-based messages for machine-machine messaging. Web services do not necessarily involve web browsers; rather, we can view web services as XML-based APIs. They use standard internet technologies to interact dynamically with one another with a well-understood security model. Web Services are loosely coupled, can be combined to form complex services, and open agreed standards connect disparate platforms. Middleware based on web services has enjoyed tremendous success in the past five years. Examples are eBay, PayPal, Amazon and Google, all big users of web services.

Built upon the Internet and the World Wide Web, Grids or the Grid network is a new overlay network infrastructure. By providing scalable, secure, high-performance mechanisms for discovering and negotiating access to remote resources, Grids promise to make possible scientific collaborations that share resources on an unprecedented scale. Geographically distributed groups can potentially work together in ways that were previously impossible. (Ref: The Grid: A New Infrastructure for 21st Century Science by Ian Foster).

The concept of sharing distributed resources is not new; however, a combination of technology trends and research advances make it feasible to realize the Grid vision: to put in place a new international scientific infrastructure with tools that, together, can meet the challenging demands of 21st-century science. (Ref: The Grid: A New Infrastructure for 21st Century Science by Ian Foster).

A Grid network can be thought of as a distributed computing infrastructure that supports the creation and operation of Virtual Organizations (VO) by providing mechanisms for cross-organizational resource controlling and sharing.

According to Foster [28], a Grid system must possess three properties:

- It coordinates resources that are not subject to centralized control,
- it must use standard, open, general-purpose protocols and interfaces, and
- it must deliver nontrivial qualities of services.

2.6 Middleware and Grid Orchestration

In Grid architecture, a Resource Management Architecture for Meta-computing Systems [42] is proposed to deal with the co-allocation problem where applications have resource requirements that can be satisfied only by using resources simultaneously at several sites. Usually that allocation of computation is co-allocated with the allocation of storage. There is no need to allocate the computation if the storage is not available. However, this architecture, does not address the issue of advance reservations of the network, and more specific the heterogeneous resource types that are necessary for realizing end-to-end quality of service (QoS) guaranteed in emerging network-based applications [23]. To address this problem, the Grid Architecture for Reservation and Allocation (GARA) was proposed [43] by GGF. By splitting reservation from allocation, GARA enables advance reservation of resources, which can be critical to application success if a required resource is in high demand. One limiting characteristics to GARA is its assumption that the network is constantly available. Our architecture goes a step further by addressing one of the most challenging issues in the management of resources in Grid environments: the scheduling of dynamic and stateful Grid services where negotiation may be required to adapt application requirements to resource availability, particularly when requirements and resource characteristics change during execution. Recently, a WS-Agreement negotiation model was proposed [44] by GGF that uses agreement negotiation to capture the notion of dynamically adjusting policies that affect the service environment without necessarily exposing the details necessary to enact or enforce the policies.

There are a number of prior works that address monitoring, discovery scheduling and allocation for Grids. They include such projects as GARA, GRAM, DUROC, MDS, and Condor [45]. Globus Toolkit 3 (GT3) [40] brokering architecture, GRAM, includes computation scalability but lacks network scalability.

GARA - Globus Architecture for Reservation and Allocation – GARA [43] was first proposed by Ian Foster and Carl Kesselman in the Grid 1999 . The concept differs from existing Grid schedulers like Condor with advanced reservation, multiple resource types, co-reservation, and support for multiple administrative domains. GARA was demonstrated with CPU and network schedulers using DiffServ for network QoS. GARA included a Java API for advanced reservations on a single resource, and co-reservation was implemented as separate "co-reservation agents" that interfaced to GARA and had their own API. Work seems to have tapered off or stopped on GARA; it was not included in Globus Toolkit 3 (GT3) or the newly released GT4.

GARA had significant limitations. Its brokering was based on the scheduling of computation and storage resources using sequencing, without considering the network dimension. This solution did not give the network equal value to computation and storage. While

GARA’s task queuing mechanism was appropriate at the time for computation and storage, it failed to meet communication needs; integrated architecture was needed to better serve applications with large volume data transfer demands. For example, GARA was proposed to match computing jobs to Grid nodes according to CPU resource requirements and availability; however, it did not extend naturally to allocating network resources.

GRAM - Globus Resource Allocation Manager - GRAM is the job submitter and job manager for Globus Toolkit, first implemented in GT2. GRAM provides an interface for clients to request resources. In GT2, clients code requests in Resource Specification Language (RSL) that had a keyword equal to a value grammar. In GT3, clients can also use RSL2, which uses XML to accomplish the same thing. In GT3, GRAM is a collection of OGSI [46] services: MJS, MJFS, and MMJFS. GRAM uses an associated service called Resource Information Provider Service (RIPS) to get information about host resources and job status. RIPS reports but does not allocate, and is not intended for generalization to other resources. In GT 4.0 the main interface switched from OGSI to WSRF [33].

Limitations for GRAM are related to its inability to perform network advanced reservation or scheduling. Neither RSL nor RSL2 includes the syntax to specify network resources. In GT-4.0, job submission moved to an XML based interface to allow simplicity and better accessibility to the data.

GASS - Globus Access to Secondary Storage - Globus Toolkit provides services for file and executable staging and I/O redirection that work well with GRAM. GASS uses GSI-enabled HTTP as the protocol for data transfer, and a caching algorithm for copying data when necessary. The GASS APIs provide programmer access to staging capabilities, which are integrated with the GRAM job submission tools.

DUROC - Dynamically Updated Request Online Co-allocator – DUROC allocates computation resources using multiple GRAM instances. It uses a slightly extended RSL as input and communicates with GRAM using RSL. Intended for jobs distributed across multiple systems, it requires code in each job fragment to "sync up." Two major limitations of DUROC are that it is not included in the standard GT package and it does not support network allocation. Therefore, it is an unsatisfactory solution for the applications that are targeted in this thesis.

MDS - Monitoring and Discovery Service – MDS aggregated "service data" from OGSA services and made the data available for query. This OGSI is in transition to WSRF. MDS started in the early days of GT2 and consisted of GRIS instances, which reported resources from both single sources and an index service (GIIS), and aggregated them into LDAP indexes. Data could be presented as a single data item or an XML document. Intent included resource location, but GRAM did not use it; the GRAM client was expected to provide resource location. The MDS concept worked for job allocations on computation resources, even for a large number of jobs and resources, with a concise enough description to be published. However, this approach is too complicated to manage network resources even on a small topology. It is not scalable, making it nearly impossible to advertise all possible host-to-host paths in even a medium-sized network.

Condor – Condor functions as a remote job submitter [45], primarily in Grid High Energy Physics research where vast amounts of computation allocation is necessary for research. Prior to computation large amounts of data must be available for allocated computation. Condor can stage files locally or makes them available remotely. Considered a fault-tolerant middleware, Condor monitors running jobs and restarts them when necessary. It can relocate a job from one computer to another, can submit to GRAM, but is not part of Globus. It utilizes a ClassAd system to advertise resources and match jobs to resources. Simply stated, it is a way to match jobs to the computers that will run them. ClassAd is symmetric, in that resource descriptions are the same whether they come from the supplier or the consumer. It attempts to avoid the monolithic model of centralized, scheduled resource management, and moves to a decentralized system with opportunistic scheduling. Matchmaker identifies possible matches but leaves it to the parties involved to negotiate and allocate by categorizing them as either "Matching" or "Claiming."

Enlightened – The Enlightened project is a grid framework that allows application to dynamically request computing, storage and network resources. The framework provides a set of tools and protocols to enable fast network reconfiguration and on-demand or in-advanced provisioning. The project views the network as a grid resource similar to compute and storage resources and abstracts the networks resources as distributed network intelligence among network control plane, management plane, and grid middleware.

This section described several grid middleware software packages in the area of resource managements, allocation and co-allocation of resources with specific focus on two data-intensive projects.

2.7 Current Efforts at Globus Grid Forum (GGF) Working Groups:

This section describes some of the current efforts by GGF working groups in the areas of distributed resource management, agreement protocol, definition language, and high performance networks.

- **DRMAA - Distributed Resource Management Application API** – DRMAA is a multi-host job submission API, with recommendation includes no network parameters, just an

API with no scheduler component. The target is to unify "legacy" environments, making apps portable.

- **GRAAP - Grid Resource Allocation Agreement Protocol** – GRAAP is an agreement-based peer-to-peer negotiation protocol, based on OGSA services, supporting any kind of agreement. It is extensible, so can be extended to network resources. GRAAP appears to be a complex, heavyweight interface, treating each agreement as a new OGSI service instance, and it was not implemented or supported.

- **JSDL - Job Submission Definition Language** – The JSDL working group at GGF is attempting to standardize the description of a job so that any scheduler can accept the same description. JSDL may take the place of RSL in Globus. This GGF working group is explicitly avoiding the entire resource description issue and trying to build it from scratch. JDSL is based on XML syntax.

- **GHPN – Grid High Performance Network** – As part of the Global Grid Forum (GGF), the GHPN Research Group focuses on the relationship between network research and Grid application and infrastructure development. The objective of GHPN-RG is to bridge the gap between the networking and grid research communities.

All of the above mechanisms emphasize functionality from the application middleware, whereas this thesis emphasizes functionality from the network middleware. Computation, data and storage middleware have substantially different characteristics from network middleware, such that full integration with network resources is prohibited. Being based on the network application, allows greater ability to manipulate network functionality. Most of the above assume constant network availability, using shared networks, packet switching, and QoS mechanisms. By contrast, our approach uses a dedicated unshared network, virtual circuit switching, with no need for QoS mechanisms. By working from the application middleware, it was necessary to install agents on computers at the edge of the network, while we were able to install the agents inside the network directly attached to the network devices along the path.

Characteristics	**Application Focused**	**Network Focused**
Agents ocation	Edge of the network	Inside the network
Agent installed	Computers	Network devices
Manage and llocate	Computers, clusters, storage, data, B, visualization devices, unique sensors	Routers, switches, ptical devices, links
Topology	End-to-end	Segment oriented
Representation	Point(s) (physical resources)	Point-to-point
Distribution	Resource on a specific location	Multiple routs etween locations
View	The network is a cloud	The network is a omplex system

Table 2.4: Distinguishing features between application and network middleware

2.8 View of Overall Architecture for e-Science

Cyber-Infrastructure for e-Science consists of many elements that can be represented in a very high view in Diagram 2.2. The work on this thesis and the specific contribution is represented in the yellow rectangle. As part of the Grid Layered Architecture proposed by Foster and Kesselman [28], we position the Lambda Data Grid as part of the Collaborative and the Resource layers. Lambda Data Grid can be viewed as part of the framework that supports e-Science.

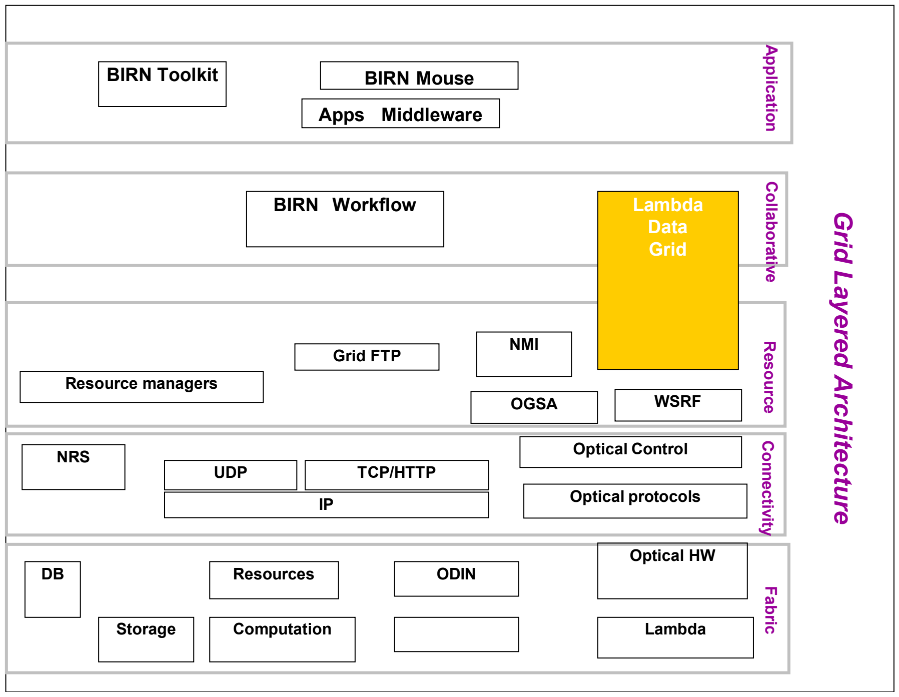


Figure 2.2 – Lambda Data Grid as part of Cyber-Infrastructure Layered Architecture

2.9 Optical Networks Testbeds

The majority of initial research and development efforts related to Grid networking focused on integrated Grid services with enhanced L3-L4 protocols [47]. Many early efforts were focused on managing QoS traffic characteristics, for example, using layer-3 approaches to obtain QoS in packet switching networks [48] [49] [50] [51]. One of such Globus projects is GARA. Previously stated limitations to GARA prohibited it from maturing. Some efforts focused on integrating Grid services with enhanced L3-L4 protocols [51] [47] [52].

More recently, it has been recognized that some data-intensive Grid computing requires the handling of extremely large data sets, shared and distributed over multiple sites with such demanding high performance and service level guarantees that their requirements cannot be met by packet switched networks. This recognition has been presented in research workshop reports, in Grid standards development efforts (a draft describing issues related to optical network architecture for Grid services has been submitted to the GGF) [53, 54], and in special issues of scholarly publications.

Today, a number of research initiatives are focusing on integrating Grid services with emerging optical network capabilities [54]. Optical networks based on wavelength switching can be considered circuit switched and are able to provide high-bandwidth and layer-2 QoS easily, providing an additional resource option for data-intensive Grid computing. Furthermore, a number of optical testbeds have been established to support the research and development of these new architectures and platforms. They include OMNInet [55], the OptIPuter, a distributed Cyber-infrastructure designed to support data-intensive scientific research and collaboration [34], I-WIRE, [56], and DRAGON [57]. Additional experimental testbeds supported by CA*net4 [58] have introduced an innovative method for "User Controlled Lightpaths (UCLP) [30] and the Ultra Science Network [1]. These new concepts are being demonstrated [59] [60]at national an international conferences, at iGRID2005, SC-2005, and GGF.

It is important to note that these initiatives are directed at creating and implementing architecture for networks based on dynamic wavelength utilization controlled at the network edge, not within a centralized environment. In part, this concept of edge access represents a fundamental migration away from the legacy idea of managed network service within a heterogeneous, centrally managed network to one that allows for highly distributed access to core network resources. These concepts are beginning to appear in national and international infrastructure, including in the TransLight [35], an innovative international network, the Global Lambda Integrated Facility (GLIF) [61], Ca*net4 [30], StarLight [62], NetherLight [63], UKLight [64], and others. It is expected that this architecture will also become part of large-scale distributed computational infrastructure such as the TeraGrid [65].

TeraGrid [65] connects a grid network of super-computers distributed in four remote locations from the Midwest to the West coast, and exchanges data at 40Gbps transport rate (4xOC-192 10Gbps Lambdas). However, these are static Lambda connections while DWDM-RAM provides a dynamic setting of Lambdas [31]. TeraGrid requires L3 routing while DWDM-RAM provides dedicated optical path(s) to the destination [32]. Therefore, the data transfer in Globus is done mainly in an L3 environment, and specifically in GridFTP [66]. These L3 routing limitations require specific optimization to the data transfer. In comparison, DWDM-RAM provides lightpath in the optical domain and not in L3 and layers above.

The **OptIPuter** [34] research program designed a new type of infrastructure based on a concept of enabling applications to dynamically create distributed virtual computers [67]. This architecture will provide a close integration of various resources, high performance computational processors, mass storage, visualization resources, and dynamically allocated distributed backplanes based on optical networks using advanced wavelength switching technologies. The first prototype is currently being implemented between StarLight [62] and NetherLight [63]. Other, more extensive, implementations should be more widely available in approximately 2008. In contrast, DWDM-RAM [68] has a narrower scope, high performance data services over dynamic Lightpath within metro areas, and a shorter timeline toward more general implementations.

GLIF [61] - Global Lambda Integrated Facility (GLIF) is an international virtual organization that promotes the paradigm of Lambda networking. GLIF participants include National Research and Education Networks (NRENs) and other institutions working with

Lambdas. The GLIF community shares a common vision of building a new grid-computing paradigm in which the central architectural element is optical networks, not computers. GLIF's links are made available for manual reservation and scheduling for scientific experiments.

In this thesis, we built an innovative technology for network service orchestration. It allows automated interaction among grid middleware, the scientific domain middleware and scientific workflows. It utilizes network resources as vital elements in the scientific process.

2.10 Summary of Related Work

In this chapter, we discuss the background and related work as they relate to this thesis. Vital elements include key distributed computing technologies such as Service Oriented Architecture (SOA), Open Grid Service Architecture (OGSA), Web Services Resource Framework (WSRF), Global Grid Forum (GGF) Grid High Performance Networks (GHPN), and intrinsic resource-scheduling mechanisms such as GARA [43], GRAM, DURAC, Condor [45], and GRIS. Next, we create a picture of what exists for e-Science architecture and reference my work within it. Additional topics discussed include some research networks such as OptIPuter, TeraGrid, UCLP, and BIRN.

What follows is a discussion of limitations inherent in bulk data transfer via current networks and the details of our solution, including how we accomplished the construction of a Grid Virtual Organization.

3 Bulk Data Transfer and Optical Networks

This chapter analyzes the fundamental features of optical networking and discusses its relevance to OGSA service architecture. It reviews earlier efforts in developing schedulers that handle Grid applications and points out the need for a network resources scheduler. Finally, this chapter discusses a number of end-to-end Grid transport protocols. This chapter provides analysis of technology directions and limitations that are necessary to understand the solutions provided by our Lambda Data Grid.

3.1 Introduction

Rapid development of agile optical networking transport technology has provided huge bandwidth capacity to e-Science applications. Presently, a single strand of optical fiber can provide hundreds of distinct 10 or 40 Gbps data channels (Lambdas), with a capacity of over 6Tbps, which is approximately the amount of traffic circulating in the Internet backbone. This innovation in optical networking technology has changed the communication paradigm from one of a narrow bottleneck to that of a fire hose.

An optical connection between two or more nodes provides data transport capability, which can be used at one time by a single user, or shared by multiple users. Time-based resource utilization and sharing requires network scheduling. A scheduled, dedicated, circuit switched connection provides a guaranteed Quality of Service and avoids many of the issues of packet switched connections, such as collisions and recovery, so that the full bandwidth is available to a single user at a time.

Without enhancements, data transport over the conventional TCP results in about 20Mbs over the cross-Atlantic optical network. New transport protocols have been proposed to improve TCP in order to cope with extreme high-speed transport by data intensive applications [51, 52, 66].

3.2 Bulk Data Networks

3.2.1 Outdated Assumptions for e-Science Applications

The Internet has evolved over the last thirty years. Internet Design Principles [51] have been based on the latest technology at any given time. New technology advances in silicon, computation, bandwidth, optical, and storage, occurred after these principles were designed. Accordingly, the cost structure has changed dramatically at many levels. As we look at the assumptions driving the original Internet Design Principles, we see that the assumptions are not applicable to our targeted e-Science applications.

3.2.2 Size Limitations

Current Internet architecture can handle applications like Web, Telnet, and email. Yet, it has fundamental limitations with new Data Intensive Applications that require access to multi-TeraBytes or PetaBytes of data. The task of moving 10KB compared to10TB, nine orders of magnitude greater, presents vastly different requirements. The requirements by High Energy Physics (HEP), for example, cannot be addressed by the current Internet architecture, and as such, necessitates changes in architecture and design.

3.2.3 Bandwidth

The handling of a transmission of 1Mbs compared to one of multi-10Gbs is different. This is four orders of magnitude greater. When dealing with small pipes in the range of 1-100 Mbs, we can do some optimizations in the protocol stack. However, when dealing with large pipes in the range of multi-10Gbs, protocol stack and application optimizations may hinder processing. Radical changes in bandwidth requirements call for innovative solutions that will distinguish between small flows and big flows.

3.2.4 WAN: Neither Expensive, nor a Bottleneck

Historically, WAN traffic has been the bottleneck of the network and has been relatively expensive. The introduction of optical transmission counteracts these constraints. Analyzing the total cost of electrical forwarding compared to optical transmission, per-bit transmission in L3 is several times more expensive than in L0. Router ports are more costly than optical ports and have been the source of some bottlenecks. In other words, the transmission of photons compared to the transmission of electrons is relatively inexpensive.

3.2.5 Optical Transmission Faster than Disk Transfer Rate

Advanced and efficient storage architectures have been built for fast data access of large data sets. These deployments are unique and very expensive. In the past, disk transfer rate was assumed to be faster than WAN. Recently, optical transmission has become much faster than the internal computer bus or cluster interconnects. The timing assumptions for seek-time and transfer-time of small buffers are different from very large data sets. Copying data to local storage may be less efficient than working on remote memory over a dedicated, small delay, large optical link. As such, these advancements have paved the way for Remote Direct Memory Access (RDMA) instead of copying to a local disk. This provides the opportunity to move the data to the computation, to move the computation to the data, or to allow remote access to the data.

3.2.6 Changing the Nature of Technology

Decreases in computation and storage prices have prompted a change in architecture considerations. Low price points are driving enormous use and phenomenal growth. New technologies are available that include disk, storage, memory, computation, communication, and last mile access. These technologies require a change in design assumptions and considerations for architecture capabilities.

3.3 Limitations of Packet Switching for DIA

3.3.1 Inefficiency in Forwarding Decisions

Internet architecture cannot realistically move tens of Terabytes or Petabytes. Packet switching is a proven efficient technology for transporting burst transmission of short data packets, e.g., for remote login, consumer oriented email, and web applications. It has not been sufficiently adaptable to meet the challenge of large-scale data. Making forwarding decisions on Ethernet frames every 1500 Bytes is sufficient for emails or 10k -100k web pages. This is not the optimal mechanism if we are to cope with data sizes six to nine orders of magnitude greater. For example, in the transmission of 1.5TB files via packet switching, the exact same forwarding decision may be made one billion times, resulting in an extremely ineffective process.

3.3.2 Packet Size and Switching Time

Setting circuit switching over optical links is a more effective multiplexing technique. Packets of 1500 bytes, or a minimum of 64 bytes, are adequate for slow traffic like T-1, 10Mbs and 100Mbs. However, high bandwidth optical links can transmit three to six orders of magnitude more bandwidth.

	1500 Byte	**64 Byte**
1Mbs	10ms	500μs

10Mbs	1ms	50 µs
100Mbs	100 µs	5 µs
1Gbs	10 µs	500ns
10Gbs	1 µs	50ns
40Gbs	250ns	12ns
100Gbs	100ns	5ns

Table 3.1: Switching time.

While silicon can easily handle the switching time of slow bandwidth, limited time budgets present a problem for silicon, as Table 3.1 illustrates. When looking at optical advances compared to silicon advances, it is difficult for silicon to keep up with the required switching time for high-bandwidth by optical transmission systems.

Tim	**1 Mbs**	**100 Mbs**	**1G s**	**10Gb**
10 econd	1 MB	100M	1G	10GB
1 Minute	6 MB	600M	6G	60GB
1 lour	3 0MB	36GB	360 GB	3.6TB
1 Day	8. 4GB	864G	8.64 TB	86.4TB
60 ays	1 TB	100T	1PT	10PTB

Table 3.2: Time for data transfer.

Table 3.2 illustrates bandwidth and storage in relation to how long it takes to transfer this data. Different applications have different time requirements. In life science, experiments are scheduled 60 days in advance to accommodate the transfer of data, while in environmental science, and specifically weather prediction, data received days later is unusable. NASA's scheduling of space launches cannot tolerate the delay of data transmission. Further, accessing remote super computing centers has proven to be a powerful tool to affect better and more accurate weather prediction. The amount of time it takes to transmit data is an important characteristic of the application type. The Internet bandwidth limitations prohibit these types of applications from transmitting data in the required time.

3.3.3 L3 Limitations

While Internet architecture has served its purpose for the past 30 years, attempting to run Data Intensive Applications might push it beyond its limits.

3.3.4 Not targeted for Large Data Sets

The existing L3 architecture is not targeted to move multi-Terabytes of data over multi-10GE Lambdas, over large round-trip-delay networks. Slow Start, Congestion Control, and Congestion Avoidance mechanisms work for multiple small streams, but are not optimized for dedicated, point-to-point, long duration, large bandwidth pipes. New improved L4 mechanisms allow higher bandwidth transmission. These greedy mechanisms do not perform well on public, shared, congested L3 links. They function best on dedicated optical links [53].

3.3.5 Forwarding decisions

During the last few years, point-to-point transmission has been mainly through optical links. Bit-Error-Rate in L3 routing is greater by several orders of magnitude than in L0-L1 optical technologies [59]. Thus, there is an increased probability of information loss in Data Intensive Applications that will force retransmissions.

3.4 Optical Grid networks

With recent advances in Wavelength Division Multiplexing (WDM), and Ultra Long Haul (ULH), transmitting data over thousands of kilometers of optical networks can be achieved without signal regeneration. This growth of two orders of magnitude in optical transmission distance means four orders of magnitudes less regeneration points in a given geographic area. When comparing this L1 transmission to L3 routing and next hop decision points, forwarding decision points are significantly reduced.

This is a fundamental change in data transport. With transmission over a very long distance, there is no need for every IP router to make the decision about the next hop. Instead, the optical transmission can be done in one, or a few, hops. The fundamental difference is that edge devices will make destination decisions on the edge of the cloud, instead of the L3 core routers making next hop decisions in the core of the cloud. For many short-range data transports, routing works well. However, for a long-range or dedicated data transport with a few edge decisions, optical transport is more effective.

Recent advances in bandwidth availability of optical networking technologies and rapid optical control have created a radical mismatch between the optical transmission world and the electrical forwarding/routing world. End-systems, which run data-intensive applications, do not have access to the abundant bandwidth of optical networks. Furthermore, even though disk costs are attractively inexpensive, the feasibility of transmitting huge volumes of data is limited. The encumbrance lies in the limited transmission ability of the IP architecture. Currently, the Internet architecture cannot effectively transmit PetaBytes or hundreds of TeraBytes, and has impeding limitations in providing fast transport service to high-end data-intensive applications.

In theory, the problem is no longer the inability of the network to move massive quantities of data in a timely manner, but rather how to take advantage of this capacity and "fill the pipe."

In most practices, optical connections have been statically provisioned and dedicated to a few users. For a dedicated user, the lead-time for getting such a connection is often several months, the cost quite high, and the efficiency of use low. On the other hand, a shared switchable dynamic optical network can be instantaneously available to a user who would pay for, and get, only the capacity needed for the time needed.

In the DWDM-RAM [32] project, we designed novice architecture to enable Grids to take advantages of a dynamic, schedulable high capacity optical network, in support of data intensive applications. We built a prototype system that demonstrated some key features and capabilities of the DWDM-RAM project. The architecture, described in more details in several papers published in conferences and journals [3, 4, 5], provides two layers between the users and their applications and the underlying dynamical optical networks.

3.5 E2E Transport Protocol for Data Intensive Applications

In the case of narrow bandwidth, network delay is not critical; whereas, in large bandwidth pipes with significant delay, the network cannot function effectively.

Responsiveness - TCP works well in small Round Trip Time (RTT) and small pipes. It was designed and optimized for LAN or narrow WAN. TCP limitations in big pipes and large RTT are well-documented [47] [66]. The responsiveness is the time it takes to recover from a single loss. It measures how quickly it goes back to using a network link at full capacity after experiencing a loss. Packet dropping is an imperative mechanism for fairness in packet switched networks. Packets loss is detected by the end-systems as a signal to slow down. This mechanism is embedded in the TCP congestion control and was designed for multi-streams sharing the same networking infrastructure. It measures how quickly it goes back to using a network link at full capacity after experiencing a loss. For example, 15 years ago, in a LAN environment with RTT = 2ms and 10Mbs, the responsiveness was about 1.7ms. In today's 1Gbs LAN with RTT of at most 2ms, the responsiveness is about 96ms. In a WAN environment, the RTT is very large, e.g., the RTT from CERN to Chicago is 120ms, to Sunnyvale is 180ms, and to Tokyo 300ms. In these cases, the responsiveness is over an hour [66]. In other words, a single loss between CERN and

Chicago on a 1Gbs link would take the network about an hour to recover. Between CERN and Tokyo on a 10GE link, it would take the network about three hours to recover [66]. OptIPuter experiments [34] using dedicated channel of 1Bbs between Chicago and Amsterdam show bandwidth of 4.36Mbs, using un-modified TCP protocol. New UDP based protocols show bandwidth of 700Mbs-920Mbs. Dedicated pipe utilization in un-modified TCP is less than 1% compared up to 92% utilization in new UDP based transport protocols. In dedicated optical links, there is no sharing, and fairness is consequently not an issue. There is no competition for network resources, and fairness can be achieved with advanced reservation, scheduling, and allocation of networking resources. For this reason, responsiveness is not a problem.

New Transport Protocols - Many new protocols were designed to address network limitations, among them are GridFTP [41] , FAST [69], XCP , Parallel TCP, and Tsunami, SABUL/UDT [47]. These research projects, along many other similar projects provided enhancements for the basic transport mechanisms by providing effective utilization of large bandwidth pipes. The enhancements in these protocols are achieved via three mechanisms: 1) tuning the TCP and UDP knobs; 2) transmitting over many streams; and 3) sending the data over UDP while the control is done in TCP.

When using long-range 1Gbs connection, findings showed that GridFTP achieved 512Mbs [66], and Tsunami achieved 700Mbs [66] . SABUL achieved 910Mbs [66], and FAST achieved 930Mbs from CERN to SLAC. New experiments show multiplexing of 1Gbs of FAST and SABUL [47] into 10Gbs Lambdas achieves better link utilization.

Fairness - In packet switching, Congestion Control and Congestion Avoidance mechanisms provide some level of fairness among different sessions on the same pipe.

Conversely, this work contains is a dedicated link; thus, fairness is not an issue. There is no competition for network resources on the same pipe.

S**hared vs. Dedicated Links** – Many of these new protocols are assumed greedy and present fairness challenges to competing streams in a shared media. In designing a protocol on a dedicated link, the fairness is not important because the optical link allocation is from the entire media capacity without competing traffic on the same link. In dedicated links (Lightpath, circuit) there is no sharing and the link is granted for a period of time.

3.6 Recent Developments

The OGSA architecture is much richer than many of its predecessors and holds much promise, but in its current state it does not handle networks well either. The time seems ripe for OGSA extensions that would make it easy to fit network allocations into the Grid environment. As a result, it is necessary to build this newfound capacity into the Internet architecture that evolves. The challenges that arise are two-fold: 1) user access to bandwidth, and 2) effective operator management of the resources.

At present, the applications do not have access to this abundant bandwidth because the current Internet architecture does not accommodate bandwidth requirements.

3.6.1 Dynamic Optical Control

Recent advancements in Agile Optical Control make reality the dynamic configuration of optical links. These advancements separate the optical control plane from the optical transmission plane. The optical transmission plane is responsible for handling enormous amounts of data, while the control plane is responsible for decisions concerning data paths to the final destination. Among these advancements are Advanced Switched Optical Networks (ASON)

[58], Advanced Switched Transport Networks (ASTN), and Generalized Multi-protocol Label Switching (GMPLS). These are suites of protocols that perform signaling, resource discovery, and link management. One of the goals of these optical control schemes is to determine the optical path between source and destination, to allow a higher level of network management or operator to select and set optical links along the path, and to configure the optical switches.

We use dynamic optical control as a mechanism to manage the movement of large amounts of data between source and destination, and to build the interface from the middleware. For example, an application that must move multi-TeraBytes or PetaBytes of data across distance can ask the network to provide the lightpath between the two ends. The network will create this lightpath between these end-points. Thus, the reservation of a connection-oriented link simplifies the routing overhead, and provides better service and higher bandwidth capacity. In many cases, the link is not on the public Internet; hence, it provides better security. Dynamic optical control mechanisms like GMPLS have started to evolve. This provides a connection oriented mechanism (circuit) of very large pipes. Examples that can use these types of connections can be found in Grid Computing, SAN, and large volume replications.

In many cases, protocol stacks are a significant overhead for the efficiency of the network. Experiments performed at the OptIPuter project [32] between San Diego and Chicago, demonstrated the ability to transmit 1Gbs non-stop, over a dedicated optical link, with almost zero packet loss, over several days. These results bring into question the efficacy of complex protocol stacks.

3.7 Change in Cost Structure

Recent cost structure changes have generated new economic considerations that drive fundamentally different architecture principles.

- **Inexpensive Optical Bandwidth** - The appearance of DWDM has provided many Lambdas, each one accommodating high bandwidth over a long distance. Thus, transmission costs per data unit are extremely low, encouraging the research community to consider transmission of bandwidth to be almost cost free. This is in direct opposition to the "bandwidth is expensive" assumption prevalent for the past 20 years. That bandwidth is almost free makes it necessary to rethink the research challenges and the design considerations, and look beyond our older assumptions.
- **Optical Costs** - IP routing architecture in OC-192 costs nearly ten times more than the optical transmission equivalent. Specifically, an OC-192 router port costs nearly five times as much as the Optical Cross Connect (OXC) equivalent. Furthermore, at intermediate nodes, the router ports are additional to optical costs.
- **Connectivity Costs** - Until recently, it cost about $1 million for a OC-192 connection, coast-to-coast. The design of the new optical ultra-long-haul connection reduces the economic restraints of big-pipe, long-haul connections.
- **Last Mile Costs** - Previously, the last-mile connections were expensive and very narrow. Due to recent economic restructuring, the Optical Metro service has changed the principles of the access. Therefore, big-pipe last mile connections are affordable to enterprises.
- **Broadband Costs** - Now, like never before, broadband access is affordable. Millions of people are taking advantage of this technology, resulting in a substantial increase in data transfer.

- **Free LAN Bandwidth** - At a new price point of $49., Gigabit Ethernet (GE) NICs have become extremely cost effective. GE has become a commodity for servers and desktops. This will drive a domino effect into 10GE. With this price point per bit in the LAN, bandwidth is almost free in these areas.

- **Storage Costs** – Today, 1 TB costs about $500. This affordability has encouraged the use of large amounts of data. In particular, a 1-PetaByte storage system costs approximately two to three million dollars, which is within the budget of large organizations. With this new economic cost structure and affordability, more organizations will build large data storage.

- **Computation Costs** - The computation power on our desks is larger than a super computer of 20 years ago, and at an unfathomably low price point comparatively. Moreover, the powerful supper computers of today are financially feasible for large organizations. This phenomenon drives massive amounts of computation at low prices.

3.7.1 Summary

This chapter analyzes the fundamental features of optical networking and discusses its relevance to OGSA service architecture [46] [36]. It reviews earlier efforts in developing schedulers that handle Grid applications and points out the need for a network resources scheduler. Further, it discusses a number of new end-to-end Grid transport protocols, dynamic optical control, and the change in cost structure of bandwidth as fundamental elements in building networks as part of scientific Cyber-infrastructure.

What follows is a bird's eye view of Lambda Data Grid as part of Cyber-infrastructure for building networks for e-Science. Establishing the network as the central entity presents some challenges and are discussed in this chapter. To further illustrate the concept, we will present an e-Science scenario as it relates to the GLIF globe-wide topology, along with my cut-through solution for data management.

4 Lambda Data Grid - Building Networks for e-Science

This chapter presents a bird's eye view of Lambda Data Grid as part of Cyber-infrastructure for building network for e-Science. Scientific research is discussed demonstrating how Lambda Data Grid provides solutions for collaborative research over a geographically dispersed area using Grid technologies to build a Virtual Organization (VO) [70]. Further, we explain how Lambda Data Grid middleware can dynamically build an optical cut-through over GLIF. This high-level view provides the framework for Lambda Data Grid architecture presented in Chapter 5.

4.1 Introduction

The innovative concepts of this thesis are inspired by a number of key challenges in Grid computing and optical networking. An evident challenge exists to match the requirements of e-Science applications that generate vast amounts of data, with the potential of optical transport that has the ability to transmit the data effectively across the globe. There are unsatisfied requirements for scientific research programs to achieve productive results. As an example, GLIF is a global optical infrastructure that allows manual cut-through of dedicated connectivity between scientific institutes to establish multiple isolated network connectivity graphs. The notion of this "few-to-few" connectivity reflects a significant difference from the notion of current Internet that functions as "many-to-many." The operation of establishing connectivity is currently performed manually with emails, phone calls, reminders, and physical reservations. These inefficient methods inspired me to build an architecture that automatically orchestrates network resources in conjunction with other resources, as an integral part of the Grid middleware and the scientific workflow.

Virtual Organization (VO), or Grids, pioneered a new era of computation, but the connectivity between distributed VOs is a vital issue, which has not been successfully resolved. One of the contributions of this thesis is to complement the VO concept with necessary network solutions.

4.2 Examining the Gaps in Collaborative Research

A characteristic of new e-Science applications is that they generate vast amounts of raw data, intermediate results from computations, and those data need to be relayed between geographically dispersed scientific organizations. This indicates a significant conceptual shift from local super-computer centers to a globe-wide computing network, where optical networks function as a backplane of a truly global distributed computing system. The amount of available bandwidth facilitated by the DWDM technology can support the transmission of the vast amount of data generated. However, how to make the solution scalable and cost effective results in a number of issues reflecting some the research gaps, which are addressed in this thesis.

4.2.1 Data size

Large-scale scientific experiments generate huge amounts of data. Many super-computer centers have the storage and computation capabilities to process these immense amounts of data, but do not have the capability to transmit the data effectively.

4.2.2 Super-networking Transforming Super-computing

Bringing intelligence to the optical network control system changes the nature of the optical transport network. It is no longer a simple data transport system, but rather, a integral part of a large-scale data distributed system. Figure 4.1 presents the optical network as a backplane for a globe-wide distributed computation system. In the past, computer processors were the fastest components while the peripherals were the bottlenecks. Now the reverse is true. The network powered by the optical transport is faster than processors, and other components such as storage, software and instrumentation, are becoming the slower "peripherals."

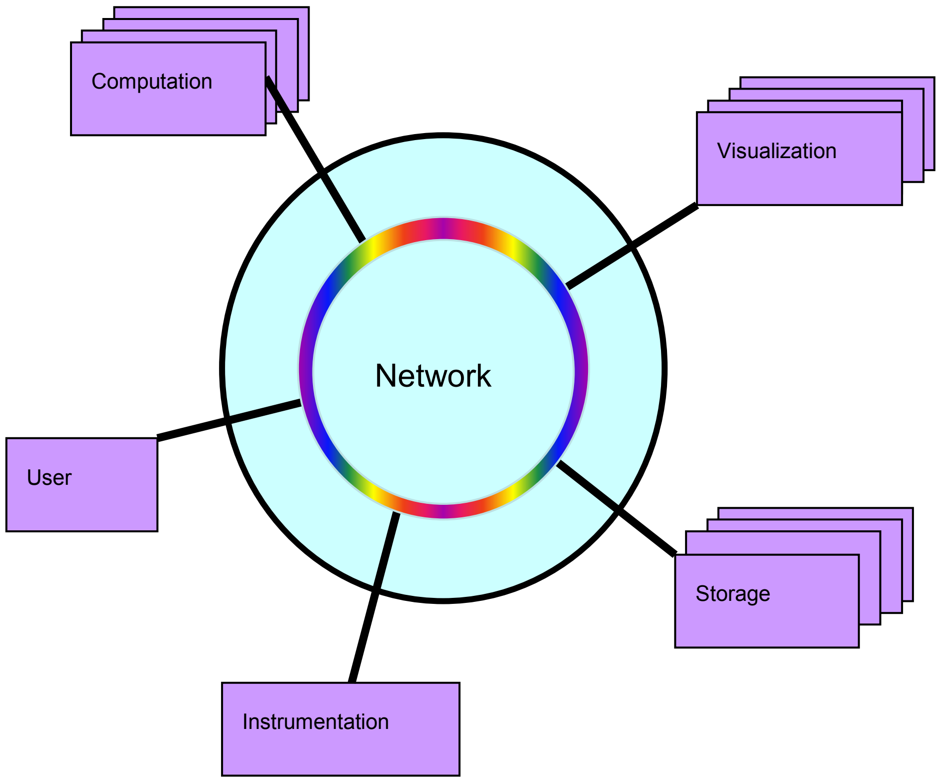


Figure 4.1 – Optical network – A backplane for a globally distributed computation system.

The ambitious vision depicted above cannot be realized due to a fundamental missing link. While the optical capacity has been enhanced at a fast pace, the method to set up the optical network remains static, for point-to-point connectivity. Hence, the ambitious vision of releasing a super-network has not yet been fulfilled. Intelligence is required to utilize this capacity of the optical networks where and when it is needed. Dynamic optical networks can become a fundamental Grid service in data-intensive applications, to schedule, manage and coordinate connectivity supporting collaborative operations. However, the integrated software that provides the intelligent control of a globe-wide distributed system is missing. A suite of software services can serve to 'glue' the optical network backplane to the other resources. Also missing is a global address space that supports the global file system with a global access in a way similar to random access memory (RAM).

4.2.3 New Optimization to Waste Bandwidth

During the last thirty years, a large body of work has been directed towards bandwidth conservation. Many software systems were optimized to conserve network bandwidth rather than computing power. See figure 4.2. While some of Gilder's optical predictions have proven wrong because of the downturn in the technology industry in 2000, his concept of bandwidth conservation remains valid.

"A global economy designed to waste transistors, power, and silicon area and conserve bandwidth above all is breaking apart and reorganizing itself to waste bandwidth and conserve power, silicon area, and transistors."

George Gilder Telecosm (2000)

Figure 4.2– Design to waste of bandwidth - Excerpt from George Gilder Telecosm (2000)

The emergence of optical networks brought a paradigm shift, and now the new focus calls for fully exploiting bandwidth instead of conserving it, with a new balance among storage, computation, and network. Therefore, there is a need to redesign software stacks and protocols so that the central element of a system is the network, not computers. For example, Grid computing can benefit from this new design because Grid services can take advantage of highly available network resources and maximize the capability of computing and data storage. Grid computing has brought a new era of computation, but lacked the network as an essential element without treating the network as an equivalent to computation and storage. While those components such as computation, storage, visualization, are viewed as single independent nodes on a graph, communication is the link between nodes. The effectiveness of Grid computing must take this necessary link characteristic into account. While building the Cyber-infrastructure of Grid computing, it is necessary to add intelligence to the network and to frame the network as a Grid service for scientific workflow and scientific middleware.

4.2.4 Transmission availability

The creation of DWDM initiated a paradigm shift where multiple wavelengths can be transmitted on a single fiber strand, each at a different frequency or color band. Erbium-Doped Fiber Amplification (EDFA) is considered complementary innovation, where amplification is done on the entire waveband without extracting each wavelength separately. These radical phenomena revolutionized communication systems and will soon revolutionize computation systems. Optical networks have seen major advancement in bandwidth and capacity. Currently, a commercial DWDM system can provide as much as 6.2Tb/s of bandwidth, while the bandwidth has reached 26 Tb/s in lab prototypes. DWDM provides parallel Lambdas to drive distributed computation during this decade similar to the way parallel processors drove datacenters in the 1990s.

4.2.5 Impedance Mismatch

DWDM increases the data transport capability of optical networks significantly; however, this leads to impedance mismatching in the processing capacity of a single processor. As is illustrated in Figure 4.3, a network can transfer more data than an individual computer usually receives. While optical transmission consists of many Lambdas at 10Gb/s or 40Gb/s, NICs processing capacity are mostly at 1Gb/s. Therefore, clusters are a cost-effective means to terminate fast transfers because they support flexible, robust, general *N-to-M* communication. Grid computing provides the means for parallelism in distant computation, while DWDM provides parallelism in distant transmission. One goal of this Lambda Data Grid research is to overcome the impedance mismatch and to bridge the gap between massive amounts of computation and transmission.

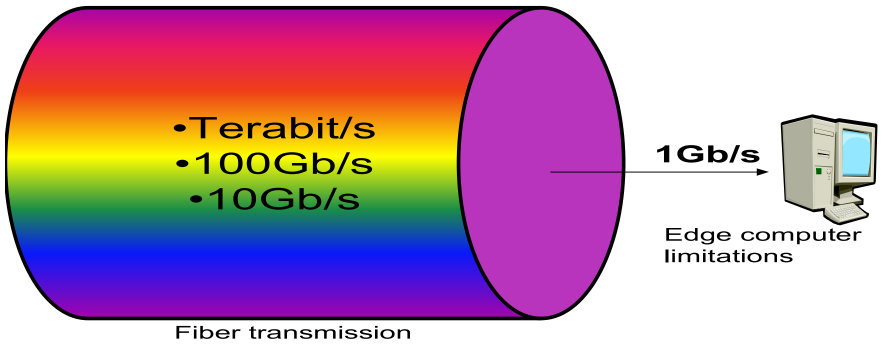


Figure 4.3 - Transmission impedance mismatch – End System Bottleneck.

4.2.6 Affordability and cost

Deployment of new Trans-Atlantic Lambdas has created a new economy of scale. Forty years ago, 300 bits/second between the Netherlands and the USA cost $4.00/minute. Now, an OC-192 (10Gbs) between NetherLight and the USA costs $0.20/minute, with thousands of fibers available: A 600,000,000 times cost reduction per bit.

4.2.7 Cost Prohibitive

Network requirements for our targeted e-Science applications are guaranteed high bandwidth links. Connections around the country would need to have a VPN of 10Gbs by OC-192, and would incur substantial costs for having this service provisioned permanently. A couple of years ago, a coast-to-coast OC-192 service cost about a million dollars per month.

4.2.8 100 Trillion Dollar Investment - Non-scalable

This situation clearly does not scale well with respect to resource utilization, even with a budget at a national level. As illustrated in Figure 4.4, the C x S x V connections of computation-end, storage-end, and visualization-end makes for a meshed network topology that is not feasible in the foreseeable future. Fully meshed static connections among C=50 compute-ends, S=40 storage-ends, and V=100 visualization-ends will require 100 million static connections. Utilizing OC-192 at a cost of $0.5M a year will require an outrageous budget of 100 billion dollars a year. For larger deployment of C=500, S=400, and V=1,000, the investment is about 100 trillion dollars. This is not a scalable solution.

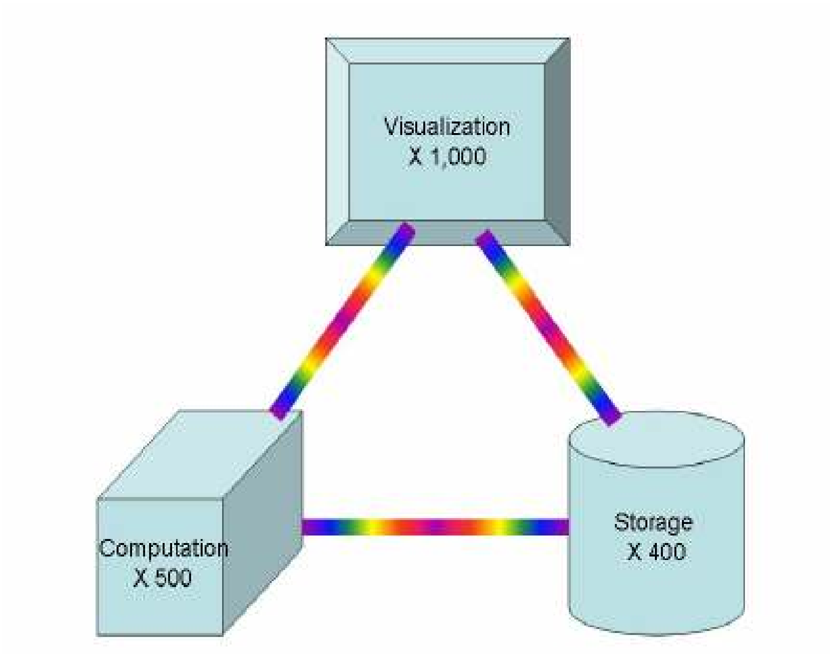


Figure 4.4 – Fully meshed static connectivity is not a scalable solution.

This cost is for the links only and no technology exists to allow dynamic switching of fully meshed network on the edge. The cost of fully meshed technology could be even more than the links calculated above. The limited scalability of this mesh illustration makes it even more prohibitive. With this comparably small setup, adding storage, computation or visualization to the mesh, would require thousands of connections per site making it impractical.

4.2.9 Few-to-Few vs. Many-to-Many

The network requirements of e-Science applications are different from the public internet as described in Table 4.1. From an architectural perspective, this is a few-to-few network compared to many-to-many network. While the public internet has billions of small connections, e-Science topology is comprised of hundreds of large connections, reflecting a completely different topology. The expectation by the public Internet user is continuous connectivity, whereas in e-Science networks the connectivity is expected only for the duration of any scientific experiment. This shift in network availability necessitates a change in user expectations and is approached with dynamic network scheduling. In the public Internet, the connectivity is shared via packet scheduling lasting milliseconds, contrasted to few-to-few networks scheduled for hours or days with full-time, unshared, very large connections.

	e-Science etworks	**Public Internet**
Topology	Few-to-few	Many-to-many
Connectivity xpectation	Per scheduling	Any time
Duration	Hours-days	Milliseconds - econds
Switching echnology	Circuit switching	Packet switching
Core bandwidth	Same as edge	Aggregated
Edge bandwidth	Dozens gigabits-erabits/second	Megabits-igabits/second
Use	Scientific	Consumers/resid ntial/business
Data size	Terabytes – etabytes	Megabytes-igabytes
Pipe utilization	Full capacity	Multiplexing
Computation	Teraflops	Megaflops

Table 4.1 – Few-to-few vs. many-to-many.

4.3 DNA Scenario

A new era of biology is dawning as exemplified by the Human Genome Project. The raw data of DNA sequence information deposited in public databases doubles every six months. In comparison, Moore’s Law predicts doubling silicon density every 18 months. As time passes, this difference is enormous as Table 4.2 illustrates. It is estimated that DNA data generation will grow about 64 times more than computation growth by the year 2010. The growth will be about 16,000 times greater by the year 2015, and will be about one million times greater by 2020.

Years	1.5	3	4.5	6	7.5	9	10.5	12
Year			**2010**				**2015**	
Moor’s aw	2	4	8	16	32	64	128	256
DNA ata	8	64	512	4,096	32,768	262,144	2,097,152	16,777,2
Diff	4	16	**64**	256	1,024	4,096	**16,384**	65,536

Table 4.2: Growth of raw DNA data vs. Moore’s Law.

In the example of protein folding, 30,000 protein structures would require about 800 years of computer time on a high-end personal computer, based on the currently availability in a public database. This can be computed within several weeks if several super-computing centers work with parallel teraflops computing, and with massive data exchanges.

With the explosion of DNA data, the conventional solution that all the data is copied to a local storage in a super-computer center is no longer practical. To function effectively, only the relevant data is copied and staged into the local storage, on demand, and only when it is needed. As part of the computation, the application does not know the exact location of the DNA data needed in advance. The application can reference the data using the bioinformatics middleware, and translate the data structure needed to the right location. When dealing with computation of small data sets, the data can be copied to local memory from a local disk, from local storage attached to the cluster, or from a SAN. When dealing with a large amount of distributed data, new approaches can be developed like Storage Resource Broker (SRB) at UCSD and SDSC. However, when dealing with the anticipated DNA data, SRB over public network will not be able to satisfy the storage, computation and network needs. In our approach, the Lambda Data Grid can be extended to be part of the addressable data storage over a distributed system.

The interaction between the BIRN middleware and the Lambda Data Grid is presented in Figure 4.5. The architecture consist of four layers: the transmission plane, the optical control plane, the network service plane, and the grid data service plan. The Data Transfer Service (DTS) and the Network Resource Service (NRS) are interacting with the BIRN middleware, workflow, NMI, and the resource manager to orchestrate the resources based on the requirements.

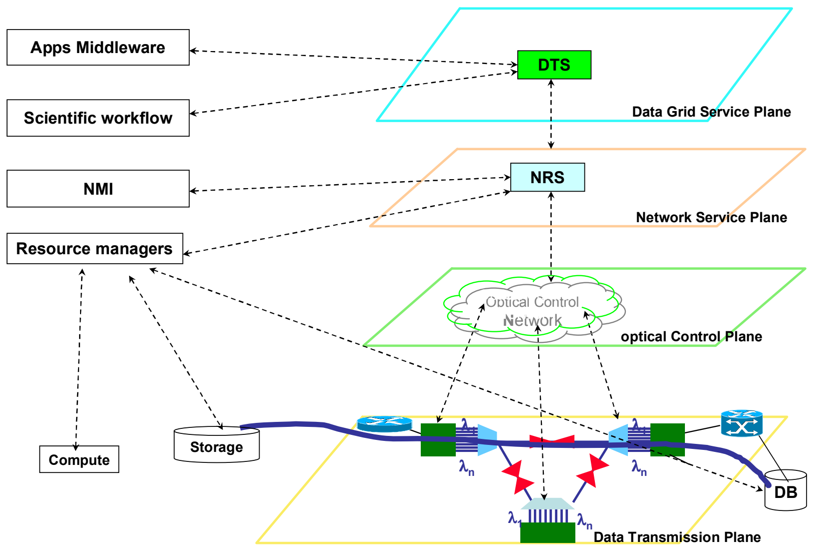


Figure 4.5–Preview for layered interaction between BIRN and Lambda Data Grid.

4.4 Lambda Data Grid Middleware

In Lambda Data Grid (LDG) architecture, the Resource Middleware layer provides OGSA-compliant services that satisfy the resource requirements of the application, as specified or interpreted by the Application Middleware layer services. This layer contains interfaces and services that initiate and control sharing of the underlying resources, including scheduling and reservation services.

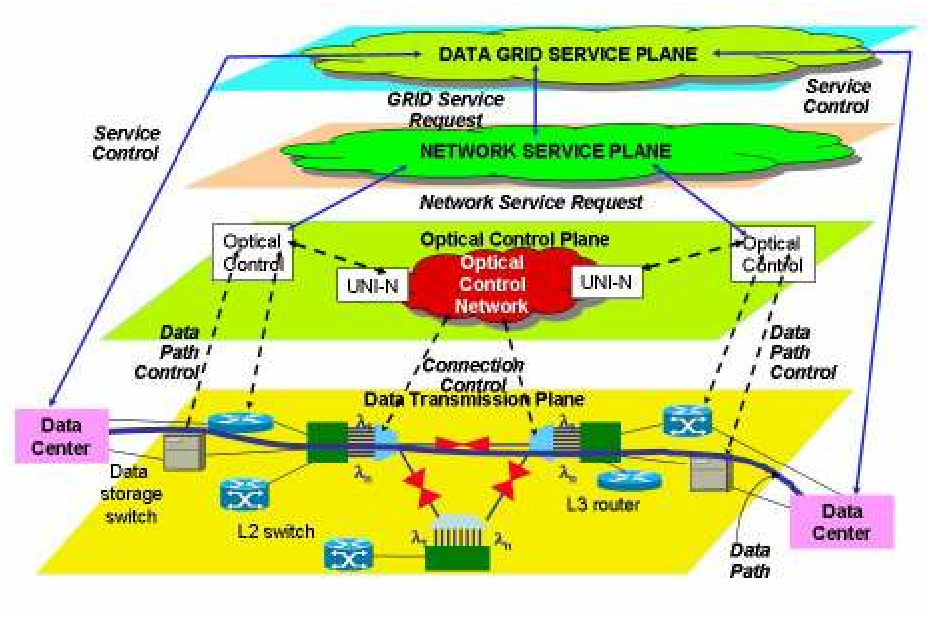


Figure 4.6–Functional interaction between Lambda Data Grid layers.

A high-level view of a Lambda Data Grid is presented in Figure 4.6. The Data Grid service plane sends Grid Service requests to the Network Service plane. The Network Service plane sends the requests to the Optical Control plane, which sends connection control messages to the Data Transmission plane. The scientific workflow is creating the Service Control between the

Data Grid service plane and the scientific applications at remote scientific institutes. Figure 4.7 depicts Compute Grid, Data Grid and Network Grid interaction with scientific applications.

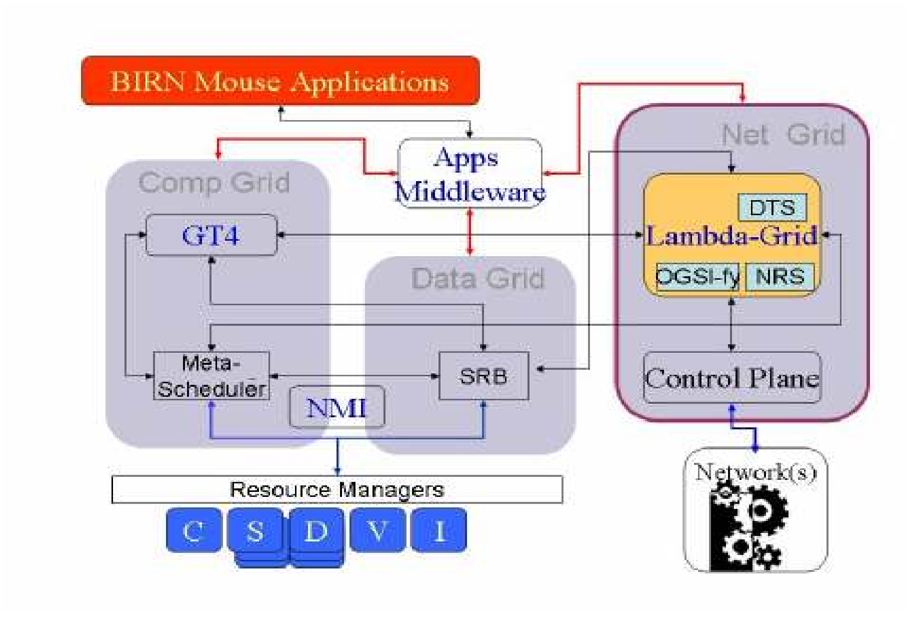


Figure 4.7 – Compute Grid, Data Grid and Network Grid interactions.

In this thesis, we have designed and built architecture that allows applications to interface directly with optical networking control, and entirely bypasses the networking layer architecture. Today's L3 networking approach for Grid Computing is not optimal for high-transport systems. For very large data transfer applications, packet switching architecture has major limitations. Dedicating Lambda in the core of the underlying transmission will be of great benefit to the Grid. In existing Grid architecture, networking is one of the limiting factors. Data must be copied to local storage in the computation machine room before processing. If it were possible to allocate on-demand Lambda, we could avoid this copy, and it would be possible to work on the data in a true distributed fashion. The data could be on one side of the globe, while the computation could be on the other. Providing an on-demand, dedicated, high-bandwidth, low-latency link could dramatically change the distributed mechanisms of Grid Computing applications. Dedicated Lambda on-demand for applications opens a new frontier to new types of applications and research that is not available today with L3 limitations.

4.5 Requirements

Available	Requirement
Static	Dynamic
Silo	Shard
Physical	Virtual
Manual	Automatic
Applications	Service

Table 4.3 – Requirements for e-Science applications.

Given the massive amounts of data, e-Science applications require dedicated high bandwidth optical links to specific data-intensive peers, on demand. It is necessary for the network to be transparent for the applications. Applications and scientific workflow need not undergo major changes in the networking design; they should be network independent. Likewise, no major changes are needed in networking requirements by end-systems or the aggregation of edge-device L3. Rather, changes happen in the underlying optical core transport with lightpath granularity. In this thesis, the focus is granularity of dedicated Lambdas, but this approach can be extended to different granularity like nx STS-1 or nx STM-1 [71]. The edge of this network aggregates traditional L3 IP at 1Gbs and multiplexes them into 10GE. These 10GE aggregations are mapped to the right lambdas towards the destinations.

Due to the enormous amounts of data processed by the applications targeting, this Lambda service is on a separate private network. To address some problems related to the limitations of L3 and routing, an optical forwarding plan of dedicated point-to-point links is a viable solution.

L3 QoS/SLA [72] was originally considered; however, this solution neither meets the network requirements associated with projected increases in amounts of data, nor addresses the issue of the expanded traffic expected from reduced computation and disk costs.

4.6 GLIF and Optical Bypass for e-Science Apps

Deploying optical infrastructure for each scientific institute or large experiment would be cost prohibitive, depleting any research budget. For temporary, scheduled use of the network, a shared system is necessary. Therefore, many organizations around the world, mainly government sponsored scientific network organizations, with ownership of optical links collaborated to build the Global Lambda Integrated Facility (GLIF) [61] consortium.

Figures 4.8 and 4.9, depict the GLIF network topology as of August 15, 2005. The world Lambda topology depicts the hub-and-spoke network model. It can be best described as dual-hubs in Amsterdam and Chicago, with many spokes to the rest of the globe. Among the second hub tiers are NY, Geneva (CERN), Seattle, Sunnyvale, Los Angeles, San Diego and others in East Asia. In the US, the topology is very simple and consists of several fiber rings with a major hub in Chicago. Unlike the Internet topology of "many-to-many," in GLIF topology there are only a few dozen nodes, and is considered to be "few-to-few" architecture.

Lambda Data Grid architecture proves to perform best in relation to a "few-to-few topology." GLIF partners are optimistic about the design and the effectiveness of the concepts presented in this thesis because the concepts are seen as valuable solutions to GLIF requirements. Chapter 7 describes several demonstrations [73] done over GLIF trans-Atlantic Lambdas between Chicago and Amsterdam, with collaboration with StarLight [62], OMNInet SURFnet [63], NetherLight, Internet-2, and CANARIE [30].

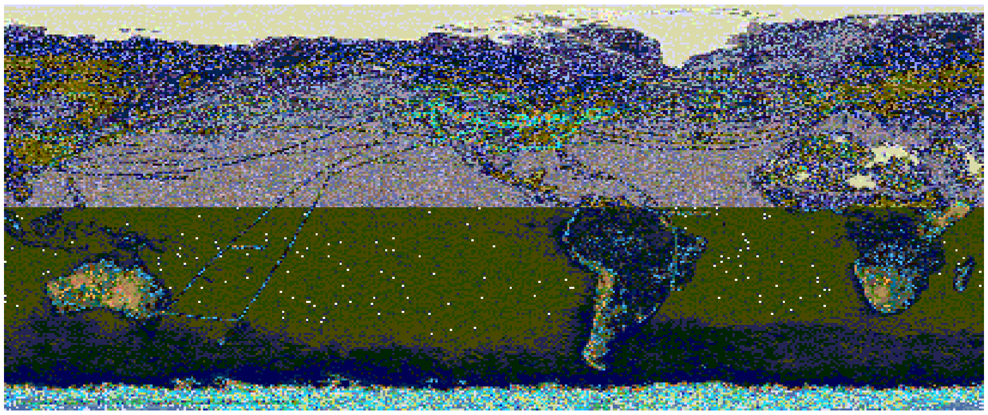


Figure 4.8 – The GLIF Lambda connectivity map (August 15, 2005).

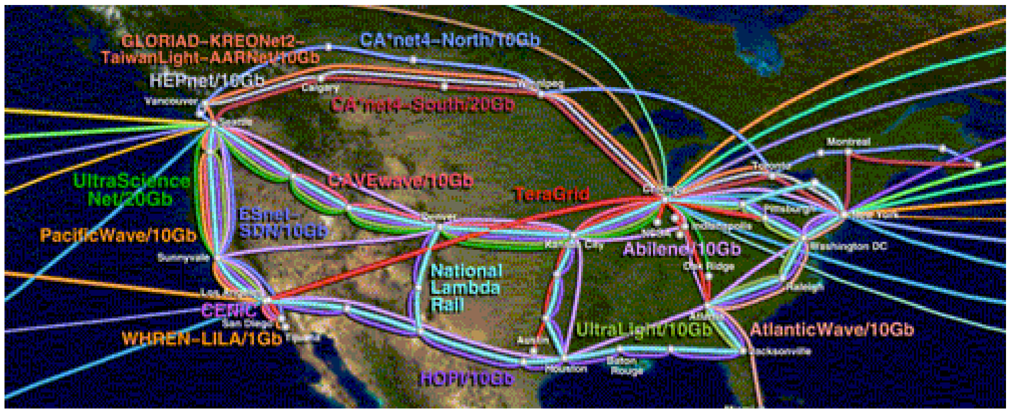


Figure 4.9 – North America GLIF (August 15, 2005)

The natural place to install the Lambda Data Grid Fabric Layer middleware is on network servers in close proximity to GLIF [61] nodes. Upper layers are installed in close proximity to the scientific computation and storage facility. What follows is a discussion of the topology and the network architecture, and specifically optical bypass as an underlay network for bulk connectivity and collaboration between scientific organizations.

4.6.1 Cut-through

As part of the overall architecture, the Lambda Data Grid acquires knowledge of the communication requirements from applications, and builds the underlying cut-through connections to the right sites of an e-Science experiment. Optical Bypass for IP traffic is an alternative mechanism to off-load capacity from L3 routers in case the required capacity exceeds the available router capacity. The Lambda Data Grid does this at the edge devices, without requiring application-level changes to end-systems. It is understood that once router traffic grows tremendously, IP becomes inefficient and optical bypass is a preferred method for high-throughput data transport. While IP services are best suited for the many-to-many paradigm, optical bypass service is ideal in the peer-to-peer occurrences of large data transfers between application peers.

The development of optical transport has brought a huge supply of network bandwidth. The cost per bit is about one order of magnitude more expensive for IP traffic than for optical transport. Optical Bypass can be realized by a network service that sets up an L1 optical shortcut directly between designated end-points or edge devices, and directs data traffic over the shortcut to bypass the IP cloud. An optical shortcut, for example a point-to-point wavelength, provides a large bandwidth pipe for high-throughput data transport in Grids. Grid applications can utilize Grid resource brokerage services to negotiate with an Optical Bypass service to satisfy the requirements and network characteristics of high-throughput file transfer. It is assumed that the bypass negotiation can use some level of resource policy and Authentication Authorization and Accounting (AAA).

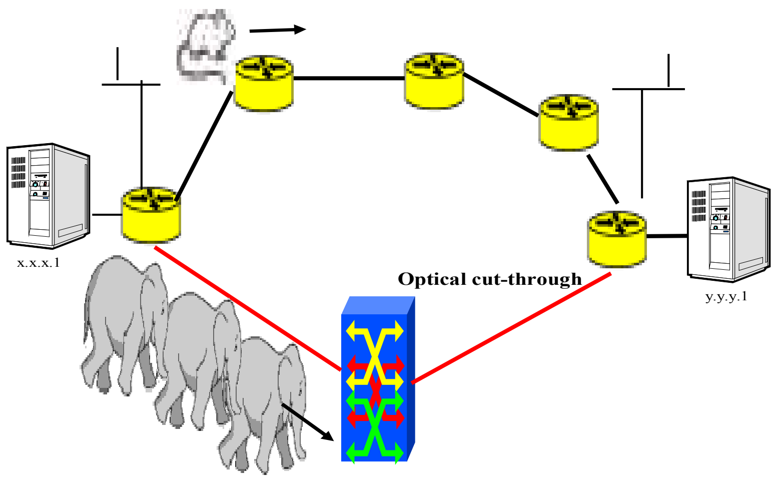


Figure 4.10 – Optical bypass - Mice and a herd of elephants.

An optical shortcut is neither a fixed optical network, nor a leased optical link. It is dynamically created to satisfy high-throughput data transfer for a certain period. It is torn down when the data transfer ends. A Grid application first makes a request of a large data transfer, then a middleware service sets up the optical bypass and causes the traffic to flow via the optical network instead of the IP cloud. The data transfer can be transparent with no changes in the applications, end-systems, or the IP network. The new bypass capabilities are installed at the edge devices. The Optical Bypass service mechanisms and the edge device must conform to the end-to-end design principle and the fate-sharing design principle, and must not add any requirement on the application side. The network control plane, application-aware intelligence, and the Grid middleware are integrated to determine the right path out of the edge device, either the IP cloud or the optical bypass. The optical control plane discovers, identifies and sets up the lightpath when necessary.

In some cases, the lightpath setup is provided by a different entity or administrative domain, or via the user who owns the fibers; hence, the data transfer bypasses the public network. With the help of intelligent network services in both Grids and optical networks, an optical bypass will be created, with sufficient bandwidth between the source and destination to support the requested transfer. Moreover, the data traffic of the file transport is routed via the edge device to the optical bypass. Once the data transport completes, the optical bypass is released or redirected for other Grid uses.

4.6.2 Control Challenge

Figure 4.11 illustrates our Control Challenge demonstration at the SuperComputing conference, using GLIF optical connection and some aspects of Lambda Data Grid. While in many other demonstrations the focus was the throughput, our focus was the control. We addressed the challenge of how to finesse the control of bandwidth across multiple domains while exploiting scalability and intra-domain and inter-domain fault recovery through layering of a novel SOA upon legacy control planes and NEs. The details of this experiment and demonstration will be discussed in Chapter 7, while the architecture will be discussed in Chapter 5.

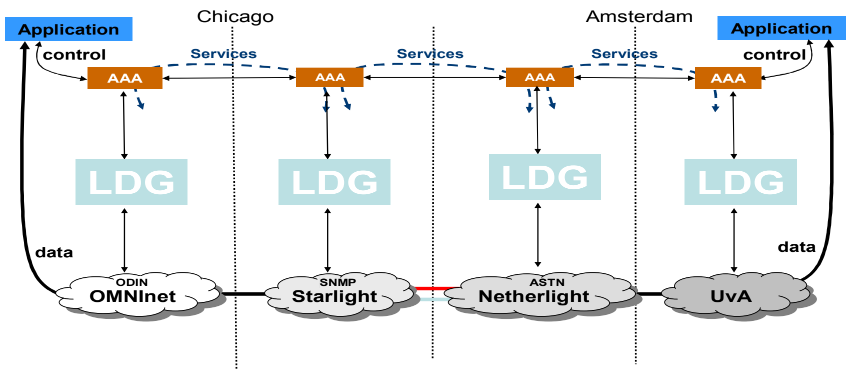


Figure 4.11 – Lambda Data Grid Topology

4.7 Summary

This chapter presented a bird's eye view of Lambda Data Grid as part of Cyber-infrastructure for building network for e-Science and depicting the use for the BIRN project using cut-through mechanism over the GLIF. Some important concepts that inspired the innovation of Lambda Data Grid have been discussed in this chapter. Key enabling technologies for transforming local super-computers to globe-wide super-computing networks are presented in terms of Lambda Data Grid (LDG). The topology of the global fiber plant is introduced with GLIF and the cut-through topology to bypass scientific traffic. The concept of the few-to-few Lambda Data Grid (LDG) connectivity is compared to many-to-many Internet connectivity. The trans-Atlantic demonstrations between Amsterdam and Chicago have been realized through experiments. A major focus in this chapter was the bird's view and the grid's topology. In the next chapter, the focus will be on the service orchestration architecture that supports this few-to-few topology.

5 Lambda Data Grid Architecture

The previous chapter presented a bird's eye view of Lambda Data Grid as part of Cyber-infrastructure for building network for e-Science. We discussed some of the fundamentals for the realization of planetary scientific collaboration and the GLIF as a global optical connectivity of few-to-few topology for scientific experiments. It also presented the availability of data transmission capabilities in these networks, and the vast amount of data to be processed in e-Science research. The missing link is the ability to provide the effective transmission capabilities of these networks to the right scientific research when needed. Chapter 5 presents the Lambda Data Grid architecture that provides the missing link to allow network orchestration for e-Science applications.

5.1 Service Orchestration

The following is an introduction of service architecture that closely integrates a set of large-scale data services with those for dynamic wavelength allocation through a network resource service. The architecture provides means for applications to use data services to capture the benefits of the vast and adaptive bandwidth capacity of the underlying dynamic optical networks. The work described in this chapter takes advantages of closely integrating Grid services, large-scale data flow capabilities, and agile optical networks. In particular, the Lambda Data Grid architecture addresses the complex issues related to orchestrating and integrating application requirements, large-scale data services, and agile optical provisioning services, based on dynamic optical path provisioning. The architectural services components presented includes mechanisms for application resource requests, resource discovery, and a specific set of defined data and optical path provisioning services, with options for both on-demand and scheduled implementations. Scheduling requirements introduce difficult issues relating to variants in provisioning timeframes, from instantaneous to lengthy, among resources, e.g., data, optical paths, and edge devices. Such scheduling also suggests a means to predict future utilization among such resources.

The network resource service allows the orchestration of complex application requirements on resources, data services and dynamic optical path services in a changing environment. Furthermore, the network resource service is compliant with the Web Services Resource Framework (WSRF), allowing large classes of Grid computing applications access to much needed network resources.

5.2 Life-science Scenario

The "BIRN Workflow" addresses the challenges of large and distributed data. The research of the Alzheimer disease requires details of brain analysis over distributed institutions with computations, storages, databases, and visualization tools.

Each brain scan comprises large amounts of data, and comparisons must be made between many scans in disperse geographic locations. A typical brain size is 1,500 cm^3. Table 5.1 represents the amount of data involved in building a brain model. A high-resolution color model with voxel size of 1 **µm,** requires 4.5 Petabytes for a single brain data.

Voxel Size	**B&W (1B/p)**	**High Res (2B/p)**	**Color 3B/p)**
cm	1.5KB	3KB	4.5KB
mm	1.5MB	3MB	4.5MB
0.1mm	1.5GB	3GB	4.5GB
10µm	1.5TB	3TB	4.5TB
µm	1.5PB	3PB	4.5PB

Table 5.1 – Data size for brain model analysis.

With the assumption of analyzing distributed data across the US of from 1,000 brains, the data staging is a big challenge. For voxel size of 0.1mm, color data size for single brain is 4.5GB, and total analysis of 4.5TB for the set of 1,000 brains. For 10µm total size is 4.5PB, and for voxel of 1µm total size is 4.5 ExaByte. This is a challenging problem for itself and cannot scale to a million brain samples in the near future. Clearly, we can see that no existing technology can handle these amounts of data and computation. The storage alone and the computation alone can be handled in the next several years, but clearly not the orchestration that bring these data to computation or the computation to the data.

The BIRN Workflow requires extracting data from remote sites, moving the brain data, analyzing the information, and comparing it to other brain data. The data is in dispersed geographically locations, and on distributed collaborative infrastructures. Each brain data requires vast amounts of computation and involves comparison to many others. BIRN networking over WAN is unpredictable and characteristically has a major bottleneck. The nature of the data in biomedical research prevents true Grid Virtual Organization (VO) research collaboration. The existing network model does satisfy needs associated with the "BIRN Workflow" model, and as such, it is not an integral resource of the BIRN Cyber-Infrastructure.

5.3 Basic Service Architectural Support

The proposed Lambda Data Grid (LDG) architecture, integrated with the underlying optical network, supports:

- both on-demand and scheduled data transmission,
- a meshed wavelength switched network capable of establishing an end-to-end lightpath dynamically,
- bulk data-transfer facilities using Lambda-switched networks, and
- out-of-band utilities for adaptive placement of data replicas.

The distributed component services control a variety of resources, including datasets residing on data nodes, computing power on compute nodes and storage capacity on storage nodes. Those nodes are connected by a set of access and optical networks. The topology of network is unconstrained. Given any two points, the optical network can be asked to reserve or construct an end-to-end path on demand, including lightpath, with some parameters, such as QoS parameters. To perform the schedule optimizations discussed later, it is necessary that the optical network has the intelligence to take some inputs and to choose a path when multiple paths are possible. The data-intensive service architecture handles all aspects of resource orchestration: discovering and acquiring appropriate resources regardless of locations, coordinating network resource allocations and data resource allocations, making plans for optimal transfer, initiating and monitoring the execution of the resources and the transfer, and notifying the client of the status of the task.

5.4 Architectural Platform

To orchestrate service, it is necessary to deploy next generation optical networks as a "first class" resource, similar to computation and storage resources in the grid resource allocation mechanisms, and use them to support Grid applications. Furthermore, Grid middleware must have the ability to allocate and schedule these resources. As noted, this is a complex issue, and many research projects exist that are directed toward addressing one or more aspects of the problem. Our work addresses both data services and dynamic transport issues based on a set of transcontinental experiments and research results; we have designed an architectural platform to orchestrate resources for data-intensive services, through dynamic optical networking.

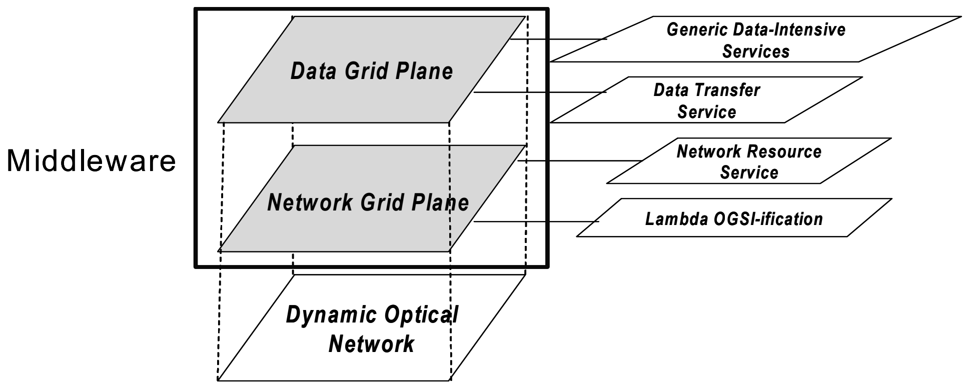


Figure 5.1 – A plane view of the architecture.

This platform separates two sets of concerns: one deals with the application data and user requests, and the other deals with the underlying resources.

As shown in Figure 5.1, the architecture identifies two distinct planes over the underlying dynamic optical network: 1) the Data Grid Plane that speaks for the diverse requirements of a data-intensive application by providing generic data-intensive interfaces and services, and 2) the Network Grid Plane that marshals the raw bandwidth of the underlying optical network, exposes it as a set of network services that are OGSA compliant and are implemented using WSRF, and matches the complex requirements specified by the Data Grid Plane.

We designed these layering services to allow abstraction of the different functionality of the service. Each layer provides a different set of services and mechanisms and each set interacts with a different set of middleware and components. Additionally, the actual physical location will be determined by optimization of the specific distributed system during deployment. It is possible that different components in different layers in different states or even continents.

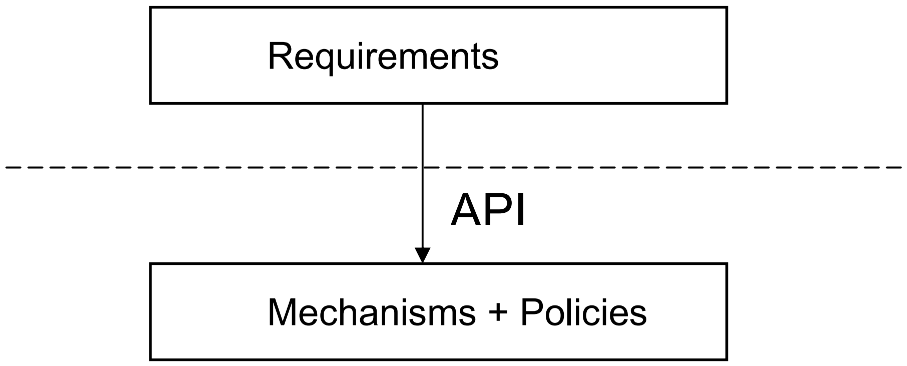


Figure 5.2 – Distinct boundaries between elements.

The interaction between the requirements and the mechanisms and policies is shown in Figure 5.2. At the top level, the scientific workflow requirements will interact with the DTS through and API where the DTS will perform the mechanisms and policies.

Figure 5.3 is a generalized illustration of the middleware system architecture. Two service layers, the application middleware layer and the resource middleware layer, lie between an application and the underlying optical network. The application middleware layer presents a service interface to users/applications and understands the requirements of the application. This layer also shields the application from all the complexities of sharing and managing the required resources. At this layer, low-level services offered by Grid resources in the resource middleware layer are orchestrated into high-level services such as Workflow, service discovery, replication, and data transfer.

The resource middleware layer provides services that satisfy the resource requirements of the application, as specified or interpreted by the application middleware layer. This layer presents a service interface which abstracts the details concerning the specific underlying resources and switching technologies to the layer above. This layer contains capabilities that initiate and control sharing of the underlying resources as well as service components for managing Grid resources such as network, processing, storage, and data handlers. The underlying network and its associated protocol provide the connectivity and fabric for the application.

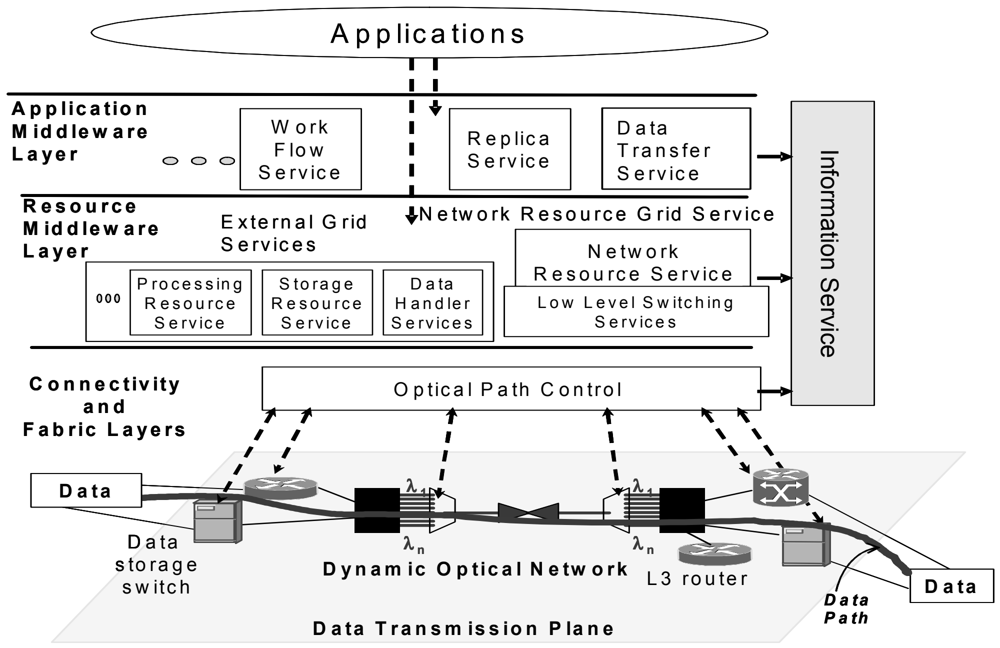
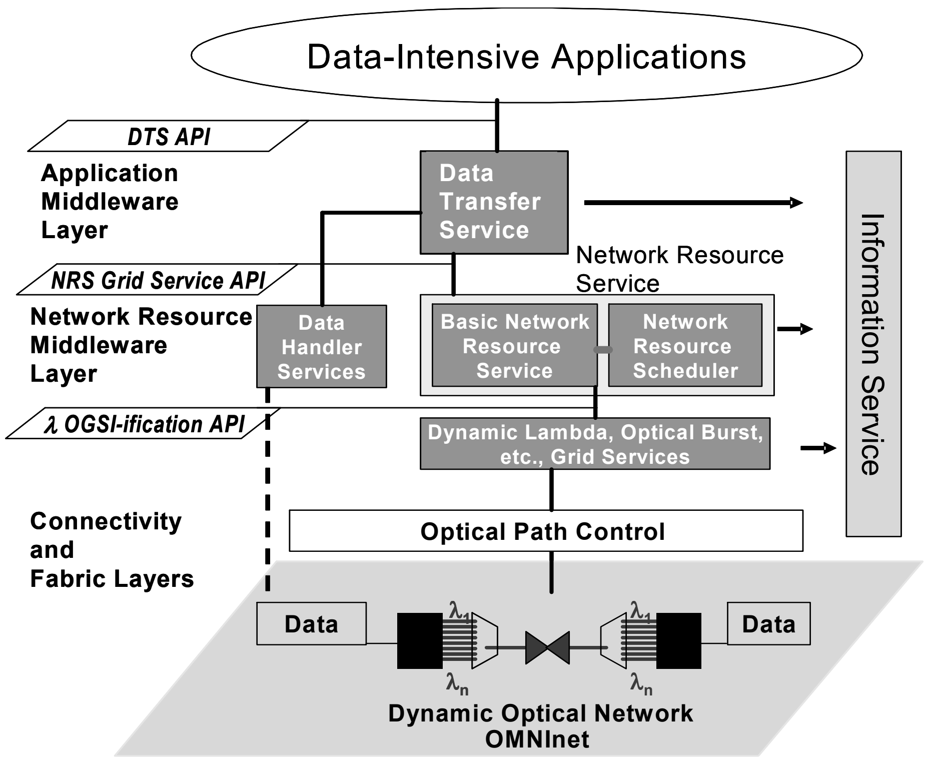


Figure 5.3 – A generic middleware system architecture.

The architecture is flexible such that it can be implemented as layers or modules via an object-oriented approach. As layer architecture, its components can be realized and organized hierarchically according to the service layer. A component at a lower layer provides services to a component at the layer immediately above it. As modules via an object-oriented approach, the architecture provides a more flexible interaction among its modules. Each module is a service, which can be composed of other services. A Grid Services interface between services was implemented to insure well-defined interaction.

Conceptually, the architecture supports data-intensive services by separating itself into three principal service layers: a Data Transfer Service layer, a Resources Service layer and a Data Path Control Service layer, over a Dynamic Optical Network as shown in Figure 5.3.

The role of network services in a scheduled transport of data between application endpoints is a concern. We implemented a Lambda Data Grid architecture that addresses the complex integrated issues concerning mainly scheduling network resources and data transfers. In Figure 5.4, we present a simplified view of my architecture, with select components of the general architecture. We have designed the layering of the middleware and the APIs between the layers to encapsulate the different functionality of the middleware components. Each of the middleware layers is responsible for a different functionality and interacts with different components of the computation storage and networking of scientific experiments. It is more effective to deploy the specific layer to be physically co-located with the other resource of the interaction. For example, in some of the experiments, the optical control middleware was in Chicago, while the Data Transfer Service (DTS) was collocated near the data in the SARA supercomputing center in Amsterdam.



Data-Intensive Applications
DTS API
Application Middleware Layer
Data Transfer Service
NRS Grid Service API
Network Resource Service
Network Resource Middleware Layer
Data Handler Services
Basic Network Resource Service
Network Resource Scheduler
λ OGSI-ification API
Dynamic Lambda, Optical Burst, etc., Grid Services
Information Service
Connectivity and Fabric Layers
Optical Path Control
Data
Data
λ_1
λ_n
λ_1
λ_n
Dynamic Optical Network OMNInet

Figure 5.4 – Lambda Data Grid System Architecture.

5.4.1 Application Middleware Layer

At the application middleware layer, the Data Transfer Service (DTS) presents an interface between the system and an application. It receives high-level client requests, policy-and-access filtered, to transfer named blocks of data with specific advance scheduling constraints. It employs an intelligent strategy to schedule an acceptable action plan that balances user demands and resource availabilities. The action plan involves advance co-reservation of network and storage resources. This middleware layer shields the application from lower level details by translating application-level requests to its own tasks of coordinating and controlling the sharing of a collective set of resources.

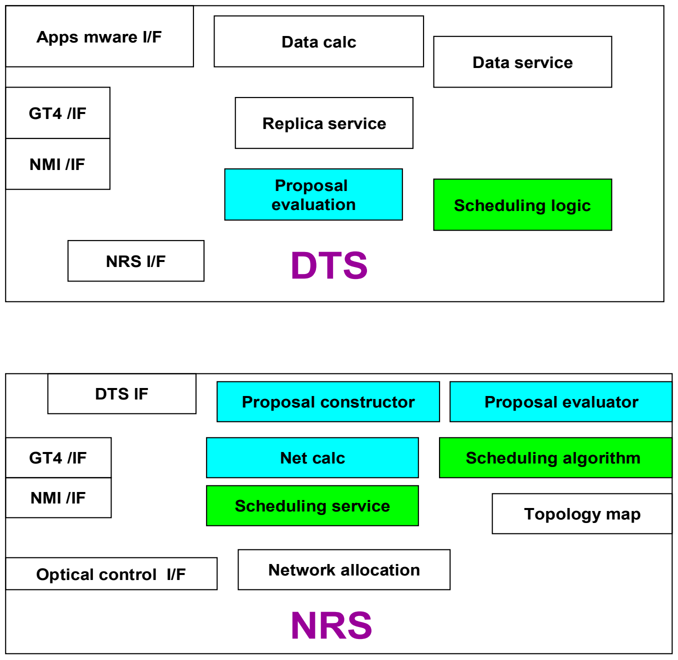


Figure 5.5 – NRS and DTS Internal Architecture and Interfaces.

5.4.2 Network Resource Middleware Layer

The network resource middleware layer consists of three services: the Data Handler Service (DHS), the Network Resource Service (NRS) and the Dynamic Lambda Grid Service (DLGS). Services of this layer initiate and control the sharing of resources. The DHS deals with the mechanism for sending and receiving data and effectuates the actual data transfer when needed by the DTS. A central piece of this layer, the Network Resource Service (NRS), makes use of the Dynamic Lambda Grid Service in encapsulating the underlying optical network resources into an accessible, schedulable Grid service. The NRS receives requests from the DTS, as well as requests from other services such as external Grid services, both scheduled and on-demand. It maintains a job queue and allocates proper network resources according to its schedule. To allow for extensibility and reuse, the Network Resource Service can be decomposed into two closely coupled services: a basic Network Resource Service and a Network

Resource Scheduler. The Basic Network Resource Service handles multiple low-level services offered by different types of underlying networks and switching technologies and presents an interface to the Data Transfer Service for making network service requests. The Network Resource Scheduler is responsible for implementing an effective schedule that facilitates network resources sharing among multiple applications. The Network Resource Scheduler can be deployed independent of the Basic Network Resource Service. The Dynamic Lambda Grid Service receives resource requirement requests from the NRS and matches those requests with the actual resources, such as path designations. The Dynamic Lambda Grid Service can establish, control, and deallocate complete paths across both optical and electronic domains.

5.4.3 Data Transfer Scheduling (DTS) Service

The DTS service is a direct extension of the NRS service. The DTS shares the same back-end scheduling engine and resides on the same host. It provides a higher level functionality that allows applications to schedule advance reservations for data transfers without the need to directly schedule path reservations with the NRS service. The DTS service requires additional infrastructure in the form of services living at the data transfer endpoints. These services perform the actual data transfer operations once the network resources are allocated. As with the NRS, the DTS started with an Open Grid Service Interface (OGSI) and moved into the new Web Services Resource Framework (WSRF). Globus Architecture for Resource Allocation (GARA) integration is a planned avenue for future development. As an extension of the NRS, the DTS also employs under-constrained scheduling requests to accommodate new requests and changes in resource availability. For further discussion of this, see Chapter 6.

It is important to note that in the Lambda Data Grid design the application at the endpoints drives the state and decisions by DTS, NRS, and ODIN; thus, is in full compliance with the end-to-end argument. The NRS retains a copy of the network topology, technology and capacity and maintains the local state of the resources. These intervening layers never attempt to reconstruct nor second-guess an application's intent from traffic. Instead, they follow the explicit command and control from the application. These layers are best positioned to appreciate both application's commands and network status, while bridging meaningfully between the two.

Additionally, Lambda Data Grid complies with the so-called fate-sharing principle. This design principle argues that it is acceptable to lose the state information associated with an entity if at the same time the entity itself is lost. In this case, DTS, NRS, and ODIN will make a point of reclaiming resources once an application is terminated, or is found to be unresponsive for a configurable time interval.

5.4.4 Network Resource Scheduling (NRS) Service

The Resource and Connectivity protocol layers form the neck of our “hourglass model” based architecture. The NRS is essentially a resource management service, which supports protocols that offer advance reservations and QoS. It maintains schedules and provisions resources to support these schedules. It is OGSA compliant and is implemented using a WSRF compliant interface to request the optical network resources. It has complete understanding of dynamic lightpath provisioning and communicates these requests to the Fabric Layer entities. In this architecture, the NRS can stand alone without the DTS (see Collective layer). GARA integration is an important avenue for future development. A key feature is the ability to reschedule reservations that satisfy under-constrained scheduling requests, to accommodate new requests and changes in resource availability. Requests may be under-constrained through specification of a target reservation window, which is larger than the requested duration, through open-ended specifications, such as "ASAP,” or through more sophisticated request specifications such as client-provided utility functions for evaluating scheduling costs. Models for priorities and accounting can also be incorporated into the scheduling algorithms.

5.4.5 Grid Layered Architecture

Our architecture is illustrated on the left side of Figure 5.6. It parallels the Layered Grid Architecture proposed by Ian Foster [44] shown on the right side of Figure 5.6. These two-layered architectures are depicted side by side to illustrate the differences and the similarities of functions in each layer. While the Grid Layered architecture is general to all components of Grid Computing including storage and computation, The Lambda Data Grid Layered Architecture is geared specifically towards optical networking. Our DTS is adjacent to the Collective Layer, coordinating multiple resources. Our NRS/DLGS is parallel to the Resource Layer, controlling the sharing of single-network resources. Our Optical Path Control is adjacent to the Connectivity Layer, for connectivity provisioning. Our Lambda Layer functions similarly to the Fabric Layer. The architecture is sufficiently flexible and can be implemented as layers or modules via an object approach. The focus in the Grid Layered Architecture is the computation, storage, and scientific collaboration with minimal control of the network. Foster's approach reflects full control of the local computation, storage, middleware and application in the data centers, whereas the network is managed by the service provider. In this case, the Grid does not have control of the network resources. In our approach, the Grid has control of the optical network, as it is a local resource similar to other local resources.

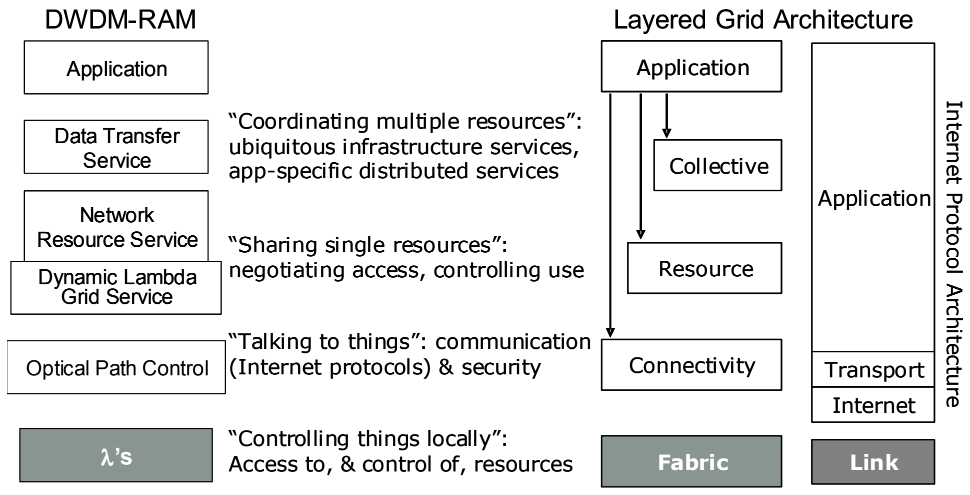

DWDM-RAM
Application
Data Transfer Service
Network Resource Service
Dynamic Lambda Grid Service
Optical Path Control
λ's
"Coordinating multiple resources": ubiquitous infrastructure services, app-specific distributed services
"Sharing single resources": negotiating access, controlling use
"Talking to things": communication (Internet protocols) & security
"Controlling things locally": Access to, & control of, resources
Layered Grid Architecture
Application
Collective
Resource
Connectivity
Fabric
Application
Transport
Internet
Link
Internet Protocol Architecture

Figure 5.6 – Lambda Data Grid vs. Layered Grid Architecture (adapted from [29]).

5.4.6 Allocating Bandwidth On-Demand

For wavelength switching to be useful for Grid applications, a network service with an application level interface is required to request, release and manage the underlying network resources. Previous work has been done in defining the OGSA compliant and implemented using OGSI based network service interface for network resources. CANARIE's UCLP, for example, is geared towards the ownership of fibers by the scientific institute, and allows Lambda allocations by the user instead of the service provider. Our approach follows a similar model with the addition of a comprehensive schedule and reservation-based Network Resource Scheduling (NRS) service, which provides user applications with access to underlying optical network resources. This service should guarantee dedicated access to the optical links, which may be requested on-demand or by advance reservation. Advance reservation requests can be under-constrained, which means the request can be satisfied by more than one possible time slot. This allows the service to reschedule reservations as needed to satisfy future requests and changing conditions.

The NRS also attempts to provide an interface with different levels of detail. Some coordinating and scheduling clients may need only high-level facilities like those for individual applications. For these applications, the interface should allow the request of Lightpaths, but these clients do not need to have knowledge of the details of the underlying network topology or management protocols. However, clients that attempt to optimize network resource use may need a richer interface, e.g., they may need to be able to schedule individual optical segments. For these clients, the interface should allow the request of individual segments. Although these clients may need knowledge of the network topology, they should still be insulated from the network management protocols.

The NRS interface suggests likely properties for a service that might implement this interface. Such a service can be implemented in a way that does not maintain any session or context information for any particular client between calls. The only necessary context information is the allocated path identifier, which the client is required to supply to deallocate a path. The service must maintain this information about these allocated paths so, in this sense, it is not "stateless," but each client call can be treated as a self-contained unit and processed entirely in a single message exchange. Thus, the interface fits the service-oriented architecture of Web Services [74] quite closely.

At the same time, we believe that it is important to conform to the emerging Open Grid Services Architecture (OGSA) Grid standard to be accessible to Grid applications that use this architecture. Hence, the NRS interface also conforms to the Web Services Resource Framework (WSRF) specification. This interface, in its basic form, is a Web Service interface with some additional conventions from the original OGSI standard. Although client access is via a normal Java interface as described above, internally, the client-service interface is a Web Service implemented using SOAP and the JAX-RPC API. The user of the NRS service is spared any direct contact with this underlying middleware.

So far, in this chapter discussed DTS, NRS, NRM, and DHS, the interaction between them as part of the Application Middleware Layer (AML) and the Resource Middleware Layer (RML). What follows is a description of the Workflow Service, and specifically the Scientific Workflows as part of our overall architecture for orchestration.

5.5 Workflow Integrated Network Resource Orchestration

Scientists have the need to access scientific data and perform sophisticated analyses on that data. Such analyses can be described as "scientific workflows" in which the flow of data from one analytical step to another is expressed in a formal workflow language. In this work, we expand on scientific workflow management systems, which can be incorporated on top of Lambda Data Grid, using a number of technologies including Web and Grid services, relational databases, and local applications implemented in various programming languages.

Through dynamic organization of scientific processes and adequate resources, workflow-based Web Services provide a flexible solution to the success of e-Science applications. As a key enabling technology of Web Service compositions, the Grid community moves forward in enhancing workflow processing capability and orchestrating heterogeneous resources among computing, storage, and other resource providers, across network administrative domains. Challenges arise in effectively realizing the workflow-oriented resource orchestration across administrative domains. This problem is compounded when a process uses inter-service data-intensive communications, and has the need for reliability and timeliness. The foremost issues lie in harnessing network resources into schedulable services for composing workflow processes to support service providers. Additionally, these services, along with a number of network-related activities for resource orchestration, must fit the needs of a variety of service-enabling technologies, including Grids, beyond workflow engines. In this work, we tackle the complicated problem from the network middleware perspective instead of the application middleware perspective. We treated the network as a Grid Service with characteristics of allocation, scheduling, and re-scheduling resources that are in conjunction with the availability of other resources and are within application requirements, as part of a scientific workflow.

To meet the needs discussed above, "Workflow INtegrated NEtwork Resource orchestration" (WINNER) was developed. This exposes the method by which network resources can be orchestrated intelligently for workflow applications. The goal of WINNER is to solve issues of two complementary components: workflow processes that provide the activities, and services that realize the activities in such processes. The workflow activities of WINNER are geared toward both workflow applications and network resources. Applications can deploy activities such as resource allocation, reallocation, release and accounting, while network resource providers can utilize resource creation, destroy, and update. Activities common to both include registry, query and secure operations of customers, and network resources. Furthermore, WINNER combines activities with the network services provided by network resource providers. As a result, WINNER allows network elements, and/or network domains, to serve their resources seamlessly for workflow applications. WINNER represents a revolutionary network resource management system that provides the predictable and optimized system performance that is valuable in Grid Computing. It addresses crucial resource issues and dynamic exception-handling workflow architecture, and is applicable to any critical resource management system.

Our approach is built on a unique synergetic relationship between the advanced results in application disciplines (e.g., workflows), operating system disciplines (e.g., real-time value-based scheduling), and networking disciplines (e.g., scalable hierarchies of services). In this approach, we extract application information, communicate with the scientific process workflow, feed this information to the network scheduler, synchronize the network scheduler and the Grid scheduler, and reserve the right communication channel for the right time. The goal is to create a network setup that builds the right network topology, at the time it is needed.

The novelty of the converged WINNER system is such that its quantitative analysis holds potential to be innovative in its own right, with new metrics sought along the dimensions of complexity, scalability, and adaptability.

Along with the workflow process, the workflow services for registration, query, resource administration, job submission, and security enforcement, are essentially Web Services. Moreover, the realization of workflow services can be based on the recent development of WSRF and other Globus technologies. WINNER envisions the use of the appropriate encapsulation of workflow services corresponding to various service-enabling technologies. Consequently, WINNER makes network resources available to workflow applications, whether they work directly with workflow engines or indirectly through Grids, Web Services, and other service environments.

WINNER achieves these goals by defining a workflow process for network resources that features a set of essential activities (e.g., resource allocation, renewal etc.) and by providing the workflow services that support these activities. The activities are designed for applications, network resource providers, or both. Along with the workflow process, the Web Services-based workflow services consist of registration, query, resource administration, and job submission.

Another challenge is in incorporating services that WINNER employs within network resource providers. These services perform resource orchestration through a number of existing and newly introduced network services. These network services can discover available resources in network domains, bind adequate resources dynamically for workflows, and conduct proactive network configurations and resource allocations with resource smarts in SLA, policy, optimization and security. The intersection of application-driven provisioning, constraint-based routing, and scheduling of optical resources warrant further research. A chief goal is to mitigate line-blocking effects while maximizing user satisfaction and network utilization. Furthermore, with the recent advances in workflow language technologies and semantic composition of workflows comes the possibility of greatly enhancing the dynamic binding between applications and the network

5.6 Scientific Workflows

Scientific workflows usually operate on large, complex, and heterogeneous data, integrated by scientists prior to computations. As a result of these computations, complex data is created as input to other workflows. Hence, the academic community is using workflows as a means of orchestrating complicated tasks over a distributed set of resources. More scientific organizations are realizing the benefits of sharing their data and computational services, and are contributing to a distributed data and computational infrastructure. The Lambda Data Grid mechanism provides seamless access remotely to large databases, which allows the opportunity of an integration of the field of scientific logic and Workflows. Some of basic requirements of this new system that need to be addressed are: 1) Seamless access to resources, 2) Scalability and Service reuse, and 3) Reliability and fault-tolerance.

This chapter discussed the Lambda Data Grid layered architecture, including the DTS and NRS. We presented the middleware, service architecture, and Scientific Workflows and provided an example of how these components work to orchestrate network resources as part of scientific experiments. What follow is the implementation details, testbeds and analysis.

6 Testbed, Experiments, Evaluation and Results

In Chapter 5, we discussed the Lambda Data Grid architecture, the service orchestration, and the interaction with Scientific Workflows. Specifically, we described the Data Transfer Service (DTS) and Network Resource Service (NRS), and the interaction of these elements with Grid middleware. This chapter describes our major experiments and demonstrations, with details of the technical architecture, inherent challenges, evaluation of results and analysis. It begins with an explanation of the dynamic provisioning of a 10GE Lightpath across a distance of 10 meters in a lab, using one experimental Micro-Electro-Mechanical Systems (MEMS) switch. Next is a description of the OMNInet testbed connecting multiple network sites in Chicago metro area, using four MEMS switches over a distance of about 10km. Lambda scheduling and dynamic allocation is stated and followed by a presentation of measurements and analysis. In the following section, the concept of dynamic provisioning is applied to alternate Lambda fail-over the cross-Atlantic fiber connections, across a distance of about 10,000km, using the CANARIE and I2, connecting SARA Supercomputing Center at Amsterdam and the Argonne National Lab at Chicago. Another significance in this experiment is the automated and dynamic process as it compares with the conventionally manual and static process, and its ensuing result of an extreme reduction of provisioning time: from about 100 days to 100 seconds. The final two demonstrations show the relevance of this technology to e-Science through intelligent middleware and workflow automation, thereby enabling the scientific process to advance to a new level of possibilities.

6.1 Dynamic allocation of metro network bandwidth

This section describes an early conceptual experiment [41] [75] of dynamic allocation of metro network bandwidth. This early experiment of service platform [76] showed the capability of dynamic allocation of bandwidth using early implementation of Resilient Packet Ring (RPR) assembling a metro network in one rack. Success in implementing this model led to the birth of Lambda Data Grid. Figure 6.1 shows the architecture of the demonstration presented at Globus World [40] in San Francisco.

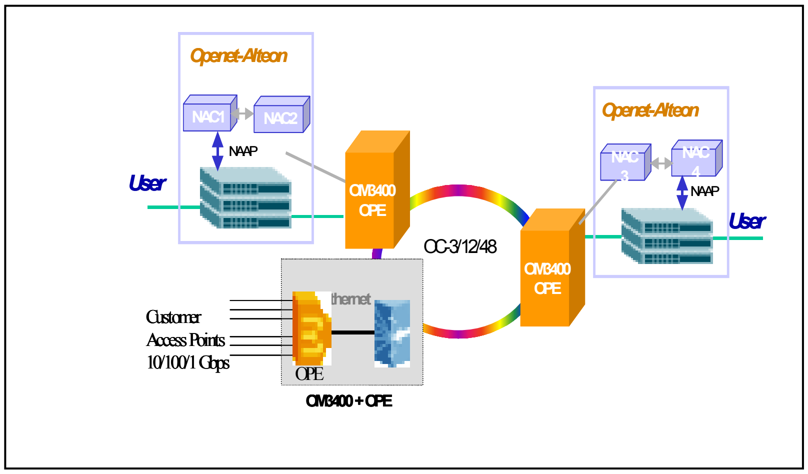


Figure 6.1: Service platform for Dynamic bandwidth allocation over RPR metro network

In this experiment, the edge of the Metro core is a service platform and proves its worth in bridging the solution gap between the users and the Optical core through the Metro. Although the metro core is not optically agile, the service platform can intelligently intercept a user request and intelligently provision bandwidth before user traffic takes its course over the Metro's SONET. This effectively demonstrates that it is possible to control the optical network based on the application requirements and the content by allocating bandwidth to users dynamically. We were able to set and remove a connection between any two points in the metro network. The bandwidth for the connection can be dynamically set to any value between 1Mbps-1Gbps in increments of 1Mbps. The task involved setting and enforcing the allocated bandwidth for the end-to-end path, involving both the electrical side and the optical side.

For the Metro demo, the service platform, the Optera Metro 3400 with OPE were utilized. This device allows configuration of fine-grain Ethernet bandwidth limits on top of Resilient Packet Ring (RPR). RPR was defined by IEEE as 802.17. RPR Metro network is not on the public Internet; it is a secured, isolated, L2 network, without the limitations of L3 and packet switching. The connection bandwidth is guaranteed. The distance limitation of this RPR is 120km, but this distance can be extended to reach a RPR in a different geographical area via a DWDM point-to-point connection. The purpose of this demonstration was to show the feasibility of the concept with manual reservations.

6.1.1 Service Platform

In this demonstration, we built a service platform to provide a network provisioning service to an application. Examples of the applications include: granting *X* Mbps for the duration of a videoconference request; allocating *Y* Mbps for *Z* minutes for a backup application between a San Francisco location and a San Jose location; and advising and provisioning most economical backup for price sensitive applications. In this case, the provisioning took 4.5 seconds and the signaling setup took 30 seconds. This might be seen as a lengthy setup time compared to traditional packet switching; however, this is a single preparation with no need to evaluate every packet. This may appear as an expensive time expenditure, but in the transferring of large amounts of data it is time effective. This early demonstration used RSVP as signaling, on a small network, without any optimization for large signaling time. Our primary goal was to build a proof of concept.

This early proof of concept led to another demonstration using OMNInet testbed that will be described in the next section. In this demo, we were able to show that we can dynamically set the network configuration of the all-optical switches based on application requirements, employing an intelligent traffic filtering property of the platform.

6.2 OMNInet Testbed and Setup

This section describes the OMNInet testbed and the Optical Dynamic Intelligent Network (ODIN). Some of the demonstrations described in this chapter were built on top of the OMNInet testbed with interaction with ODIN. For part of this work, we collaborated with Northwestern University in Chicago. While OMNInet and ODIN are an essential part of the experiments in this thesis, these two components were the responsibility of a different group. This thesis does not claim credit for the development of these components. To fully understand Lambda Data Grid experiments the following is a description of OMNInet and ODIN.

6.2.1 The OMNInet testbed

The testbed for the DWDM-RAM was implemented on a next-generation large-scale Metro/LAN dynamic optical network, which is known as the OMNInet. The OMNInet project is a multi-organizational partnership, which was established to build the most advanced metro area photonic network testbed. OMNInet uses metro dark fiber infrastructure, which has been provided by SBC to connect locations sponsored by Northwestern University in Chicago, and a mix of product and research equipment from Nortel Networks.

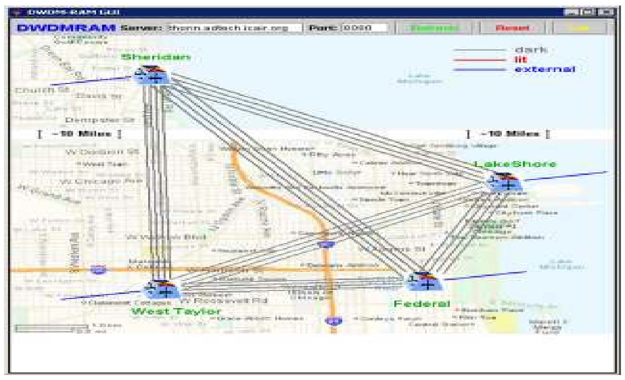


Figure 6.1: OMNInet Topology over Chicago Area Map

OMNInet consists of four photonic nodes and has been deployed at four separate locations in the Chicago metro area. It is configured in a partial-mesh 10GE DWDM network. These nodes are interconnected as a partial-mesh, with lightpaths provisioned with DWDM on dedicated fiber. Each node includes a MEMS-based (Micro-Electro-Mechanical Systems) 8 x 8 Wave Division Multiplex (WDM) photonic switch, an Optical Fiber Amplifier (OFA) and optical transponders/receivers (OTRs), and high-performance L2/L3 switches. The core photonic nodes are not commercial products, but unique experimental research implementations, integrating state of the art components. The OMNInet configuration is shown in Figure 6.2.

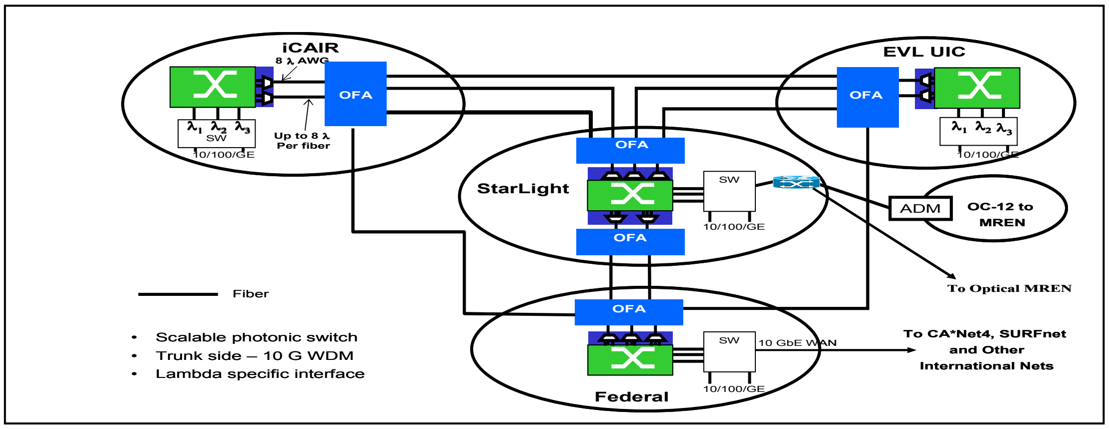


Figure 6.2 – OMNInet Testbed Configuration.

The OMNInet control plane is separated from a data plane, and provisioned out-of-band, using completely separate fiber. Such control planes could also reside on a supervisory lightpath. This control plane enables User-to-Network Interface (UNI) control signaling via a UNI interface to the optical transport network and bi-directional signaling to the connection control plane. A 10GigE trunk interface using 1500nm DWDM wavelengths have been implemented, with a specialized set of protocols that allows for enhanced optical network intelligence, including a lightpath signaling protocol, a lightpath routing protocol, and an optical link management protocol. To provide for reliability and optimal L1 performance, OMNInet is provisioned with sophisticated pre-fault detection mechanisms, which monitor network conditions and adjust resources in response to specific detected characteristics.

6.2.2 Optical Dynamic Intelligent Network Services (ODIN)

Optical Dynamic Intelligent Network Services (ODIN) was developed by Northwestern University. This architecture provides for receiving communication service requests from higher-level processes and translating those requests into network resources, primarily dynamically provisioned paths, lightpaths and extensions of those lightpaths, to edge devices through VLANs.

Optical Dynamic Intelligent Network[1] (ODIN) is implemented as a software suite that controls the OMNInet through various lower-level API calls. ODIN server software is comprised of components that, a) accept requests from clients for resources (the client requests a resource, i.e., implying a request for a path to the resource – the specific path need not be known to the client); b) determine an available path, possibly an optimal path if there are multiple available paths; c) create the mechanisms required to route the data traffic over the defined optimal path

[1] ODIN research was made possible with support from the National Science Foundation: award - ANI-0123399

(virtual network); and d) notify the client and the target resource to configure themselves for the configured virtual network.

The dynamic lightpath provisioning of the OMNInet is provided by the ODIN services module, which manages the optical network control plane and resource provisioning, including path allocation, deletion, and setting of attributes. The technological breakthrough in the design of the OMNInet photonic switches, coupled with the tight integration of the control software, brings down the provisioning time from months to a few seconds. This is the critical enabling factor in on-demand provisioning.

We initiate processes by accepting requests from clients for network resources. The client requests activity that implies a request for a path or paths to resources. Given request attributes and complete knowledge of available network resources, ODIN designates appropriate paths. ODIN also creates the mechanisms required to route the data traffic over the defined optimal path (virtual network), and transmits signals that notify the client and the target resource to adjust in order to match the configured virtual network. An implementation of ODIN has successfully been used for large-scale experiments with science data on the OMNInet testbed. In the ODIN research the focus was the optical and the physical layers; whereas, my work was a collaboration with them to provide the upper layers of intelligence, including scheduling, network services, network middleware, and application interaction as part of my Lambda Data Grid research.

OMNInet was the testbed for some of the Lambda Data Grid demonstrations interacting with ODIN to dynamic allocate Lambda over the Chicago metro area. The results from these experiments were the input for the analysis described later in this chapter.

6.3 Prototype Implementation

An early proof of concept for Lambda Data Grid is that the DWDM-RAM was implemented [32] in a metro area, on top of OMNInet and with interaction with ODIN. The DWDM-RAM architecture is shown in Figure 6.3. This service architecture closely integrates a set of large-scale data services with those for dynamic wavelength allocation [68] through a network resource middleware service [73]. Several layers mediate between the former and latter, while abstracting out low-level resources, and yielding opaque "tickets" to upper layers. Early standardization work in OGSA has shaped some of the layers' interfaces. The architecture provides means for applications to use data services to capture the benefits of the vast and adaptive bandwidth capacity of the underlying dynamic optical networks. This architecture leverages the Globus Toolkit 4's functionality.

The network resource service allows the orchestration of complex application requirements, data services and dynamic Lambda services. Furthermore, the network resource service is fully OGSA compliant, and presents a WSRF to external applications, allowing large classes of Grid computing applications access to much needed network resources.

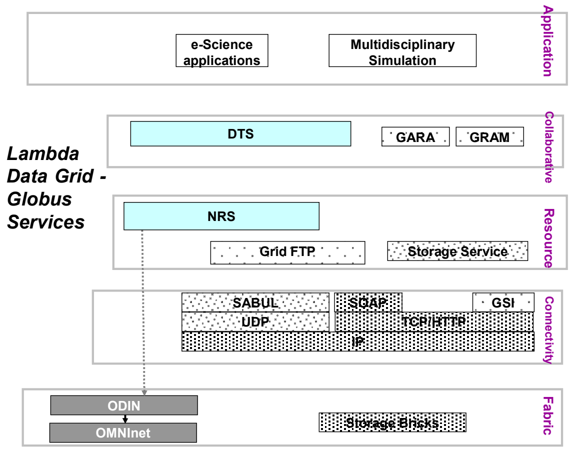


Figure 6.3: The Lambda Data Grid architecture.

The service implementation includes the Network Resource Service (NRS), the Data Transfer Service (DTS), and the Data Handler Service (DHS). The primary goal of our implementation was to test the architectural concepts described in Chapter 5 on a real metro-area optical network. In this implementation, we have:

1) Implemented a suitable application-level interface to the Network Resource Service for allocating bandwidth to applications
2) Developed a resource scheduling infrastructure suitable for optical networks and other resources
3) Implemented a platform that can interoperate with Grid applications and services using OGSA/OGSI standards
4) Collected experimental data and analyze Grid data-intensive service over a metro area network

6.3.1 Network Resource Service

The implementation of the NRS provides client applications and services with an interface for advanced reservation of lightpaths with guaranteed bandwidth. The description and the architecture of the NRS were overviewed in Chapter 5. Lower-level details related to the optical network are hidden from the application.

To request a lightpath reservation, a client application specifies:

1) Two hosts requesting connection
2) Connection duration
3) Time window in which the connection can occur as specified by the starting and ending time of the window

The NRS returns a "ticket" describing the resulting reservation, if it is possible to make a reservation meeting the given requirements. This ticket includes the actual assigned start and end times, as well as the other parameters of the request. The ticket can be used in subsequent calls to change, cancel, or obtain status on the reservation. The NRS will allocate the indicated lightpath at the agreed-upon time, as long as the reservation has not been canceled or changed since it was made. Sample pseudo code for a client calling the NRS is shown in Figure 6.4.

```
// Bind to an NRS service:
NRS = lookupNRS(address);
//Request cost function evaluation
request := ≤pathEndpointOneAddress,
              pathEndpointTwoAddress,
              duration,
              startAfterDate,
              endBeforeDate≥;
ticket = NRS.requestReservation(request);
// Inspect the ticket to determine success, and to find
the currently scheduled time:
ticket.display();
```

Figure 6.4 – NRS client pseudo-code.

6.3.2 Data Transfer Service

The Data Transfer Service (DTS) is a middleware service above the NRS. It provides an interface for applications to request the transfer of named data sets from one location on the Grid to another location. The implementation source host and path name are explicitly required. With the use of a replica location service, only an abstract source data set name is required and DTS chooses an appropriate physical source location. Data transfer requests can also specify scheduling parameters such as window start and end times, and transfer duration.

DTS processes the request before calling on the NRS scheduling facility. This includes finding the appropriate data source; verifying its existence, size, and accessibility; coordinating with the destination system storage scheduler, and the like. The client's request, possibly modified by these additionally discovered constraints, is passed to the NRS scheduler, and the received job ticket is passed back to the client. A cron-like entry is made for DTS to wake up at the scheduled network allocation and data transfer time.

At the scheduled transfer time, the DTS sends a message to a Data Handler Service (DHS) running on the destination host, which then opens a connection to the source host and transfers the data. Status is sent back to the DTS and is available to client queries and for registered client callbacks.

6.3.3 Resource Allocation Agent

We have implemented a resource allocation agent implementing a Grid Network Service (GNS). Each participating network domain had one or more instances of the GNS running. In case multiple instances of GNS are running in a single domain, a master instance is elected. This GNS master instance manages the domain and the inter-domain connectivity through peer messaging.

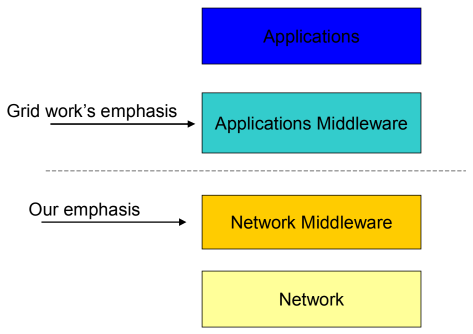


Figure 6.5 : GNS Coupling API

As depicted in Figure 6.5, GNS exposes a coupling API allowing coupling with applications. The interface to applications is bi-directional, enabling network performance and availability information to be abstracted upwards toward the application. Applications can request network services through this API. Applications can, for example, request a "cut-through" (high bandwidth, low latency) service allowing applications to bypass Layer 3 and directly transfer data over Layer 1 connections. Applications can further specify if they want this service on demand or via a time-of-day reservation. This kind of functionality is deemed especially valuable for the data-intensive applications used in research networks.

6.4 Network Service Interface Implementation Details

We have implemented a network service with a simple application-level interface. This service is able to allocate dynamically dedicated end-to-end lightpaths requested on demand. The service interface reaches the following objectives:

1) It is usable by Grid applications
2) It is network-accessible from any authorized host
3) It is standards-based and does not require any proprietary technology
4) It allows the application to specify parameters of the desired optical path that are relevant at the application level

5) It hides the details of the underlying network topology and network management protocols

6) It supports the immediate allocation of bandwidth and is easy to extend to incorporate future scheduling

7) At the highest level, the interface described here is wrapped in a Java implementation that shields the caller from the details of the application-level protocols that are needed to communicate with the service. The service itself hides various lower level network topology and management protocols from the caller.

This interface exposes two methods to the user: `allocateNetworkPath` and `deallocateNetworkPath`.

The `allocateNetworkPath` method requests that an optical path be allocated. The path allocated should meet the criteria specified in the parameter object passed to the method. These parameters include the network addresses of the hosts to be connected, the minimum and maximum acceptable bandwidths, and the maximum duration of the allocation. The `allocateNetworkPath` method returns to the caller an object containing descriptive information on the path that was allocated. This object also serves as a handle for the `deallocateNetworkPath` request. By adding parameters representing earliest and latest acceptable start times, this interface can be extended to accommodate under-constrained advance reservations

6.4.1 Prototype Grid Integration

This platform anticipates access to resources within the context of the Open Grid Services Architecture (OGSA). This platform will be interoperable with Grid applications that can utilize WS-Agreements to negotiate services to satisfy the requirements and network characteristics of high throughput file transfer.

Therefore, the first implementation was based on fully-OGSI compliant Grid Service interfaces for use by Grid applications. These interfaces were implemented using Version 3 of the Globus Toolkit [40], with SOAP and the JAX-RPC API. Additionally, the interfaces are wrapped in Java classes that effectively hide the OGSI middleware from the applications that call them. Further implementation was migrated to GT4 and WSRF.

These interfaces are intended to provide the applications that call them with the basic functionality they need, but without requiring any knowledge of the underlying networks or network control and management protocols.

6.5 Experiments and Results

The primary objective of these experiments was to demonstrate that the underlying connection-oriented end-to-end network resources can be encapsulated within a network resource service that offers both on-demand and scheduled network services to Grid applications. Here, the Lambda Data Grid architecture was deployed to demonstrate data-intensive file transfer and memory transfer services over a high-speed optical networking testbed to requesting applications. The purpose of this thesis was to show the feasibility of adding intelligence to the network middleware from a systems and architecture perspective, with minimal focus on the exact algorithms. Future research can concentrate on fine-tuning the algorithms and optimization.

6.5.1 Memory-to-Memory Transfer

To measure effective bandwidth without overhead of the file systems, caches, hard disk and driver, a memory-to-memory transfer in Chicago metro area over the OMNInet was performed. The purpose of these experiments was to isolate side effects and to provide results for the analysis section presented later in this chapter. The results are presented in Figure 6.6. The smooth curve is the average effective data transfer rate, including the startup costs of allocating the lightpaths, which was about 25 seconds. The jagged curve, averaging around 440 Mbps, is the “instantaneous” throughput, measured in 0.1-second intervals on the receiving machine. This periodically jumped to about 530 Mbps for a sustained period of about 100 seconds. For the 30 GB data transfer, the startup costs begin to be more or less fully amortized after about 230 seconds or 12.2 GB. The reason for the lower than expected maximum bandwidth and oscillations are believed to be measurement artifacts, in part. Figure 6.7 shows the results of transferring 10GB memory to memory on the Santa Clara testbed. In this experiment, RAID devices were used as end systems; a throughput of about 900Mbs memory-to-memory transfer was achieved and the measurement artifacts were eliminated by sampling at the larger interval of 1 second.

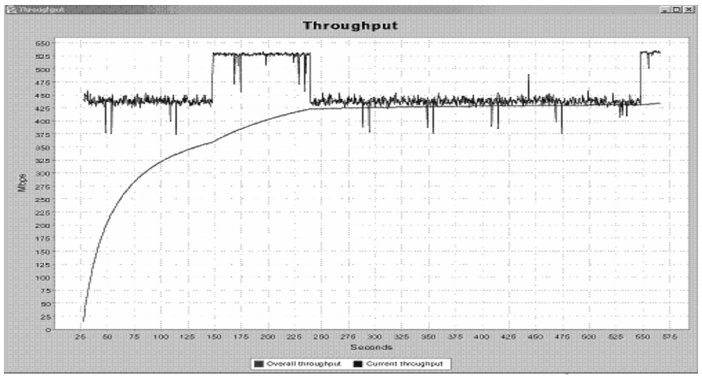


Figure 6.6–Instantaneous and average throughput for a 30 GB memory-to-memory transfer over OMNInet.

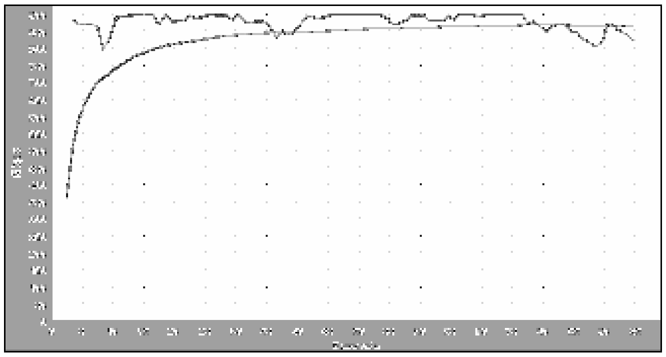


Figure 6.7 – Instantaneous and average throughput for a 10GB memory-to-memory transfer over the one-rack testbed.

6.5.2 End-to-End File Transfer (Non-optimized)

This section demonstrates that by deploying our service platform, connection-oriented end-to-end data-intensive file transfer operations can be set up and executed on-demand, easily and effectively.

The preliminary file transfer results presented in Figure 6.8 were obtained from 20GB file transfer experiments performed in Chicago metro are on the OMNInet testbed and were demonstrated during GGF-9, in Chicago, and SuperComputing, in Phoenix. The bandwidth oscillation is due to end-systems' performance, file system behaviors, and the measurement method. With simple optimization, the oscillation can be reduced. The demonstrations were run on dual PIII 997 MHz machines, running Red Hat Linux 7.3 (Kernel: 2.4.18-3smp) and using 1 GigE NIC cards.

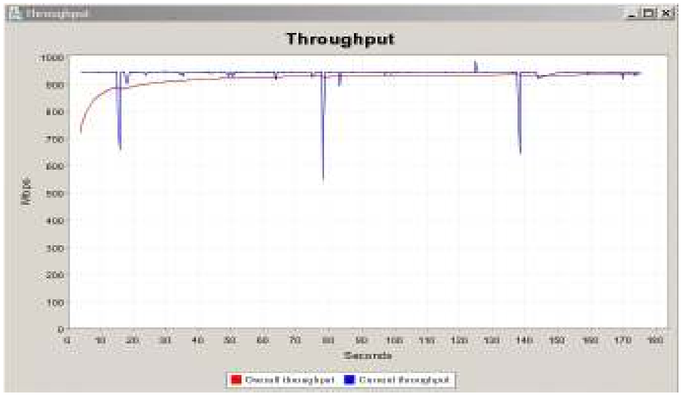


Figure 6.8 – The throughput recorded for a 20 GB file transfer. The effective file transfer rate measured over OMNInet was 920 Mbps using *ftp*.

Table 6.1 shows the breakdown of times measured for all the steps in the process. The bulk of the time was taken in the actual file transfer, but there was a significant cost of about 25 seconds to configure, and 11 seconds to tear down the lightpaths. It is important to note that only a minuscule fraction (approximately 20 milliseconds) of this delay is actually due to the setting up of the path within the optical switch. The major portion of the delay is contributed by switches at the edge of the optical network, which switch the data to and from the end hosts. This is an artifact of the current network setup. Another contributor to this setup delay is the disk access time when reading configuration files, and when writing updated switching tables to the disk.

The path tear down is 11 seconds, which is due to the centralized path update from all optical nodes. Through fine-tuning, we expect to substantially reduce the network configuration and the path release overheads. The purpose of this experiment was to demonstrate feasibility and to illustrate the range of time required to this time of preparation.

Event	Seconds
Start : File transfer request arrives	0.0
Path Allocation request	0.5
ODIN server processing	3.6
Path ID returned	0.5
Network reconfiguration	25
FTP setup time	0.14
Data transfer (20GB file)	174
Path deallocation request	0.3
Path tear down	11

Table 6.1. Breakdown of end-to-end file transfer time.

6.5.3 Scheduling

An early prototype scheduling service was also demonstrated at the SuperComputing [27] conference in Phoenix, Arizona. Several applications may request data transfers that involve the use of the same network segment for durations that may overlap. Figure 6.8 illustrates how the use of under-constrained requests and rescheduling works by considering requests for a single segment. In Figure 6.9-a, initially, a request has been made for a 70 minute block (A), sometime between 4:00 pm and 8:10 pm. Since there were no previous requests, it was granted for the first 70 minutes in this time window.

In Figure 6.9-b, a second request (B) was made. The resource requested was in the 4:15 to 7:00 pm range, for 105 minutes. In order to meet B's request, the Scheduler placed it at the beginning of its time window and moved A, the first request, to immediately follow the second request's completion. The first request, A, still satisfied its under-constrained window.

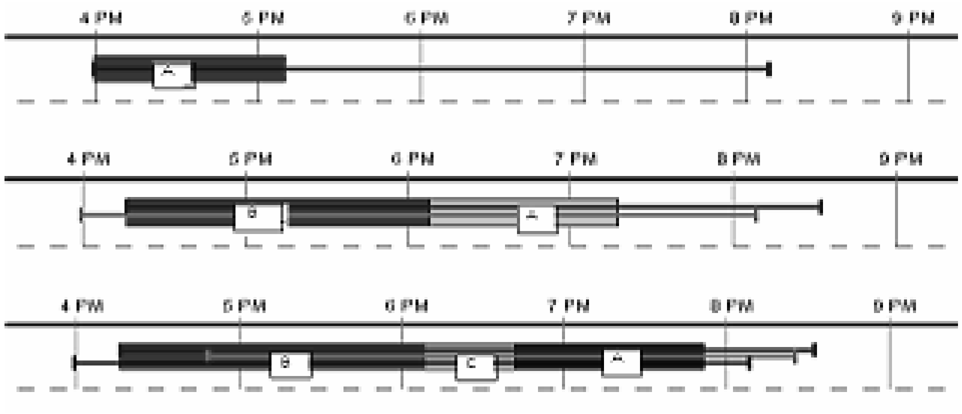


Figure 6.9a, 6.9b, 6.9c – Behavior of the scheduling algorithm as three successive requests are made to use one segment.

In Figure 6.9c, a third request (C) was made between 4:45 and 7:00 pm, for 30 minutes. Again, request A was rescheduled to accommodate C, and B was left at its original allocated time.

The results demonstrated that with the network resource service, application requests can be scheduled and rescheduled to make efficient use of network resources.

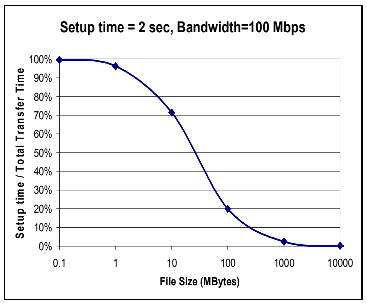
6.6 Analysis of Lambda Scheduling and Dynamic Allocation

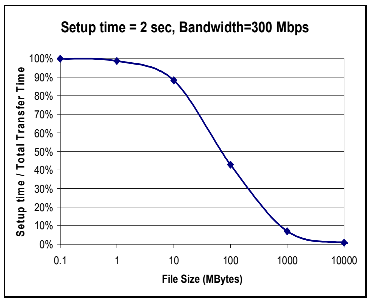
Rapid-prototyping was developed to demonstrate the proof-of-concept, whereby a high-speed optical networking testbed was able to provide on-demand lightpath allocation to requesting applications. The objective was not to achieve bit blasting, but to demonstrate the availability and efficacy of a bandwidth rich network resource, especially in the case of applications requiring bulk-data transfers. The technology does come, however, with its own processing overhead and is not meant to be a one-size-fits-all solution. Several experiments were performed on data sets of varied sizes to ascertain the threshold at which the benefits of using this network infrastructure are evident. It would be overkill for transaction oriented, low volume traffic. However, at any transfers of more than a few GB, the throughput gains are enormous. Relatively low setup times were recorded, even when using off-the-shelf software and commodity storage components.

In the past, circuit switching has been criticized for its long path setup time. In spite of the time taken for the setup, circuit switching, and optical networks for that matter, are the perfect solution for non-transaction oriented Grid computing. In reality, the time taken in the transference of the huge amounts of data generated by Grid applications amortizes the time spent in the setup. The graphs in Figures 6.10, 6.11, and 6.12 illustrate this fact. Each graph shows the amount of time spent in the path setup as a percentage of the total transfer time versus data transfers of different sizes. These numbers were used to motivate the usage of optical networks in different scenarios. In each graph, the knee of the curve indicates the point in which the setup time, i.e., the overhead, is amortized and becomes insignificant.

In Figure 6.10, we assume a setup time of 2s and a bandwidth of 100Mbps. In this case, the knee of the curve is at 1GB. In Figure 6.11, we assume a setup time of 2s and a bandwidth of 300Mbps. In this case, the knee of the curve is at 5GB. In Figure 6.12, we assume a setup time of 48s and a bandwidth of 920Mbps. In this case, the knee of the curve is at 500GB. Since Grid applications usually transfer terabytes or even PetaBytes of data, it is clear that the time for the path setup is negligible in Grid computing.

The 48-second and 2-second path setup time are current practical upper bounds and desired lower bounds. Of equal importance is the time required to deallocate resources to enable subsequent usage. Again, this does not prove to create any major bottleneck and has been observed to be around 11s, as discussed in Chapter 5.





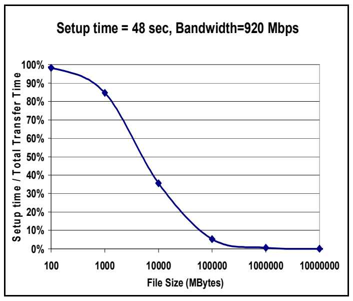


Fig 6.10: Overhead is insignificant at 1 GB

GB

Fig 6.11: Overhead is insignificant at 5

00 GB

Fig 6.12: Overhead is insignificant at

Figures 6.13 and 6.14 compare the efficiency of packet switching with Lambda switching, executing with different amounts of bandwidth, by plotting the amount of data transferred against the time taken in seconds.

These two graphs were generated synthetically. In Figure 6.13, the path setup takes 48 seconds while, in Figure 6.14, the path setup takes 2 seconds. The path setup time includes the time required to request a path, allocate the path, configure the layer 1 and 2 components of the network and persist this information in the relevant tables. The 48 sec was observed average setup time

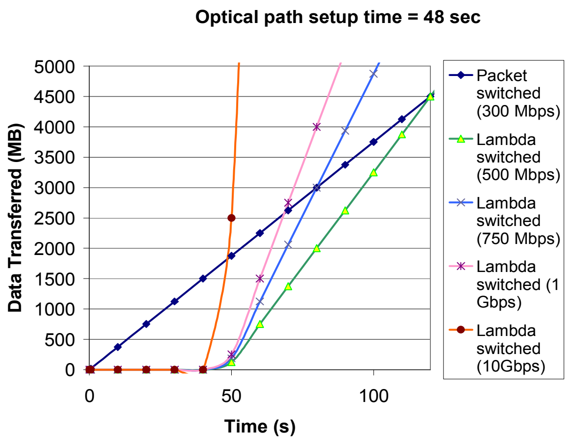


Figure 6.13: Packed Switched vs. Lambda Network -- Setup time tradeoffs.

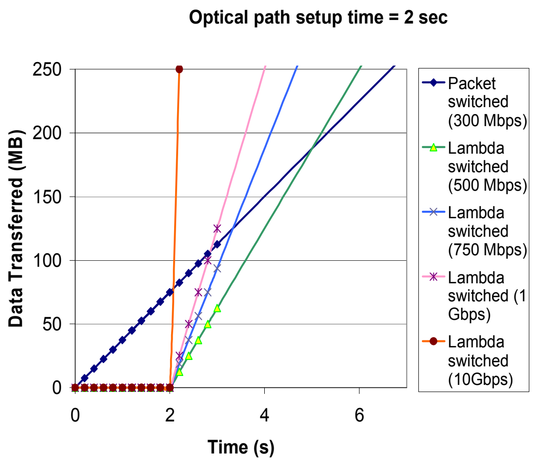


Figure 6.14: Packed Switched vs. Lambda Network -- Setup time tradeoffs.

The important information obtained by these graphs is the crossover points at which the optical network becomes a better option than the packet switched network. Note that when the path setup takes 48 sec, the crossover point can be at files as small as 2.5GB up to 4.5GB. When the path setup takes 2 sec, the crossover takes place between 75MB and 200 MB. These graphs affirm the enormous advantage of the DWDM-RAM system over other packet switching systems when transferring large data sets, and present an exciting option for the Grid community.

Figure 6.15 compares the time to transfer data using OMNInet and the public Internet. The times shown in this graph represent the time measured to transfer data sets of different sizes over the optical network and the public Internet. Note that the optical network does a better job even for files as small as 1GB. The graph shows that the difference increases significantly as the file gets larger, and that the larger the file the greater the benefits provided by the optical networks.

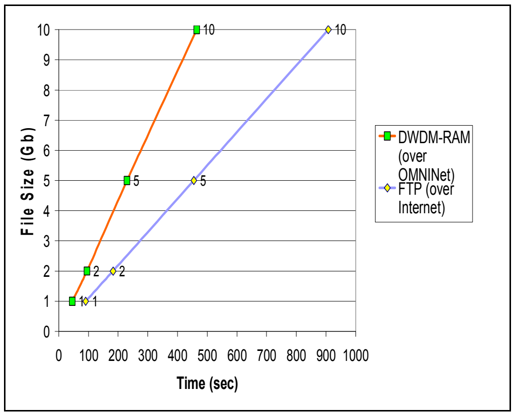


Figure 6.15: File transfer time.

Note that, for transferring smaller amounts of data, the traditional packet switching approach is a better option. The appropriate network to use, in each case, should be decided by co-allocation services, as defined in NMI. These co-allocation scheduling services use information about the resource's capabilities and the application's requirements to allocate a performance-efficient set of resources for the application.

It should be emphasized that the focus of the experiment is not on exploring various schemes for maximizing the data transfer throughput, rather it focuses on encapsulating the underlying network resources and offering network scheduled services according to an application request. For this reason, only the standard TCP based networking protocol stack readily available in the Linux kernel was used. The OMNInet testbed network guarantees high bandwidth channels since the network setup allocates a Lambda of 10Gbs to each data flow. In spite of the network supporting this very high bandwidth, it was observed that the end-to-end data flow between hosts did not fully utilize this bandwidth.

As the OMNInet is inherently reliable with extremely low bit error rates and fairness is not an issue (dedicated lambda per flow), it is an overkill using the reliable TCP as the transport protocol for such data-intensive applications. Rather, a performance tuned transport protocol like FAST [69] tailored to benefit from a circuit switching path with no L3 routing, would be more effective in achieving high throughput over the 10Gbps pipe.

We observed that consistent oscillations in the instantaneous throughput severely affected the aggregate throughput. Two possible areas could be further explored to yield better performance. One of these areas is the transport protocol used. The standard TCP based networking protocol stack available in the Linux kernel has been used. The OMNInet testbed network is inherently reliable and guarantees high available bandwidth and low bit error rates. Sharing of channels is not an issue, so fairness concerns do not apply. As such, the reliability-oriented functionality for the data transfer is unnecessary baggage. Moreover, the network setup allocates Lambdas of 10Gbs. In spite of the network supporting this very high bandwidth, it was observed that the end-to-end data flow between hosts does not fully utilize this bandwidth. A performance tuned transport protocol like FAST [69] or SABUL/UDT [47], tailored to benefit from a circuit switching path with no L3 routing, would help to fill the 10Gbps pipe.

Another candidate for optimization is the peripheral equipment (on the hosts), which is not tuned to handle extremely high data rates. For instance, the access to secondary storage is clearly a bottleneck. It was observed that striping data to multiple secondary storage media, deployed in a RAID configuration, helps to gain much better performance. Experiments on our Santa Clara testbed with a RAID system, with the source and sink machines connected directly with a 1 GE link, gave us 350 Mbps throughput. This was achievable on two Pentium IV 2.8 GHz machines, each with a 1 Gbps copper interface, connected by using the Reiser file systems on source and sink RAID machines, as well as disabling hyper-threading in the Linux kernel. However, this was still far from utilizing the full 10Gbs of available bandwidth. With a memory-to-memory direct transfer on this same hardware, better results were achieved.

6.7 Network Scheduling

A Grid Scheduled Network Service is just like any other Grid service; it has to expose its service through a Grid Service Port Type. For example, an interface for requesting lightpath connectivity should be rich enough to allow applications to express their own flexibility via under-constrained (or loose-constrained) requests. This allows for optimal scheduling, and for automatic rescheduling if necessary. The scheduling network service considers the flexibility in the requests, the flexibility inherent in any conflicting current reservations, and other factors such as job priorities or predictive load balancing. It provides guarantees or advance reservations for channel availability between specific endpoints, with certain constraints. The reservations may be periodic, and may be many days in advance. They may be policed by various resource reclamation policies, such as periodic application resiliency requirements. When rescheduled within the bounds of the under-constrained request to meet new requests, these changes are reported via a middleware query or notification mechanism.

6.7.1 A New Provisioning Model

Grid users are accustomed to allocating and relinquishing some virtualized sets of computational, storage, and/or visualization resources. They do so with a high degree of automation, using software feedback loops and schedulers taking the place of GUI portals and operators.

In many Grid scenarios, the network element turns out to be a resource as important as computation and/or storage. As such, Grid users require the same level of control towards subsets of well-defined amounts of network resources for the duration of a specific Grid task.

A chief goal of this service plane is to turn the network into a virtualized resource that can be acted upon and controlled by other layers of software, for example applications or Grid infrastructures. In other words, the network becomes a Grid managed resource much as computation, storage, and visualization are layered upon the optical network control plane

The service plane is typically concerned with path allocation, optimization, monitoring, and restoration across two or more domains. A service plane must be designed to be extensible from the ground up. It should allow adaptation of various control plane interfaces and abstract their network view, or element set, into the service plane. Examples of underlying control planes are: ASTN, GMPLS, JIT, etc. Each control domain has exclusive control of its resource and is typically able to signal neighboring domains to create end-to-end paths on behalf of the involved Service Providers.

The Grid Network Services agent is a key ingredient of the service plane. A Grid Network Service (GNS) agent advertises network capabilities to its neighbors. As such, each agent is able to generate a complete topology view in order to construct future optical end-to-end paths. Our agent can optionally support source based routing to select a preferred path through the individual optical clouds.

6.7.2 Grid Network Service Agent

We implemented the Grid Network Service agent as part of multiple instances of Lambda Data Grid (LDG). The Grid Network Service agent was implemented as an extension of the basic NRS presented in chapter 5. When multiple agents are running in a single domain, a master instance is elected. This LDG master instance manages the domain and the inter-domain connectivity through peer messaging. LDG core framework includes services like a policy engine, a topology discovery engine, workflow utilities, inter-domain routing facilities and smart bandwidth management fixtures.

LDG exposes an API allowing coupling with applications. The interface to applications is bi-directional, enabling network performance and availability information to be abstracted upwards toward the application. Applications can request network services through this API. Applications can for example request a "cut-through" (high bandwidth, low latency) service allowing applications to bypass Layer 3 and directly transfer data over Layer 1 connections. Applications can further specify if they want this service on demand or via a time-of-day reservation. This kind of functionality is deemed especially valuable for the data-intensive applications used in research networks.

6.8 Applications Drive Secure Lightpath Creation across Heterogeneous Domains

In this section, we describe how the above components were integrated into a system-wide platform preformed on transatlantic model between Amsterdam and Chicago [37]. In Fig. 6.16 the testbed is depicted showing the three optical networks, each considered as a single administrative domain.

This demonstration realized an open, programmable paradigm for application-driven network control by way of a novel network plane the, “service plane,” layered above legacy networks. The service plane bridges domains, and exposes control to credited users/applications, and is part of the Lambda Data Grid. We have experimented with such service plane in an optical, large-scale testbed featuring two hubs (NetherLight in Amsterdam, StarLight [62] in Chicago) and attached network clouds, each representing an independent domain. The dynamic interconnection of the heterogeneous domains occurred at Layer 1. The interconnections ultimately resulted in an optical end-to-end path (lightpath) for use by the requesting Grid application. This experiment was done in collaboration with University of Amsterdam, this section focuses on my contribution and it relevancy to this thesis.

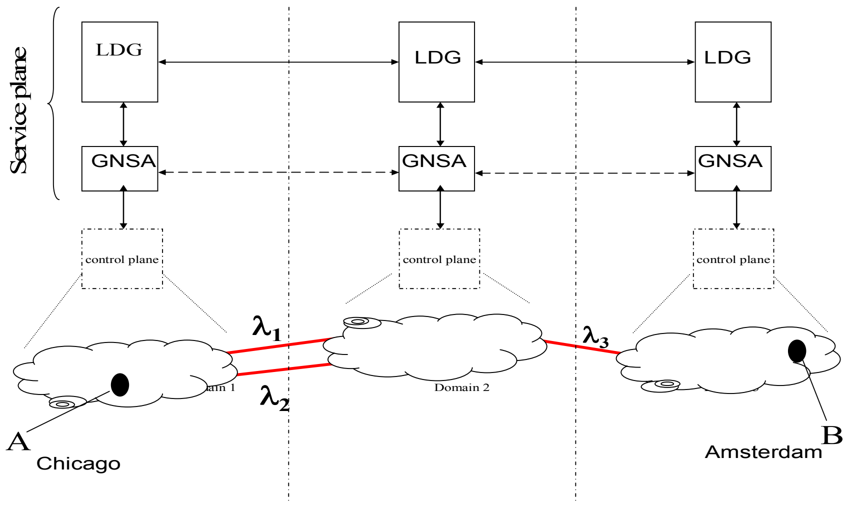


Figure 6.16: Transatlantic Lambda failover architecture.

In Grid networks, users and applications need to gain greater control of network resources for them to exploit their atypical traffic patterns and meet their throughput/latency requirements. A domain is an independently managed network cloud exposing a set of ingress and egress points associated with Service Specifications. Provisioning an optical end-to-end path while crossing different domains is quite a challenge [2]. The optical control planes may differ among the multiple domains. It becomes crucial to establish common syntax and semantics for accessing network resources. This section presents a provisioning architecture that has the ability to integrate different network allocation approaches, as well as different styles of control over network resources. We reduce this architecture to practice via a "service plane" — a new software layer that resides on top of control planes such as ASTN, GMPLS, and JIT. The service plane uses Grid Network Services (GNS) software agents. These agents allow users and applications on their behalf, to negotiate on-demand network services such as low latency connection, high throughput transport, network knowledge services, and third party services.

6.9 Transatlantic Lambda Failover

The purpose of this experiment was to show Lambda failover between Amsterdam and Chicago. We demonstrated this provisioning model during the Super Computing Conference [27] held in Pittsburgh, PA, USA. Three optical domains, NetherLight, StarLight, and OMNINet were part of the experimental testbed, representing Domain 1, 2 and 3 respectively in Figure 6.16. NetherLight is the optical infrastructure in Amsterdam and the University of Illinois at Chicago manages StarLight. The inter-domain lambdas are a collection of optical entities necessary to establish a connection between peer networks. For instance, the connection between NetherLight and StarLight is a collection of SONET based optical Exchange Points [9] and transit provider links for long distances.

In each domain, a single agent was given responsibility for the setup of intra-domain connections. It also connects ingress points to egress points in the domain under its control. In this testbed, we simulated a link failure by switching off a port on one of switches providing inter-domain connectivity, thus generating an inter-domain failure event. We then measured end-to-end link restoration times across the three domains shown in Fig. 6.16. The elapsed time for both detection and restoration of the inter-domain failure was in the order of a minute. Although there are several ways to optimize such times, this is already a vast improvement of the usual hours to days required to restore a connection using conventional means such as phone or email.

6.9.1 Experiment Analysis

The Grid middleware was rebuilt for resource allocation and automatic failover of Lambdas between Amsterdam and Chicago, via NY and Canada, over a distance of about 10,000km. This experiment measured the reservation and allocation of lightpath in about 100 seconds, compared to the manual allocation of about 100 days. Unlike a commercial model with thousands potential links, this experiment used two Lambdas with much less complexity. This initial experiment demonstrated the potential to automate the initial concept into larger experiments with automatic provisioning and failover.

The computational middleware was able to reduce the allocation time from months to seconds, allowing integration of these interfaces to Grid Computing applications and e-Science Workflows. Shifting from manual allocation to an automated computational reservation system and allocation via the Grid Web Services model opens the door for significant advancements in scientific research.

With a novel provisioning architecture built upon a service plane, we have shown that it is possible to perform programmable, application-driven network control in an open fashion. The model has a minimal reliance on adjacent clouds implementing the same control plane standards. We have latched our open approach onto the software evolution curve (Web Services, Service Oriented Architecture) to best exploit dynamic service discovery and feature introspection across domains, without extensive code rewrites. Through a layered approach, we have shown how restoration can be fine-tuned to occur at the different places — i.e. the control plane for intra-domain failures and the service plane for inter-domain failures.

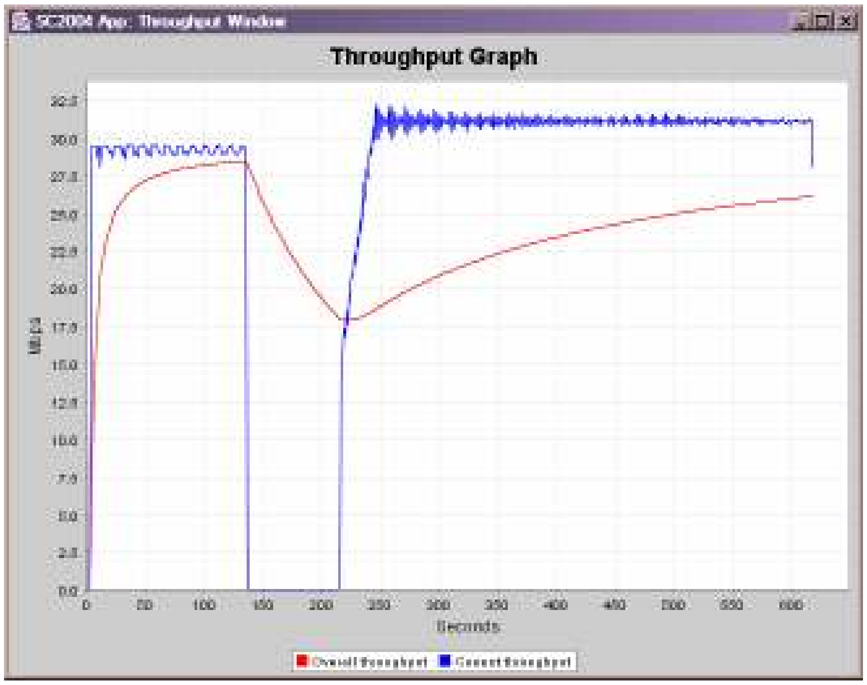


Fig. 6.17: The current throughput line (top) shows the time interval at [140, 210] seconds that is required for the service plane to detect and recover the simulated inter-domain failure. The bottom line represents the overall throughput.

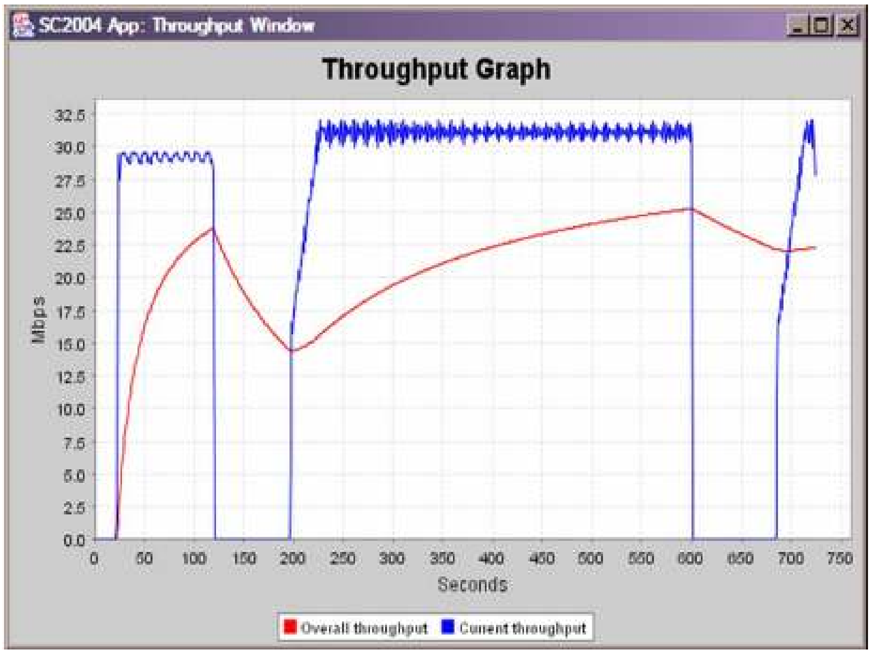


Fig. 6.18: Similar to the above setup with the addition of getting back to the original Lambda from Amsterdam to Chicago.

6.10 Conclusion

In this chapter, we presented the implementation of Lambda Data Grid and some measurements and analysis. Setting lightpath takes a longer time them traditional routing. The measurements showed that it may take longer in orders of magnitude in setup time; however, for large files the importance of time for setup is reduced in relation to overall efficacy.

The demonstrations and prototype systems described in this chapter have shown the value of provisioning a dynamic network through a Grid service platform. The system presented here overcomes limitations of traditional implementations and facilitates on demand optical channel (lightpath) provisioning. The Grid service implementation is OGSA compliant and is implemented using OGSI interface developed to ease the usage of the OMNInet by Grid applications. The service interfaces presented meets the needs of many user applications, and could be extended to include other features required by scientific applications. What follows is the introduction of Time Value and Time Window as part of network scheduling, an essential component in building scientific Cyber-infrastructure.

7 Scheduling – Time Window and Time Value

In the previous chapter, we discussed the testbed, the experiments and analyzed some of the results. In this chapter, we will present scheduling service in terms of a multi-dimensional model. No longer is it satisfactory to use computation job scheduling or prioritization of resources and availability. The number of necessary services and the interactions between them complicates scheduling. Orchestration must take place in the scheduling of other services including computation, storage, visualization, and unique sensors, e.g. the telescope array field in the Netherlands.

Constant availability is a present day Internet assumption. In reference to massive amounts of data comprising e-Science research, this assumption is not valid. Unlike the constant connectivity of the pubic Internet, scientists build dedicated small virtual topology for each experiment. The topology consists of a small span point-to-point topology, built for one experiment alone, used for the duration of the experiment, and is torn down at its conclusion. A typical experiment consists of three to seven connections between scientific equipment or institutes. As such, there are new challenges that arise. In this chapter, we will discuss these challenges along with proposed solutions.

The following is an introduction of two new concepts: Time Window and Time Value. Connectivity is required in a specific time window and for a specified duration, accomplished as part of the Scientific Workflow in conjunction with the availability of other resources. Further, there is an inherent value associated with the connectivity based on time. For example, there would be no value in having a dedicated Lambda from Osaka's massive microscope if the computation at UCSD were not available, and/or if the visualization at Chicago's EVL were not ready for presentation.

In this chapter, we introduce a new concept devised to build a system of cooperation between the network middleware, the scientific middleware, and the Grid middleware.

7.1 Scheduled Connectivity Service

Network elements such as routers/switches, end devices, and physical links are essential for creating connections between end- users. A Grid network infrastructure is essentially an overlay network over physical networks for connecting end systems belonging to a Virtual Organization. As such, connectivity between end systems is an essential resource that glues the infrastructure together.

With packet switched networks, connectivity is assumed always available under best-effort service and statistical multiplexing. A connection is never denied, but quality of the connection degrades progressively depending on the traffic conditions at the time of the connection. With most circuit-switched telecommunications networks, a connection can be denied or blocked if its QoS requirements are not met or network resources are not available.

To provide a degree of stability and predictability for Grid applications, connectivity should be treated as a scheduled service, a Grid Scheduled Connectivity Service. With Grid Scheduled Connectivity Service, Grid applications can utilize a WS-Agreement service to negotiate connectivity resources that satisfy their QoS requirements before their actual deployment. Furthermore, the Grid Scheduled Connectivity Service allows network resources to be utilized flexibly and efficiently.

Connectivity services can be classified into several types depending on the specific type of service provisioning: DS-MPLS, pure DiffServ, and Lightpath provisioning, to name a few. In a dynamic optical network environment, connectivity between end systems can often be established by concatenating lightpath segments between two endpoints. In a DiffServ network, connectivity can be established by concatenating logical hops of the same class or DS codepoint.

A Grid Scheduled Connectivity Service is just like any other Grid service; it has to expose its service through a Grid interface. For example, an interface for requesting lightpath connectivity should be rich enough to allow applications to express their own flexibility via under-constrained or loose-constrained requests. This allows for optimal scheduling, and for automatic rescheduling if necessary. The scheduling connectivity service considers the flexibility in the requests, the flexibility inherent in any conflicting current reservations, and other factors such as job priorities or predictive load balancing. It provides guarantees or advance reservations for channel availability between specific endpoints, with certain constraints. The reservations may be periodic and may be many days in advance. They may be policed by various resource reclamation policies, such as periodic application resiliency requirements. When rescheduled within the bounds of the under-constrained request to meet new requests, these changes are reported via a middleware query or notification mechanism.

Once a WS-Agreement service is created for the connectivity resource, higher level scheduled services such as data transfer, storage, computation, instrumentation and visualization can be activated to support an application.

As an example of use of the Grid Scheduled Connectivity service, a File Transport service can be scheduled between end systems by deploying a scheduled storage service and a Grid connectivity service. Often data-intensive applications require the transfer of a massive amount of data sustained over a considerable period. For this reason, a Grid Scheduled Connectivity service is essential, and some form of high throughput transport protocol is required. GridFTP is often used to provide high-throughput file transport. Alternatively, an optimized transport protocol like SABUL/UDT [47] can also be used.

7.2 Scheduling Service Scenario

Consider an environment in which a client requests a certain large file to be transferred to its site under certain constraints related to the actual time of the transfer operation. For example, a High Energy Physics group may wish to move a 100TB data block from a particular run or set of events at an accelerator facility to its local or remote computational machine farm for extensive analysis. An application client issues requests related to data named in an abstract namespace. In this model, a client may be associated with a unique data store node on the network, and the request is to obtain a copy of the data to that node. It is important to ask, "Is the copy the 'best' copy?" The determination of "best" depends on, a) the content, b) the network connectivity/availability, and c) the location in the context of the process. The client issues the request and receives a ticket in response describing the resultant scheduling, including a method for modifying and monitoring the scheduled job. The client does not know, or care, about the actual source of the data, which may come from any of the nodes of the network; indeed, the source might be any one of a number of replicas of the original data file, chosen by the Data Transfer Scheduling Service, in interaction with other Grid services. The replica can be constructed out of data slices on several sources similar to some of the P2P concepts.

Client requests include scheduling specifications. A typical scheduling specification is, "Copy data X to the local store, on machine Y, after 1:00 and before 3:00." At the application level, the Data Transfer Scheduler Service creates a tentative plan for data transfers that satisfies multiple requests over multiple network resources distributed at various sites, all within one Administrative Domain. The scheduled plan is formed based on knowledge of the requirements of the requests, the existence of data and its size, its locations and availability. At the middleware level, a network resource schedule is formed based on the understanding of the dynamic lightpath provisioning capability of the underlying network and its topology and connectivity. Co-reservation of network resources occurs at this level. At the resource provisioning level, the actual physical optical network resources are provisioned and allocated at the appropriate time for a transfer operation. Finally, a Data Handler Service on the receiving node is contacted to initiate the transfer. At the end of the data transfer process, the network resources are deallocated and returned to the pool.

In the above scenario, there is a variety of procedural possibilities: Network resources may be unavailable, better plans can be scheduled, a better agreement can be negotiated, and a firm schedule will eventually be executed within the predetermined time window. Job requests that cannot be accommodated within the context of a current schedule may result in callbacks to clients asking them to reschedule their allocated jobs to better satisfy all current requests, or to satisfy newer, higher priority requests. In all, the system tries to satisfy all data transfers and network bandwidth requests, while optimizing the network usage and minimizing resource conflicts.

7.3 Time Window Example

We developed a time window model that allows facilitation of scheduling under-constrained requests. Figure 7.1 illustrates how the use of under-constrained requests and rescheduling works by considering requests for a single segment. In Figure 7.1a, a request has been made for a 30-minute block (W), sometime between 4:00pm and 5:30pm. Since there were no previous requests, it was granted for the first 30 minutes in this time window.

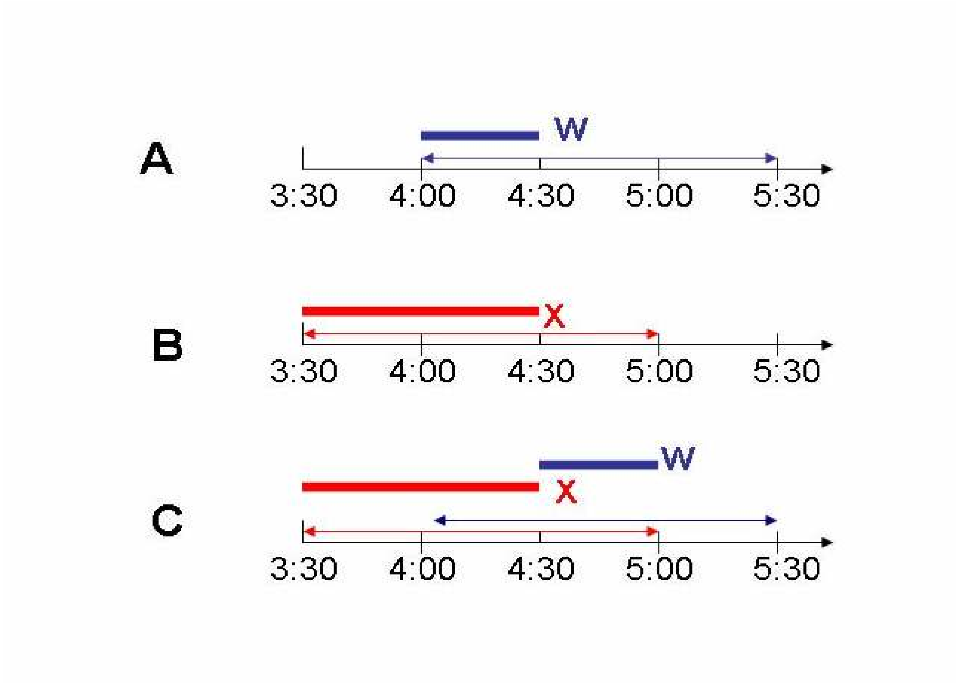


Figure 7.1a, 7.1b, 7.1c – Behavior of the scheduling algorithm as three successive requests are made to use one segment.

In Figure 7.1b, a second request (X) was made. It was in the 3:30 to 5:00pm range, for 60-minute slot. The time-window to satisfy the (X) does not meat (X) constrains if placed after the end of (W), this is in addition to waste of resources for the time between 3:30 and 4:00.

In Figure 7.1c, the scheduler needs to shift the request to the beginning of the time window (3:30) and move (W), the first request, to immediately follow the second request's completion. Both requests (W) and (X) are satisfied their under-constrained windows.

The model presented above is a simplified view of time window and duration. An enhancement of this model was done by DTS requesting the data size instead of the time; e.g the request is to transfer a terabyte of data between 4pm and 8pm. This can be done in time X if we allocate one Lambda, or in X/2 if DTS allocates two Lambdas. This DTS enhancement gives the scheduler more flexibility and another dimension for the allocation of dynamic durations instead of static time windows.

During the waiting time for the data transfer, DTS keeps the Handler for callbacks from NRS for potential new proposed schedules within the time window. During each of the new proposed schedules, DTS, DHS and NRS exchanged information to synchronize the right sliding window for the right data transfer. The window is a floating window that could change based on the availability of managed service and in contrast to the always-available transmission concept.

7.4 Time-Value and Time-Value Curves

When looking at scheduling of network services it is important to look at the value of scheduling as a time function. With the shift from network on demand to scheduling, we introduce Time-Value and time value curves as functions to measure the total value of a network service at a specific time. The value of the data over time to enable non-functional requirements is to be taken into account when ranking service offerings to select which of several service offerings should be used to define the best value for the specific scientific experiment. For example, the value of a specific experiment may be extremely important and the value of the data might be extremely high if it is able to be executed within the next 24 hours. After that time, the value of the data may be of significantly less value, since other events that could benefit from completion of the experiment may, by that time, have already occurred. The value of the information is different at different stages.

The time-value information may be implemented as time-value curves that which may allow the value of the data process to increase over time, decrease over time, remain constant over time, or otherwise change over time. For example, we can look the time-value curves in Figure 7.2. The workflow may specify that the value of the network increase for a given period, plateaus for a given period, and then decreases over time. Figure 7.2 illustrates several graphs of different time-value curves that may be implemented. Possible time value curves, including:

- a linear increase in value over time
- an asymptotic increase in value over time
- a linear decrease in value over time
- an asymptotic decrease in value over time
- a level value over time
- a plateau shaped curve in which the value increases linearly, remains constant, and then decreases linearly
- a step value curve in which the information value is initially at one value and then drops to a second value at a particular point in time

Other time/value curves may be used as well and the invention is not limited to an implementation in which one of these particular illustrated time/value curves is used to specify and/or adjust the priority of an experiment over time.

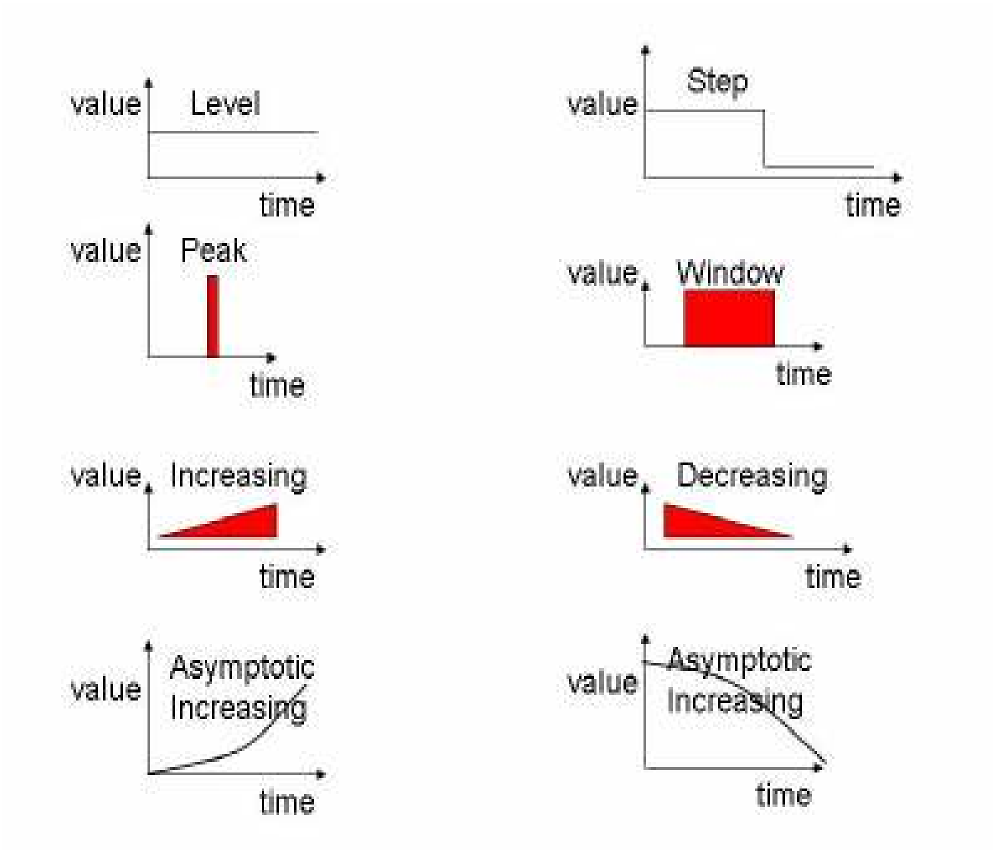


Figure 7.2 - Time-Value Curves.

7.5 Time-Value Curve Evaluation and Data Stream Transfer Adaptivity

This work describes an architecture that makes use of an effective but relatively simple model for calculation of priorities from time-value curves. These were then utilized to determine data stream transfer decisions involving transfer initiation. We present some alternate advanced mechanisms for implementing such adaptivity which can augment the versatility of advanced scheduling tools.

- **A time-value curve library** can include a variety of predefined curves suited for particular task requirements. Curves would support non-linear as well as linear piecewise components.

- **Weighted task system** allows changing field circumstances to be fed back to the scientific workflow in the form of weights to be applied to the calculated priorities. This allows scientists to have input based on experiment results, for example.

- **Integrated models for calculating value: past, current, and predicted future**. Rather than taking a point value in time for the time-value curve calculation, the total value through the current time can be used for the calculation.

- **Statistical prediction -** Integrated models for calculating values: past, current, and predicted future could utilize statistical information from previous similar missions for time-value curve calculation and decision making.

- **Adaptation rezoning – sampling interval, window size, sliding window**. These are modifications and refinements to calculation of the priority from an integral of time-value curve up to the present time.

- **Experiment can have "Importance Level"**. This is similar to a weighting system for the time-value curves.

- **Black box Time-Value-Priority curve functions**. This allows "plug-and-play" functions of arbitrary complexity and assignment to be utilized depending on experiment requirements. These could be deployed at any time during the computation to support changing experiment requirements.

- **Predicted future value** can be used for intelligent scheduling and also in the absence of sufficient current information for decision making.

7.6 Scheduling Service Architecture

The Scheduling Service Architecture is a realization of the general architecture and was demonstrated at Super Computing in Pittsburg. Figure 7.3 illustrates the architecture. Applications can interact directly with either the Network Resource Service (NRS) or the Data Transfer Service (DTS) or both, as well as with other Grid services. Important to note is the separation of network services from file transfer services. The latter depends on the former, but network services may be requested on-demand or via advance scheduling, independent of any file transfer request.

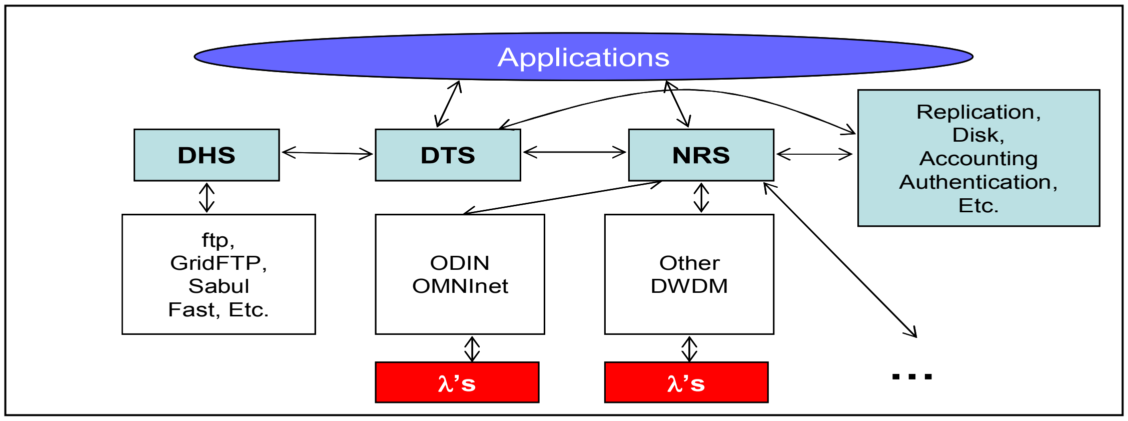

Applications
DHS
DTS
NRS
Replication, Disk, Accounting Authentication, Etc.
ftp, GridFTP, Sabul Fast, Etc.
ODIN OMNInet
Other DWDM
λ's
λ's
...

Figure 7.3 – The Scheduling Service Architecture.

A request for network bandwidth to NRS may be satisfied by its own network scheduling and routing modules. Both end-to-end and segment-by-segment requests are allowed. NRS decides which underlying networks to use to satisfy a request. In the OMNInet [55] testbed described in Chapter 7, resources are controlled by ODIN [55] software developed under an NSF grant at Northwestern University. Other networks might also be available and NRS can hide their implementation details from upper layers and present a uniform interface to the application layer. A request is authenticated in terms of security, and may be integrated with a policy server. Then the source data is verified according to certain criteria: Does the data exist? Is it readable to this user? How big is it? Data set size determines how long the transfer operation should take, given expected network speeds over the segments chosen for the transfer, as well as IO capabilities of the end-point machines. At the scheduled time of a data transfer, the NRS allocates the segment-by-segment path. The DTS then sends a request to the DHS running on the destination machine. When the transfer is complete, DHS informs the DTS, which then tells the NRS to deallocate the path and return those resources to the pool available to service other requests. Results of the prototype implementation were presented at Super Computing at GHPN.

7.7 Design of the Network Resource Service

There are three primary goals for the design of the Network Resource Service: 1) to provide a high-level interface for applications and services to allocate bandwidth; 2) to provide transparent interfaces to a variety of underlying network control mechanisms; and 3) to provide reservation and scheduling functionality.

A usable application-level interface is essential to making dynamically provisioned high-bandwidth optical network resources available to a wide variety of applications, while hiding the complexities of the underlying network and network control mechanisms. Unlike packet switched networks, a dynamically provisioned wavelength is dedicated to only one user at a time. Thus, efficient utilization of dynamic wavelength provisioning requires scheduling, both to optimize the utilization of the underlying resources, and to provide applications with predictable and reliable bandwidth.

7.7.1 Scheduling Services

Although the Network Resource Service supports both on-demand and scheduled allocations, scheduling is perhaps its most significant feature, and scheduling is an integral part of the design.

The interfaces for requesting bandwidth should be rich enough to allow clients to express their own flexibility via under-constrained requests. This allows for optimal scheduling, and for automatic rescheduling as well. The network scheduling authority considers the flexibility in the requests, the flexibility inherent in any conflicting current reservations, and other factors such as job priorities or predictive load balancing. It provides guarantees or reservations for channel availability between specific endpoints, with certain constraints. These reservations are controlled and accessed through tickets, which can be shared between applications many days in advance. They may be regulated by various resource reclamation policies. The reservations may be periodic, following periodic application resiliency requirements. When rescheduled within the bounds of the under-constrained request to meet new requests, these changes are reported via a client query or notification mechanism.

Middleware services, such as data transfer services, must coordinate their activities very precisely with network allocations. Therefore, the underlying scheduler used as the network authority can also be used to drive schedules exposed by other services. These services may add their own request language and do initial pre-processing of the request before translation to the network layer. For example, a data transfer service may receive requests that specify data in some abstract fashion. The data transfer service may consult a replica location service to determine the actual endpoints, and it computes the length of the requested network reservation based on the requested file size.

For network scheduling purposes, each lightpath on each network segment is viewed as a sharable resource that can be dedicated for only one use at a time. The scheduler can also link these lightpath segments into scheduled end-to-end lightpaths. Generally the bandwidth provided by these end-to-end lightpaths is of interest to applications, so the interface hides the underlying segment-level scheduling from callers that do not need it.

The scheduling of network paths requires coordination of multiple resources with heterogeneous constraints. For example, different segments may lie in different domains of administrative control. Some segments may have constraints such as limited windows of availability or high costs. When multiple paths are possible, the choice of segments allows a further area of optimization. All of this may be reconsidered as the system attempts rescheduling to satisfy new requests. The extension of the scheduling problem to higher-level services such as data transfer introduces additional constraints for optimization.

Some application services that use the network scheduler may allow additional avenues for optimization. To take advantage of this, the network scheduler allows clients to request that they be included in a voluntary rescheduling protocol. Via this protocol, a client is requested to consider rescheduling a previously granted reservation. For example, a data transfer service may be able to source data from a different location to free up a network segment that is needed by some other service during the currently reserved time.

7.8 Prototype Design

We designed and implemented prototypes of the Network Resource Service (NRS), the Data Transfer Service (DTS), and the Data Handler Service (DHS). The primary goal for this implementation was to test the architectural concepts described above on a real metro-area optical network. In particular, we wanted to 1) implement a suitable application-level interface to the Network Resource Service for allocating bandwidth to applications; 2) develop a resource scheduling infrastructure suitable for optical networks and other resources; 3) provide for interoperability with Grid applications and services using emerging standards; and 4) collect meaningful test data using the prototype implementation on a real network.

7.8.1 Network Resource Service Interface and Functionality

The design of the NRS presents client applications and services with a Java interface that represents advanced reservation requests in application-level terms. Lower-level details including the underlying optical network control and management protocols, the network topology, and the OGSI / Web Services interface are hidden from the application.

To request a bandwidth reservation, a client application simply specifies the two hosts it wants to connect, the duration of the connection, and the time window in which the connection can occur, specified by the starting and ending time of the window. As an enhancement to this approach, the DTS takes the data size and optimize the duration and bandwidth required for transferring this data size.

The NRS returns a “ticket” describing the resulting reservation, meeting the given requirements. This ticket includes the actual assigned start and end times, as well as the other parameters of the request. The ticket can be used in subsequent calls to change, cancel, or obtain status on the reservation. The NRS will allocate the indicated lightpath at the agreed-upon time, as long as the reservation has not been canceled or changed since it was made.

There is no requirement in this implementation for the client application to “reconfirm” or “claim” the reservation, although the application can do this by checking status at the time it uses the allocated lightpath. The NRS considers the reservation confirmed when it is made, but external events could intervene, for example a network outage, which would make it impossible for the NRS to honor the reservation.

Because the current underlying network lacks an interface to provide topology information, our initial implementation does no dynamic topology discovery; instead it relies on tables to describe the network topology.

7.8.2 Data Transfer Service

The Data Transfer Service (DTS) is a middleware service built on top of the NRS and its scheduling facilities. It provides an interface for applications to request the transfer of named data sets from one location on the Grid to another. In the initial implementation, the source host and path name are explicitly required. In the future, with the use of a replica location service, only an abstract source data set name will be required and DTS will choose an appropriate physical source location. Data transfer requests can also specify scheduling parameters for the transfer, using the same syntax as those to NRS: window start and end times, and transfer duration.

DTS may do some processing before calling on the NRS scheduling facility. For example, it might find the appropriate data source, verify its existence, size and accessibility, and interacts with the destination system storage scheduler. The client's request, perhaps modified by additionally discovered constraints from this process, is passed to the NRS scheduler, and the received job ticket is passed back to the client. A cron-like entry is made for the DTS to wake up at the scheduled network allocation and data transfer time.

At the scheduled transfer time, the DTS sends a message to a Data Handler Service (DHS) running on the destination host, which then opens a connection to the source host and transfers the data. Status is passed back to the DTS and is available to client queries and for registered client callbacks.

7.8.3 Grid Integration

The choice of designing framework of the prototype was in large part dictated by the fact that the wealth of distributed, data-intensive Grid applications are very likely candidates for using the network and data services described here. In short, we chose to implement our services within the OGSA framework [29], as OGSI Grid Services [44]. The current trend is to implement using Web services, follow OGSA standards, and be part of WSRF.

Our first implementation represents fully-OGSI compliant Grid Service interfaces for use by Grid applications. Because OGSI is an extension of Web Services, our implementation is also accessible via standard Web Services methods. In their current form, our services fit the stateless connection model of Web Services. We have not made use of OGSI provisions that provide for a more stateful, object oriented model, such as service data and transient service instances, nor have we implemented any callback facilities.

We implemented these interfaces using Version 4 of the Globus Toolkit [40], using SOAP and the JAX-RPC API. Additionally, the interfaces are wrapped in Java classes that effectively hide the OGSI middleware from the applications that call them. In later experiments, we advanced the system from an OGSI base to comply with WSRF.

These interfaces are intended to provide the applications that call them with the basic functionality they need, but without requiring any knowledge of the underlying networks or network control and management protocols.

Overall, our work has confirmed our initial optimism and opened up many possible approaches to exploiting the network infrastructure as an always available and easy to use first class resource.

7.9 Scheduling Future Work

The rich and complex nature of the issues of providing architecture for interfacing the Grid to the dynamic optical network as a service provides fertile ground for further work, especially as middleware and reference applications mature and further experience with the behavior and implications of this work is gained. We can separate the ideas, more or less, into first step elaborations or extensions of the prototype we have described in this chapter, second step, of a grander vision in a third step of long term directions, although there is much overlap in the categories.

In the first step, it is necessary to flash the initial concepts demonstrated in the early functionality discussed here. This includes exploring more complex topologies, mechanisms to route through them, and the interaction of this facility with scheduling in relation to the underlying optical network; creating and exposing interfaces for various administrative client services; optimizations previously noted in the ODIN and OMNInet circuit allocation and deallocation; and other items that would allow a better characterization of the architecture and its implications for the Grid.

In the second step, some areas of improvement will continue to evolve. These include data persistence and restart capabilities for the various services; enhancements to the scheduling modules to allow for a richer request language, including job priorities, event based start times, user-specified cost/utility functions, network/segment cost models, an application resiliency protocol, and the ability to interface to various types of client callbacks; enhancements to the topology routing components to perform re-routing as needed and to use the cost models and utility functions in their figure of merit calculations; a privileged administrative client allowing control of the services from a remote application; and other reference applications, oriented both towards e-Science and towards possible future commercial uses of this infrastructure.

Although the current interfaces to the DTS and NRS provide applications a way of allocating network and data resources, we believe that in many circumstances it would be easier for the user, and make more efficient use of resources, if Grid job schedulers and resource allocators could also allocate and co-allocate these resources in conjunction with other resources needed by a job. For example, a job might need to run after certain very large files had been staged via an optical link, or in a compute environment where and when a dedicated optical path was available to another location. Substantial work remains to be done to define and implement the necessary interfaces to provide these facilities.

Third step areas include extensions of many of those mentioned in the previous paragraphs, as well as exploring the use of, and by other Grid services, such as:

- replica location services, security services, other schedulers, both as clients and as services used;
- enhancements to the Grid services model, as discussed above, to allow the indexing and advertising of services by something as simple as *serviceType/host/port*; negotiations between services, running on multiple administrative domains;

- use of predictive models of network and other resource use to inform scheduling and routing activities;
- multiplexing multiple Lambdas to create "fatter pipes" for even greater throughput;
- and exploring ways of filling these "fat pipes" through collaboration with others in the Grid community addressing these issues.

There are truly great opportunities here from the highest application level, through the interaction of Grid middleware services, down to the various optical networking layers, to add to the functionality and usefulness of this architecture that weds the needs of the Grid to the network as a service.

7.10 Summary

In this chapter, we discussed network scheduling, introduced the new concepts of time value and time window, and elaborated on the need to embed these concepts into a Cyber-infrastructure. We designed and implemented prototypes of the Network Resource Service (NRS), the Data Transfer Service (DTS), and the Data Handler Service (DHS). Within the framework of building the new NSF Middleware Initiative (NMI) for revolutionizing science through Cyber-infrastructure, my goal is to incorporate the innovations into the broader scheme, which to date is missing these vital elements. In case of multiple under-constrained requests, we may face a situation where the system cannot meet all requests. In this case, the final decision and the priority are determined by the Scientific Workflow. The network resource information is shared with the scientific domain middleware across the vertical infrastructures.

What follows in Chapter 8 are algorithms and sets of sub-algorithms to support the scheduling presented in this chapter.

8 Scheduling Algorithms for Network Path Allocations

In Chapter 7, we described scheduling service architecture, Time-Window and Time-Value. We now shift from the service to the algorithms. The algorithms reflect the multi-dimensional nature of the environment, and illustrate the complexity on a very small optical topology. It is necessary for the network domain to hide the internals and simultaneously to expose its ability to adapt, while maintaining the network authority. Complexity arises from the impedance mismatch across technology domains and administration domains.

In reference to the public Internet, it is possible to look at network scheduling as a one-dimension problem, and as such, it appears highly solvable. However, when data requirements are thousands of times greater than typical Internet cases, the scope of the problem dramatically increases.

Previous research work tends to treat network scheduling as a one-dimensional problem. We do not address network scheduling as a self-contained entity, but rather, as a process that must exist within the context of network reservation and allocation inside the big e-Science workflow picture, and as part of the Cyber-infrastructure. We address the interaction, collaboration, and negotiation between the network middleware and the scientific workflow, scientific middleware, and grid middleware.

The assembly of a unique network topology for each scientific experiment requires an orchestration of resources. This can only occur if the network "intelligence" software completely understands the requirements and can adapt to all other components

What follows is a description of a simple topology within a simple scenario, and a schematic that addresses challenges that arise. We introduce two new ideas of Segment Reservation Authority (SRA) and Path Reservation Authority (PRA), and show how to discover a set of candidate paths, construct proposals for satisfying the reservation request, evaluate all proposals, and select and implement the best proposal. All of this is done with PRA and several SRA negotiations in order to contract the right path proposals. Furthermore, we explain how the information is communicated to the Cyber-infrastructure middleware such as the National Partnership for Advanced Computational Infrastructure (NAPACI) and NSF Middleware Initiative (NMI)], and how the resources negotiate and align according to the availability of other resources.

8.1 Resource Scheduling

An optical connection between two nodes is one that can only have a single user at a time; thus, the concept of network scheduling comes into play. A scheduled, dedicated, circuit-switched connection could be an essential tool for e-Science application transferal of immense amounts of data. In practice, optical connections have been statically provisioned and dedicated to a single user. The proposed shift is from statistical multiplexing to a plane with an orchestrated, dedicated connection allocated for a period of time.

Another differentiation is that a Lambda Data Grid provides a knowledge plane that understands the application requirements and allocates network resources. Unlike traditional packet switching networks, the forwarding decision moves from packet headers in L3-L4 to external intelligence that interacts with L7 and L0.

Figure 8.1 represents a scheduling reroute. In the upper Figure, a “red” route was allocated between “A” and “B”; a new “blue” request comes in for “X”, the segment in use. In the bottom Figure, first the “red” route can be altered to use a different path to allow the “blue” route to also be serviced in its time window.

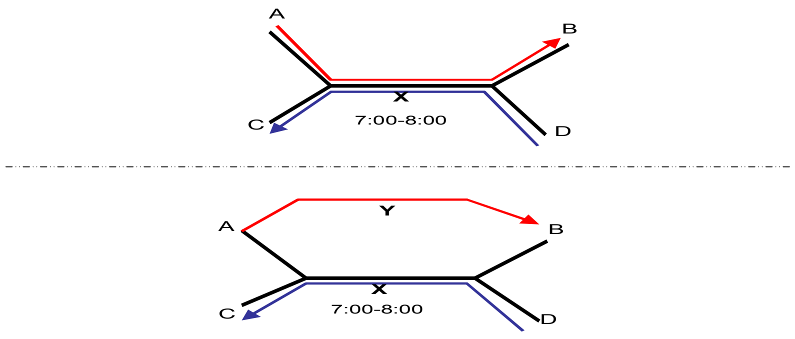


Figure 8.1 – Scheduling via alternate route for the same Time-Window.

Figure 8.2 is a screenshot for running the algorithm with three time windows and no conflicts. The algorithm is running on a simple case of four nodes and possible six connections as depicted on the bottom of the screen. The top section, each of the six lines represent single link, where we can see in narrow line the time window requested and in the wide line the time allocated.

Figure 8.3 is a screenshot for the results after adding the conflicted requests, meaning additional request for the same segment for the same time-window. For example segment 3-4 shifter -

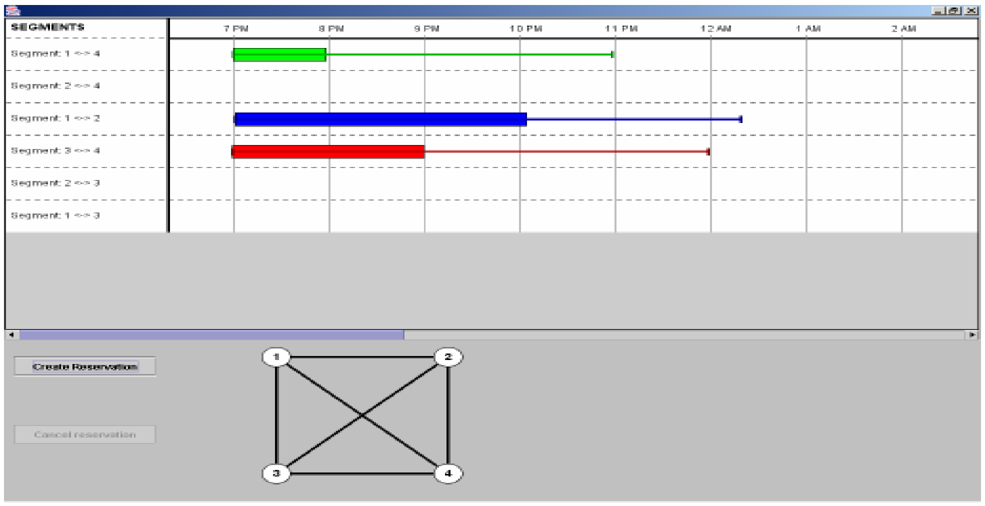


Figure 8.2 – Simple time-window requests with no conflicts.

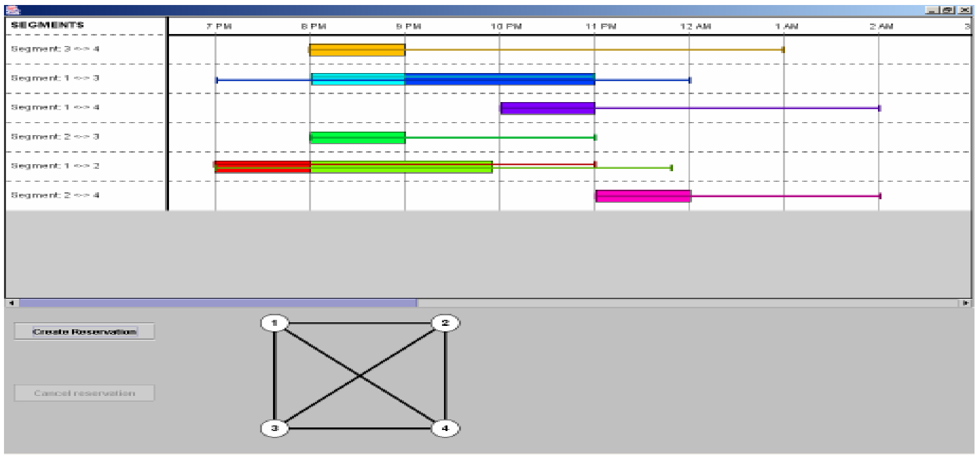


Figure 8.3 – Adding new requests will be resolved to avoid conflicts.

What follows is a discussion of, 1) overview and terminology, 2) distributed scheduling algorithm, 3) simple scenario of durations with windows, and we conclude with, 4) future work and current limitations.

8.2 Overview and Terminology

We consider networks as consisting of multiple segments, with special nodes termed endpoints and paths constructed between those nodes. The network may have dynamic topology, the segments may have special qualities such as variable channel number, and there may be many types of cost functions associated with network loading. Our work focuses on the general problem of optimizing a schedule of advance reservations for paths between two given endpoints. There may be multiple sets of segments that qualify as a path between the two endpoints, and the advance reservation generally does not specify the segments that will be involved. Of particular interest is the case of reservations that may be rescheduled within associated bounds, because this case allows for a significant degree of optimization, regardless of the cost model in use.

In general, segments are assumed to be independently scheduled, each with its own advance reservation agent termed a "Segment Reservation Authority," or SRA. The primary client of an SRA is a "Path Reservation Authority," or PRA. A PRA is a higher-level service that provides advance scheduling of paths; that is, it reserves network paths between two endpoints by getting segment reservations from the SRAs. The client of a PRA can be an end application, or a further middleware service. In the following discussion, we assume only one global PRA, and no other clients of the SRAs. However, this algorithm can be extended to allow multiple PRAs sharing some SRAs as needed, as in the case of multiple administrative domains.

SRAs issue fixed reservations, while the PRA issues reservations that could possibly be rescheduled within certain requested bounds. Such functionality is important for achieving optimal usage of network resources, but requires client applications to work with a reservation-ticket model in which the reservation may be moved unless it is locked down. The PRA attempts to satisfy client requests for advance reservations of paths between endpoints. Optionally, the client-requests may contain other attributes, such as number of channels. In general, requests are under-constrained, i.e. they allow some flexibility in how they are satisfied. This creates an optimization problem, a challenge to satisfy requests in the most globally cost-efficient manner. If previously scheduled reservations may be shifted within the bounds of their requests, better global solutions may be realized. The client request language for path reservations may be considered in general terms. The simplest case is treated in more detail below, and reflects a request for a path between endpoints End A and End B, to last a given duration D, with a specified "start after" time SA and "end before" time EB, with all network segments having a single available channel. However, the high-level algorithm described here could apply to much richer scenarios, such as a request for a given total aggregate throughput within a certain time frame, with soft boundaries, or "utility functions," supplied as part of the request, and with more than one possible channel per segment.

The issue of concurrency and synchronization deserves a special note. This high-level algorithm can be applied in multiple situations that may require various notions of synchronization. At one extreme, the entire system can be synchronized at the request level, so that multiple client requests are queued and executed sequentially. This is feasible if using a single monolithic service for all SRAs and the PRA. At another extreme, multiple SRAs and even multiple PRAs can be coordinated to allow concurrent request processing using a transaction protocol. The design of such a transaction protocol is a significant project in its own right, one that is not being attempted here.

8.3 Distributed Scheduling Algorithm

By assuming certain provided services, it is possible to present a high-level algorithm for the basic scheduling problem as described above in a general setting. This is a framework for specific low-level algorithms that are adapted to the needs of a particular implementation. The purpose is to find good-enough solutions, not necessarily the ultimate ideal ones. This is a distributed algorithm, with essential elements of computation performed by potentially independent entities.

8.4 High-Level View

At the highest level, the algorithm consists of four steps. Given a request for a path reservation as described above, the PRA attempts to create a reservation as follows:

1) Discover a set of candidate paths between the two endpoints, exclusive of reservations. In a large network, it may not be feasible to discover all paths.

2) Construct proposals for satisfying the reservation request, which may involve changing other existing reservations. If so, the proposal includes the reservations to be changed and the exact way in which they should be changed; e.g., this may involve using a different path for an existing reservation. Proposals are guaranteed to be "correct" in that all proposed reservations may be implemented without a conflict with any existing reservation.

3) Evaluate all proposals and select the one that best maximizes utilization, balancing all costs.

4) Implement the chosen proposal, beginning with any required changes to existing reservations.

From a high-level point of view, each of these steps is straightforward, except for step two. Topology discovery and proposal evaluation may use complex heuristics in any specific low-level implementation; this is not of concern here.

The bulk of the algorithm discussed here is concerned with the construction of plausible reservations. If rescheduling of previously issued reservations is not allowed, a very straightforward algorithm suffices to find a suitable open window of time shared by each segment in the path, with optimization of any utility functions supplied with the request, for each candidate path. If no window can be found, the request is added into a waiting list with notification to the application. The system is constantly changing and optimization is on going. A waiting list is taken into considerations during this optimization. When a window is found, the application is notified specifying the available future window.

If rescheduling is allowed; however, significant optimization in total network usage can be achieved. This algorithm is concerned with the problem of identifying proposals that may involve rescheduling, while satisfying all reservation constraints, prior and current.

Reservation proposals, including reservation change proposals, are constructed via a recursive interplay between the PRA and the "segments" (SRAs). In brief, the PRA asks the segments for proposals. In turn, the segments ask the PRA about the feasibility of changing existing path reservations with conflicting segment reservations. The PRA checks on this by effectively removing the reservation in question and then attempting to schedule it again as a new request, using the original reservation request. This, in turn, calls back into the segments, and the recursion continues. The recursion must take care that the reservation is never actually removed, and also to pass a context of presumed reservation changes in order to allow changes involving multiple reservations. This algorithm permits multiple points at which recursion depth is controlled, e.g. through cost functions. This would most likely be a tunable parameter, assuming different values appropriate for different network sizes.

The remainder of this section describes this more precisely, as depicted in Figure 8.4. The top-level of the "four-step algorithm" is "CreateReservationFromRequest," executed by the PRA given a client request. This makes use of a sub-algorithm "ConstructProposals." The recursion between segments and paths enters through the SRA function "GetWindowsForRequest," which calls back to the PRA "ProposeReschedule." ProposeReschedule effectively executes steps 1 through 3 of the top-level CreateReservationFromRequest, and in so doing may recurse.

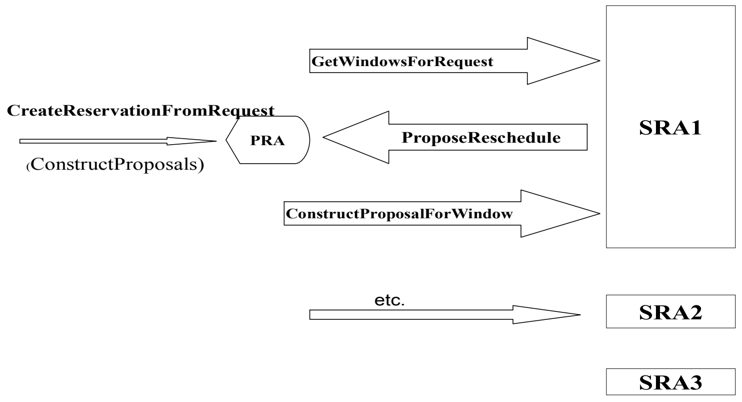


Figure 8.4 – Four-step algorithm.

8.5 PRA: CreateReservationFromRequest

Definitions:

SegmentReservation – an advance reservation for the use of a segment, with specific attributes such as bandwidth, associated with an 'owner' PathReservation, with a specified reservation authority and reservation ticket in the case of multiple PRAs.

PathReservation – an advance reservation for network resources between two endpoints, with specified start and end times, using a specified list of segments and associated SegmentReservations, all of which share the same attributes. Every PathReservation is associated with a unique PathReservationRequest.

PathReservationRequest – a request for a PathReservation specifying the endpoints but with no reference to the choice of segments. The request contains "Constraints" that affect the possible satisfaction of the request. This is typically a temporal constraint, but it could have other qualities as well, such as number of channels, or a request that the reservation not be rescheduled once issued. Although referred to generically, there is a simple procedure to determine if a given proposed PathReservation satisfies a given PathReservationRequest. Optionally, this procedure may produce a "score" or "cost" rather than a binary satisfied/not satisfied distinction.

PathReservationProposal(PRP) – a proposal for satisfying a PathReservationRequest contains a list of segments, a start and end time, other attributes such as number of channels, and a PathReschedulingPlan for existing path reservations as described below.

PathReschedulingPlan – an ordered list of existing PathReservations along with possibly new paths and new start/end times. A PathReschedulingPlan can be executed by first deleting all of the existing PathReservations referenced by the plan, and then recreating each reservation using the new segment list and new start/end times. Optionally, a PathReschedulingPlan may contain an associated cost computed during its construction.

NetworkResourceReporter (NRR) – a service encapsulating knowledge of the network resources (topology, channels, etc.), including schedules of availability. The NRR reports known network resources independent of the SegmentReservations managed by the SRAs, possibly consulting outside entities or other hardware-level schedules. The NRR provides a single function:

[path_records] = ReportPaths
(*endpoint1,*

endpoint2,
timewindow)

Given a pair of endpoint nodes and a specific time interval, it reports a set of paths between these two nodes that is valid during this time interval. Alternatively, it may report paths that are available for only some sub-interval, along with the available sub-interval. Note that if the network is too large to construct all possible paths, the NRR produces a list of plausible candidates, and only these are considered by the scheduling algorithm. This functionality is treated as a “black box” by the algorithm.

Given:

– a network of segments with named endpoint nodes, an NRR as defined above, and associated SRAs

– an existing set of PathReservations, each PathReservation consisting of a set of SegmentReservations and an associated PathReservationRequest

– a new PathReservationRequest for a path between *Endpoint1* and *Endpoint2*, with associated Constraints

Problem: Construct a PathReservation that satisfies the PathReservationRequest while minimizing the costs of the request constraint satisfaction and the associated ReschedulingPlan, or report that no satisfaction is possible. The nature of the cost is determined by many factors. The input to this algorithm is a linear number that is the result of a function of a vector of many parameters. In this algorithm, we did not determine the exact value of each parameter, nor the exact function. The value is based on the time value described in Chapter 7, application requirements, negotiation and renegotiation with the applications, predicted network availability at a given time in the future, and the cost of other resources required in the co-scheduling.   The goal is to minimize the cost of an individual reservation and to maximize the utilization of the entire system.

Step 1 (Topology): Obtain a set of possible paths ≤P≥

1a. Invoke the NRR service's `ReportPaths` function to get a set of possible paths ≤P≥ from the PathReservationRequest's *Endpoint1*, *Endpoint2*, supplying a time-window large enough to cover all possible times that would satisfy the request Constraints. Open bounds can be represented by arbitrarily large values.

1b. (optional) Compare the subinterval reported with each possible path to the request Constraints, and remove any paths from ≤P≥ which could not satisfy the Constraints.

At this point ≤P≥ is a set of paths known to satisfy the physical resource constraints of the request. The required resources will be physically available, but may be otherwise reserved. In simple scenarios with small networks, this step may consist of nothing more than a table lookup, assuming all possible paths between all endpoint pairs are pre-computed.

Step 2 (Proposal Generation): Obtain a (possibly empty) set of possible PathReservationProposals, ≤PRP≥. For each path P in ≤P≥, with optional constraints reported by the NRR, get a (possibly empty) set of PRPs by invoking the PRA ConstructProposals function with the path P (see below). Accumulate these sets for all paths in ≤P≥ to generate the set ≤PRP≥.

Step 3 (Proposal Evaluation): Assign global costs to each proposal in the set ≤PRP≥, by computing the satisfaction measure of the proposed reservation against the request Constraints, and optionally adding the reported ReschedulingPlan costs or any other cost measure. This step is dependent on the exact client-request language and cost models.

Step 4 (Proposal Execution): Select the minimal cost proposal from the ≤PRP≥ set. Execute the associated ReschedulingPlan, changing existing reservations as specified in the indicated order (see ReschedulingPlan definition above). Create the new proposed PathReservation, creating new SegmentReservations as specified.

END **CreateReservationFromRequest**

This algorithm obtains the topology, propose set of possible path reservations, evaluates the proposals, and selects one with minimal cost. It assumes a simple and small topology, on a single domain, with full control, and equal bandwidth on all links. This is not the case on a typical network, but provides an initial view of the needs of a real network. Without any optimization, the algorithm runs for minutes on a standard PC before selecting a minimal cost proposal on a small network. The complexity will grow substantially with a real network. The assumption is that this type of computation is done in the background during hours and days prior the path reservation.

8.6 PRA: ConstructProposals

Define:

SegmentReservationRequest – identical to PathReservationRequest, but with a single specified segment instead of endpoints.

SegmentReservationAuthority (SRA) – a service providing low-level advance reservations of a particular segment. In practice, many or all segments may be serviced by the same "meta" SRA service. The SRA provides the following functions, which are described as sub-algorithms below in Figure 8.5.

```
    ≤[currentWindow, largestWindow]≥ = GetWindowsForRequest
    (  PathReservationRequest,
       ≤allowedWindows≥    )
    PRP = ConstructProposalForWindow
    (  window,
       priorReschedulingProposal      )
```

Figure 8.5 – SRA sub-algorithms.

Given: a path P of segments and an existing set of path and segment reservations and a new ReservationRequest as described above.

Problem: construct a set of PathReservationProposals (PRPs) that employ this particular path and satisfy request constraints.

Step 1: Pick any segment, S1, and call the associated SRA:GetWindowsForRequest function, with an allowed window constructed from the request large enough to cover all possible reservation satisfactions.

Step 2: Iterate the above for all segments in the path, using the resulting set of windows as the AllowedWindows parameter for the next segment.

If the window set becomes empty, return an empty set from ConstructProposals.

Step 3: The final window set is now a set of possible windows for a PRP. For each window in turn:

3a: Construct a specific reservation time within the window chosen to optimally satisfy the original request Constraints.

3b: Call ConstructProposalForWindow on the SRA associated with any segment, with null ignoredPathReservation and no prior rescheduling proposal.

3c: Iterate the above for all segments in the path, using the result PRP from the previous segment as the prior PRP for the next. If this method fails (returns null) for any segment, advance to the next tested window.

3d: If all segments pass, add the final PRP to the ConstructProposals result set, and advance to the next tested window.

END **ConstructProposals**

This sub-algorithm constructs a set of path reservation proposals that employ this particular path and satisfies request constraints.

8.7 SRA: GetWindowsForRequest

Given: a path P of segments and an existing set of path and segment reservations and new path request as described above.

Problem: construct a set of pairs of windows that represent possible reservation times for this segment that match the conditions of the request. Each window pair consists of an existing window and a maximum window size. The maximum window is obtained by allowing rescheduling of existing path reservations.

Step 1: Examine the existing SRA schedule to identify all possible windows that intersect the request constraints.

Step 2: For each window identified in step 1, identify the PathReservations corresponding to bounding SegmentReservations that constrain the window size.

Step 3: Perform sequential selective winnowing based on the existing schedule:

Step 3a: For each PathReservation identified in step 2, call the PRA ProposeReschedule function with an exclusion window calculated to cover the maximum useful window for the original request Constraints.

Step 3b: Use the results from the ProposeReschedule calls to compute a maximum resized window for this window, and append the "window/maximum window" pair to the return set, or exclude this window if the ProposeReschedule fails to construct a large enough improvement.

END **GetWindowsForRequest**

This sub-algorithm adds another new dimension of time-window selection to the previously presented path selection. Selecting the right time-window for the right path selection is a complicated problem. For the tested assumptions presented in Section 8.6, and the time-window presented in Section 8.7, we found additional complexity in the order of about a magnitude.

8.8 SRA: ConstructProposalForWindow

Given: a path P of segments and an existing set of path and segment reservations and a new path request as described above, a specific set of reservation times, and a prior rescheduling proposal of reservation changes to assume. The specific reservation times were previously reported as possible by the GetWindowsForRequest function.

Problem: Construct a new rescheduling proposal that incorporates the prior proposal and adds new rescheduling as needed to allow a reservation for this segment at the specified times.

Step 1: Identify the PathReservations corresponding to bounding SegmentReservations that interfere with the requested reservation times.

Step 2: For each PathReservation identified in step 1, call the PRA ProposeReschedule function with an exclusion window equal exactly to the requested reservation window. Return the accumulated rescheduling proposal, or null if the ProposeReschedule fails. This may happen due to rescheduling interactions that could not be detected during the GetWindowsForRequest procedure.

END **ConstructProposalForWindow**

For a small network, this looks like a linear extension of path selection.

8.9 PRA: ProposeReschedule

Given: a path P of segments, an existing set of path and segment reservations, an existing path reservation, a supplied "exclusion window" that the reservation should avoid, and a prior rescheduling proposal of reservation changes to assume.

Problem: construct a PathReschedulingPlan that reschedules this path reservation to avoid the exclusion window as much as possible. This may entail a choice of a different segment list for the path.

8.9.1 Schema:

ProposeReschedule follows a parallel execution track to the top-level CreateReservationForRequest. However, it uses variants of each of the functions above (ConstructProposals, GetWindowsForRequest, ConstructProposalForWindow) that take additional parameters "ignoreReservation" and a prior rescheduling proposal, then the original client request language is extended with the exclusion window. The "ignoreReservation" parameter is used to specify the reservation being rescheduled, and it effectively means that the particular reservation should be ignored for purposes of schedule planning for its successor.

END **ProposeReschedule**

This sub-algorithm does not guaranty to construct a new reschedule proposal. Similar to the results of this sub-algorithm, after several simple manual analytical or graphical exercises with pencil and paper, we could not find a window for a rescheduling solution. For future work, we suggest to enhance this algorithm to guaranty a solution.

8.10 Simple Scenario: Durations with windows

This section describes a specific implementation of a simple case. The client request language is for reservations of a specified duration, with a specified “start after” and “end before” time and no bandwidth information. The client cost function returns a simple success or failure to satisfy these constraints. The same monolithic server implements all reservation authorities and path reservation authority.

In this scenario, the function of the NRR (step 1 of the main algorithm) is performed by a simple table lookup, and proposal evaluation (step 3) picks the proposal with the fewest required reschedulings. It is assumed that there are no other sources of network allocations.

ConstructProposals, GetWindowsForRequest, ConstructProposalForWindow, and ProposeReschedule all “slide” reservation windows forward or backward in time, respecting the requested “start after” and “end before” constraints of all reservations and doing all the recursive bookkeeping. This greatly simplifies the recursion logic, and guarantees recursive termination since reservation changes are always in the same direction. The first proof-of-concept prototype does not change the order of path reservations, but simply moves them forward or backward while retaining their relative order.

8.11 Summary

This chapter presents a four-steps algorithm that, given a request for a path reservation the PRA, attempts to create a reservation as follows:

1) discover set of candidate paths,

2) construct proposals for satisfying the reservation request,

3) evaluate all proposals and select the one that best maximizes utilization, balancing all costs, and

4) implement the chosen proposal, beginning with any required changes to existing reservations.

The four-step algorithm reflecting the multi-dimensional nature of the environment and illustrate the complexity on a very small optical topology. It was necessary for the network domain to hide the internals and simultaneously to expose its ability to adapt, while maintaining the network authority. Complexity arises from the impedance mismatch across technology domains and administration domains.

9 Summary and conclusion

In this thesis, we describe the building of a new network middleware that is integral to Grid middleware to manage dedicated optical networks. In simple terms, we built an orchestration for specifically dedicated networks for e-Science use only. We believe this research represents one of the pioneer efforts in encapsulating the network resources into a Grid service, accessible and schedulable through the enabling architecture. As such, it opens up several exciting areas of research discussed in the future work section.

The nature of new e-Science research requires middleware, Scientific Workflows, and Grid Computing in a distributed computational environment. This necessitates collaboration between independent research organizations to create a Virtual Organization (VO). Each VO addresses organizational needs across a large scale geographically dispersed area, and requires the network to function as a fundamental resource. We have built a technology that allows access to abundant optical bandwidth using Lambda on demand. This provides essential networking fundamentals presently missing from Grid Computing research, overcoming bandwidth limitations, to help make VO a reality.

Essential to e-Science research is the ability to transfer immense amounts of data, from dozens to hundreds of TeraBytes, and even PetaBytes, utilizing Grid Computing as the fundamental platform to conduct this big-science research. Vast increases in data generation by e-Science applications, along with advances in computation, storage and communication, have changed the nature of scientific research. During this decade, we will continue to see advancements, including crossing the “Peta” Lines: Petabyte, Petaflop, and Petabit/s.

In the next decade, e-Science requirements will be for a network with a capacity millions times greater than today’s public Internet. The distribution of large amount of data is limited by the inherent bottleneck nature of the public Internet architecture and packet switching technologies. Bandwidth limitations inhibit the advancement and utilization of new e-Science applications in Grid Computing. These emerging e-Science applications are evolving in data centers and clusters; however, the potential capability of the globally distributed system over large distances is yet to be realized.

Current network orchestration is done manually, in a labor-intensive manner via multi-party conference calls, emails, yellow sticky notes, and reminder communications, all of which rely on human interaction for outcome, crossing organizational boundaries. The work in this thesis automates the orchestration of networks with other resources, better utilizing all resources in a time efficient manner. Automation allows for a vastly more comprehensive use of all components and removes human limitations from the process. The network becomes a first-class managed resource akin to computation, storage, data, unique sensors, and visualization. Together with computation and data, scientists can co-allocate and co-schedule resources for greater utilization.

This thesis reports automated Lambda provisioning from Chicago to Amsterdam in 70-100 seconds, compared to approximately three month it takes for manual scheduling and allocation. Automation allows for a vastly more comprehensive use of all components within the Scientific Workflow, to achieve results not possible by manual allocation. Albeit, this is a small-scale model using only two transatlantic Lambdas and is minute compared to the millions of lambdas provided, it is one step toward the solution of managing globally distributed data in enormous amounts.

Lambda Data Grid provides the knowledge plane that allows e-Science research to allocate Lightpath. This will enhance e-Science research by allowing large distributed teams to work efficiently, using simulations and computational science as a third branch of research.

The following is a summary of the three fundamental challenges this thesis addresses.

9.1.1 Challenge #1: Packet Switching – the wrong solution for Data Intensive applications

Packets are appropriate for small amounts of data like web pages and email. However, they are not optimal for e-Science applications similar to Visual Observatories, for example, that will generate Petabytes of data annually in the next decade. It is impossible to transfer large-sized data on today's public Internet using L3 packet switching. Such an attempt would grossly destabilize Internet traffic. When dealing with nine-orders of magnitude difference in transfer size, questioning the usefulness of current methodologies is necessary. Intrinsic to the packet switching challenge are the following considerations:

- The demand for the transfer of bulk data challenges us to examine the scalability of data transfer at the core of Internet.
- The kind of cut-through methods that could be effective. The integrity of end-systems and the edge devices must remain intact, but the underlying optical core demands rethinking.

- Statistical multiplexing can work for many-to-many small traffic patterns, as found in systems like today's Internet. For few-to-few bulk traffic patterns, as seen in astrophysics research, statistical multiplexing loses its benefits.
- For normal use of the network, constant availability is assumed. In most cases, there are no mechanisms to regulate when to use the network, under what conditions, and how much data to transmit. This is efficient because the "normal" transfer data size is a tiny fraction of the available bandwidth at the core. However, this is not expected in data intensive applications.
- During the last 30 years, we followed the fundamental design principle of bandwidth conservation. Advances in optical networking brought forth the concept of wasting bandwidth rather then conserving it.

The nearly universal decision to use packet switching rather than circuit switching was arrived at for a variety of good reasons. However, in optical switching, utilization, bandwidth optimization, conservation and transmission costs, are not primary goals. With new tradeoffs for large data sets, lightpath is the optimal solution. Lightpath is actually an implementation of circuit switching.

9.1.2 Challenge #2: Grid Computing Managed Network Resources

Typical Grid applications require the management of highly distributed resources within dynamic environments. A basic problem is matching multiple and potentially conflicting application requirements to diverse, distributed resources within a dynamic environment. Other problems include methods for network allocation of large-scale data flows, and co-allocation with other resources like computation and storage. Abstraction and encapsulation of network resources into a set of Grid services presents an additional challenge. Future related unmet challenges include scheduling, co-scheduling, monitoring, and fair-shared usage within a service platform.

Common architectures that underlie traditional data networks do not incorporate capabilities required by Grids. They are designed to optimize the small data flow requirements of consumer services, enterprise services, and general common communication services. Many Grid applications are data-intensive, requiring specialized services and infrastructure to manage multiple, large-scale data flows of multiple Terabytes and even Petabytes, in an efficient manner. Such capabilities are not effectively possible on the public Internet or in private routed packet data networks. For this type of traffic, the standard Internet or even QoS mechanisms on the public Internet will not accommodate the quantity of data. The underlying principle of constant availability and a shared network is not affective for this type of traffic. Further, scheduling of network resources and co-scheduling with other resources is not part of the existing mechanisms. Negotiation and interaction between the network and the applications regarding requirements and availability does not exist today.

It is necessary to provide applications with direct, flexible access to a wide range of optical infrastructure services, including those for dynamically provisioned optical path channels within an agile optical network. There is a need to design network architectures that can support Grid applications in association with emerging optical networks.

9.1.3 Challenge #3: Manage Data Transfer for Big Science

In the world of scientific research, bringing together information and collaboration is crucial for scientific advances. Limitations in technology and the inability to orchestrate resources prohibit the usability of these one-of-a-kind facilities and/or instruments by the wider community of researchers. To function effectively, e-Science researchers must access massive amounts of data in remote locations. From massive, one-of-a-kind, real-time remote sensors, or from immense remote storages, researchers must filter the data and transfer minuscule portions for their use. The challenge is to get the right data, to the right location, at the right time.

Further, non-experimental work could benefit from very high capacity networking. Consider for example interlinked models used for climate simulation. There might be an atmospheric model that interacts with an oceanic model as well as with a solar model to address how radiation flux and solar storms affect the upper atmosphere. Econometric models could look at how climate will affect land use patterns, agriculture, etc. and how it might feed back into atmospheric effects. Each simulation would run at its own center of expertise, requiring high-speed data connections to communicate at each time step.

9.2 Our main contributions are:

9.2.1 Promote the network to a first class resource citizen

- The network is no longer a pipe; it is a part of the Grid Computing instrumentation. In addition, it is not only an essential component of the Grid computing infrastructure but also an integral part of Grid applications. This is a new design principle for Grid and High-throughput Computing. The proposed design of VO in a Grid Computing environment is accomplished and lightpath is the vehicle, allowing dynamic lightpath connectivity while matching multiple and potentially conflicting application requirements, and addressing diverse distributed resources within a dynamic environment.

9.2.2 Abstract and encapsulate the network resources into a set of Grid services

- Encapsulation of lightpath and connection-oriented, end-to-end network resources into a stateful Grid service, while enabling on-demand, advanced reservation, and scheduled network services. In addition, a schema where abstractions are progressively and rigorously redefined at each layer. This helps to avoid propagation of non-portable implementation-specific details between layers. The resulting schema of abstractions has general applicability.

9.2.3 Orchestrate end-to-end resource

- A key innovation is the ability to orchestrate heterogeneous communications resources among applications, computation, and storage, across network technologies and administration domains.

9.2.4 Schedule network resources

- The assumption that the network is available at all times, to any destination, is no longer accurate when dealing with big pipes. Statistical multiplexing will not work in cases of few-to-few immense data transfers. We have built and demonstrated a system that allocates the network resources based on availability and scheduling of full pipes.

9.2.5 Design and implement an Optical Grid prototype

- We were able to demonstrate dynamic provisioning of 10Gbs in 100 seconds, replacing the standard provisioning of at least 100 days. This was shown in a connection from Amsterdam to Chicago during Super Computing and on the conference floor in Pittsburg. Currently, these types of dedicated connections must be performed manually, and must take into consideration all possible connections. We have automated this process, in a small conceptual model of two connections with alternate routes. For technology demonstrations, Cees De Latt

[43] described the previous standard process of provisioning 10Gbs from Amsterdam to Chicago in general terms as follows: It took about 300 emails, 30 conference and phone call and three months to provision the link. This lengthy process is due to complications associated with crossing boundaries: organizational, domain, control, administrative, security, technology, and product interoperability. These boundary challenges are in addition to issues such as cost structure, billing, policy, availability, and priority. Provisioning within boundaries has vastly improved thanks to new Lambda service, which takes only a few dozen seconds to create an OC-192 coast-to-coast, compared to the three to six months it takes commercially.

9.3 Future Work

There are a number of interesting directions future work can take. Some are extensions of work in this dissertation while others address more general problems of integrating dynamic Lambdas as part of scientific research. The addition of other underlying file transfer protocols is one area to explore. Our work presents simple scheduling with a limited topology. More complex scheduling algorithms on larger topology should be investigated, including the ability to query the network for its topology and the characteristics of its constituent segments and nodes, to be able to route over the topology and to do segment-level scheduling, allocation and de-allocation.

More complex will be the development of cooperative protocols for interacting with other Grid resources (such as replica location services and local storage management services) and schedulers, both providing services to them and using them to provide inputs into the schedules and connectivity we provide.

But most of all, we believe that the greatest learning will be achieved by working with a user community with pressing needs, real networks, and large amounts of data, to be sure that Lambda Data Grid solves the right problems in ways that are immediately useful and transparent to the user community. To that end, work must be done with potential users to fully understand how they can use the Grid middleware services presented here. We must work together to address the important issues about using these services, which promote the network to a higher capacity, and functions as a reliable, schedulable entity.

REFERENCES

==

1. T. DeFanti, M.B., Eds., , *NSF CISE Grand Challenges in e-Science Workshop Report* National Science Foundation Directorate for Computer and Information Science and Engineering (CISE), Advanced Networking Infrastruture and Research Division, (Grant ANI 9980480), , 2001(University of Illlinois at Chicago).
2. Ron Whitney, L.P., Wu-chun Feng, William Johnston, *Networking Challenges Roadmap to 2008.* Department of Energy (DOE) Office of Science, 2003.
3. *Moor's Law - Cramming more components onto integrated circuits.* Electronics Magazine, 1965(April, 19).
4. Coffman, K.G. and A.M. Odlyzko, *Growth of the Internet*, in *Optical Fiber Telecommunications IV B: Systems and Implementation*, I.P. Kaminow and T. Li, Editors. 2002, Academic Press. p. 17-56.
5. Gilder, G., *Telecosm: The World After Bandwidth Abundance*. 2nd ed. 2003: Free Press.
6. Zimmermann, H., *OSI Reference Model—The ISO Model of Architecture for Open Systems Interconnection.* IEEE Transactions on Communications, 1980. **28**(4): p. 425 - 432.
7. Pablo Molinero-Fernadez, N.M. and H. Zhang, *Is IP going to take the world (of communicaions)?* ACM SIGCOM Computer Communicaions Review, 2003. **33**(November 1, 2003): p. 113-118.
8. *CyberInfrastructure Vision for 21st Century Discovery.* National Science Foundation (NSF), 2006(Ver 0.5).
9. *LHC at CERN*: http://www.cern.ch/LHC.
10. *Review of the Stanford Linear Accelerator Center Integrated Safety Management System.* U.S. Department of Energy Office of Science, 2005. **Final Report**.
11. *Collider Detector at Fermilab*. [cited; Available from: http://www-cdf.fnal.gov/.
12. Chiyan Luo, M.I., Steven G. Johnson, and J. D. Joannopoulos, *Ring Imaging Cherenkov Detector (RICH)* Science, 2003. **299**(368–371).
13. *LBNL http://www.lbl.gov/*.
14. *Babar http://www.slac.stanford.edu/BFROOT/*.
15. *SLAC http://www.slac.stanford.edu/*.
16. *National Virtual Observatories, the NVO, http://www.us-vo.org/*
17. *LIGO - Laser Interferometer Gravitational Wave Observatory http://www.ligo.caltech.edu/*.
18. *EROS http://edc.usgs.gov/*.

19. NASA, *Goddard Space Flight Center (GSFC)* *http://www.gsfc.nasa.gov/*.

20. *Protein Data Bank (PDB)* *http://www.rcsb.org/pdb/Welcome.do*.

21. *The National Institute of Health (NIH),* *http://www.nih.gov/*

22. Arie Shoshani, A.S., Junmin Gu Nineteenth *Storage Resource Managers: Middleware Components for Grid Storage.* Nineteenth IEEE Symposium on Mass Storage Systems, 2002 (MSS '02)

23. Hoo, G., et al., *QoS as Middleware: Bandwidth Reservation System Design*, in *The 8th IEEE Symposium on High Performance Distributed Computing*. 1999.

24. *Web Services Addressing, World Wide Web Consortium*: http://www.w3.org/Submission/ws-addressing/.

25. *Enlightened Computing* *http://www.enlightenedcomputing.org/*.

26. *http://www.globus.org*. [cited.

27. *Super Computing Conferance 2004* *http://www.supercomputing.org/sc2004/*.

28. Foster, I. and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*. Vol. 2nd Edition. 2004: Morgan Kaufmann.

29. Foster, I., et al., *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.* Grid Forum Document, No. xx, June 2002.

30. *The CANARIE project at* *http://www.canarie.ca/canet4*.

31. Figueira, S., et al., *DWDM-RAM: Enabling Grid Services with Dynamic Optical Networks*, in *Workshop on Grids and Networks*. April 2004: Chicago, IL.

32. Lavian, T., et al., *A Platform for Large-Scale Grid Data Services on Dynamic High-Performance Networks*, in *First International Workshop on Networks for Grid Applications (Gridnets 2004)*. October 2004: San Jose, CA.

33. *The Web Services Resource Framework (WSRF) Technical Committee, Organization for the Advancement of Structured Information Standards*: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf.

34. Smarr, L., et al., *The OptIPuter.* Communications of the ACM, Nov. 2003. **46**(11): p. 68-67.

35. DeFanti, T., et al., *TransLight: A Global Scale LambdaGrid for E-Science.* Communications of the ACM, Nov. 2003. **46**(11): p. 34-41.

36. Foster, I., C. Kesselman, and S. Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations.* International Journal of High Performance Computing Applications, 2001. **15**(3).

37. Leon Gommans, F.D., Cees de Latt, Arie Taal, Alfred Wan, Tal Lavian, Inder Monga, Franco Travostino, *Applications Drive Secure Lightpath Creation accross Heterogeneus Domains.* IEEE Communications Magazine, 2006. **44**(3): p. 100-106.

38. *Human Genome Project (HGP)* 2003.

39. *BIRN -* *http://www.nbirn.net/*.

40. Foster, I., *Globus Toolkit Version 4: Software for Service-Oriented Systems*, in *International Conference on Network and Parallel Computing*. 2005, Springer-Verlag. p. pp, 2-13.

41. Lavian, T., et al., *An Extensible, Programmable, Commercial-Grade Platform for Internet Service Architecture.* IEEE Transactions on Systems, Man, and Cybernetics, 2004. **34, Part C**(1): p. 58-68.

42. Nabryski, J., M. Schopf, and J. Weglarz, eds. *Grid Resource Management*. Fall 2003, Kluwer Publishing.

43. Roy, A. and V. Sander, *GARA: A Uniform Quality of Service Architecture*, in *Resource Management: State of the Art and Future Trends*. 2003, Kluwer Academic. p. 135-144.

44. S. Tuecke, K.C., I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, and D. Snelling, *Open Grid Services Infrastructure (OGSI) Version 1.0,"* Global Grid Forum Draft Recommendation, 2003.

45. Dugla Thain, T.T., and Miron Livny, *Condor and the Grid.* Grid Computing Making the Global Industry a Reality, 2003: p. 299-335.

46. Tuecke, S., et al., *Open Grid Services Infrastructure (OGSI) Version 1.0.* Grid Forum Document, No. xxx, June 2002.

47. Gu, Y. and B. Grossman, *SABUL - Simple Available Bandwidth Utilization Library/UDT (UDP-based Data Transfer Protocol)*: http://sourceforge.net/project/dataspace.

48. Foster, I., A. Roy, and V. Sander, *A Quality of Service Architecture that Combines Resource Reservation and Application Adaptation*, in *8th International Workshop on Quality of Service.* 2000.

49. Sander, V., et al., *A Differentiated Services Implementation for High Performance TCP Flows.* Computer Networks, 2000. **34**: p. 915-929.

50. Sander, V., et al., *End-to-End Provision of Policy Information for Network QoS*, in *The Tenth IEEE Symposium on High Performance Distributed Computing*. 2001.

51. Allcock, W., *GridFTP: Protocol Extensions to FTP for the Grid*. Grid Forum Document, No. 20. April 2003.

52. Allcock, W., *GridFTP: Protocol Extensions to FTP for the Grid*. 2003: Grid Forum Document, No. 20.

53. Simeonidou, D., et al., *Optical Network Infrastructure for Grid*. 2004: Grid Forum Document, No. 36.

54. Czajkowski, K., et al., *Agreement-based Grid Service Management (OGSI-Agreement).* GWD-R draft-ggf-czajkowski-agreement-00, June 2003.

55. *OMNInet*: http://www.icair.org/omninet.

56. *http://www.iwire.org*.

57. *http://www.east.isi.edu/projects/DRAGON/*.

58. Blanchet, M., F. Parent, and B.S. Arnaud, *Optical BGP: InterAS Lightpath Provisioning*, in *IETF Network Working Group Report*. March 2001.

59. Mambretti, J., et al., *The Photonic TeraStream: Enabling Next Generation Applications Through Intelligent Optical Network at iGrid 2002.* Journal of Future Computer Systems, August 2003: p. 897-908.

60. Grossman, B., et al., *Experimental Studies Using Photonic Data Services at iGrid2002.* Journal of Future Computer Systems, August 2003: p. 945-956.

61. *The GLIF web site at http:/www.glif.is*.

.

62. *The Starlight project at Startap http://www.startap.net/starlight*.

63. *The SurfNet project at the Netherlands http://www.surfnet.nl*.

64. *The UKLight progect at http://www.uklight.ac.uk*.

65. *The TeraGrid: A Primer*: http://www.teragrid.org/about/TeraGrid-Primer-Sept-02.pdf.

66. Allcock, W., *GridFTP and Reliable File Transfer Service*, in *Globus World 2005*. 2005.

67. DeFanti, T., et al., *Optical switching middleware for the OptIPuter.* IEICE Trans. Commun., Aug 2003. **E86-B**: p. 2263-2272.

68. Hoang, D.B., et al., *DWDM-RAM: An Architecture for Data Intensive Services Enabled by Next Generation Dynamic Optical Networks*, in *IEEE Globecom*. 2004: Dallas.

69. *FAST: Fast active queue management*: http://netlab.caltech.edu/FAST/.

70. Foster, I., et al., *Grid Services for Distributed Systems Integration.* IEEE Computer, 2002. **35**(6): p. 37-46.

71. *G.872: Architecture of optical transport networks*. Nov. 2001: Telecommunication Standardization Sector (ITU-T) of thr International Telecommunication Union (ITU).

72. Nguyen, C., et al., *Implementation of a Quality of Service Feedback Loop on Programmable Routers*, in *IEEE International Conference on Networks (ICON2004)*. 2004.

73. Franco Travostino, R.K., Tal Lavian, Monga Inder, Bruce Schofield, *DRAC - Creation an Applications-aware Networks.* Nortel Technical Journal, 2005. **1**(1).

74. *Web Services Description Language (WSDL) 1.1, World Wide Web Consortium*: http://www.w3.org/TR/wsdl.

75. Monga, I., B. Schofield, and F. Travostino, *EvaQ8 - Abrupt, high-throughput digital evacuations over agile optical networlks*, in *The 1st IEEE Workshop in Disaster Recovery Networks*. 2001.

76. Lavian, T., et al., *Edge device multi-unicasting for video streaming*, in *The 10th International Conference in Telecommunications, ICT2003*. 2003.