CHAPTER 1 INTRODUCTION TO GRID COMPUTING

CONTENTS

1.1 Grid Computing Concept
1.2 History of Distributed Computing
1.3 Computational Grid Applications
1.4 Grid Computing Infrastructure Development
   - Large-Scale US Grids
   - National Grids
   - Multi-National Grids
   - Campus Grids
1.5 Grid Computing Courses
1.6 Grid Computing Software Interface
1.7 Summary

Do not distribute.

Copyright B. Wilkinson

This material is the property of Professor Barry Wilkinson and is for the sole and exclusive use of the students enrolled in the Fall 2008 Grid computing course broadcast on the North Carolina Research and Education Network (NCREN) to universities across North Carolina.
CHAPTER 1
Introduction to Grid Computing

In this introductory chapter, we describe the concept of Grid computing, its history, pioneering landmark Grid computing projects and applications, and future directions. Access to Grid resources by users can be done through a command-line interface but more usually, it is done through a web-based portal. We describe our Grid computing course portal and registration process using the established Gridsphere portal toolkit as representative of Grid computing portal access. This would be a starting point for users and for a hands-on Grid computing course. How the Grid computing infrastructure works behind the portal will be described in subsequent chapters.

1.1 GRID COMPUTING CONCEPT

Grid computing uses distributed interconnected computers and resources collectively to achieve higher performance computing and resource sharing. It developed in the mid 1990’s with the growth of high speed networks and the Internet that allowed distributed computer systems to be readily interconnected. Grid computing has become one of the most important techniques high performance computing by providing resource sharing in science, technology, engineering, and business. By taking advantage of the Internet and high speed networks, geographically distributed computers can be used collectively for collaborative problem solving. In Grid computing, different organizations can supply the resources and personnel, and the Grid infrastructure can cross organizational and institutional boundaries. This concept has many benefits, including:
INTRODUCTION TO GRID COMPUTING

• Problems that could not be solved previously for humanity because of limited computing resources can now be tackled. Examples include for understanding the human genome or for searching for new drugs.

• Users can have access to far greater computing resources and expertise than available locally.

• Inter-disciplinary teams can be formed across different institutions and organizations to tackle problems that require the expertise of multiple disciplines.

• Specialized localized experimental equipment can be accessed remotely and collectively within a Grid infrastructure.

• Large collective databases can be created to hold vast amounts of data.

• Unused compute cycles can be harnessed at remote sites, achieving more efficient use of computers.

• Business processes can be re-implemented using Grid technology for dramatic cost saving.

Collaboration.  Perhaps the most important and differentiating feature of Grid computing is collaborative computing. Grid computing is about collaboration and resource sharing as much as it is about high performance computing. Certainly distributed computing existed before Grid computing as we shall review, but the easy prospect of developing teams of geographically distributed workers, a hallmark of Grid computing, became a reality with the development of the Internet. It is common practice to use the word Grid as a proper noun (i.e. G is capitalized) although that does not refer to a specific named Grid. Without qualification, the word Grid refers to a Grid computing infrastructures in general. There are many Grid infrastructures. Similarly the first letter of Web is capitalized.

Although very high performance Grid projects employ their own dedicated high speed interconnection networks, using the Internet to interconnect the distributed computers really makes Grid computing possible to all. The original driving force behind Grid computing was the same as behind the early development of networks that became the Internet - connecting computers at distributed sites for high performance. Grid computing came from the recognition that the Internet and Internet-type interconnections provide a unique opportunity for implementing a geographically distributing computing system. Some Grid projects involves computers spread across the globe and others are more localized, depending upon the goals of the project. For example, a project close to one extreme of vast geographical distances is a Grid computing demonstration at the Supercomputing 2003 conference in which 21 countries were represented and hundreds of computer systems (The Global Data-Intensive Grid Collaboration). A project close to the opposite extreme in terms of interconnectivity is the VisualGrid project (VisualGrid Project) at UNC-Charlotte which involved forming a grid at two sites, UNC-Charlotte and UNC-Asheville, North Carolina universities which are separated by about 130 miles.

The word “Grid” in Grid computing is often compared to the word “Grid” in
national electrical grids that supply electricity across countries and make electricity immediately available at outlets. The vision in using this word is that Grid computing will make high performance computing as easy to access. Some companies, such as IBM, have taken on this vision by offering (Grid) on demand computing, that is, providing access to Grid computing resources that are paid for when used. The term utility computing is also used for using Grid resources in similar way as utilities such as electrical, gas, and water are metered. Customers expect no interruption in electrical, gas, and water service (except through acts of God) and similarly on-demand or utility computing should provide high resilience to faults and security attacks.

Providing computing on demand for pay has led to Cloud computing in which companies offer a “cloud” of services as a business model. There are Grid computing research projects that focus on accounting, quality of service, service level agreements, such as the GridBus project (The Utility Grid Project).

Grid computing is a form of distributed computing and builds upon earlier concepts in distributed computing, but it also introduces new important aspects. As mentioned, the most important aspects are the involvement of teams and resource sharing. Grid computing often involves computers from multiple organizations and crosses organizational boundaries and enables the creation of distributed teams. The term virtual organization has been coined to describe groups of people, both geographically and organizationally distributed, working together on a problem, sharing computers and other resources such as databases and experimental equipment. Sometimes, the term virtual organization just refers to the people, but we will use the term to include the physical resources.

Crossing multiple administrative domains is another hallmark of larger Grid computing projects and introduces challenging technical and political challenges. The resources being shared are owned either by members of the virtual organization or donated by others and may be “leased”. They may also be used by others outside the virtual organization. There could be limited times of availability to a virtual organization, or a subset of the complete resources allocated. There could be multiple virtual organizations. Grid networks such as SURAGrid (SURAGrid) and TeraGrid (The TeraGrid project) have been established to support multiple Grid projects. In such Grid infrastructures, the use of resources are donated to the Grid for the common good although obviously strict usage policies are usually in place. Multiple virtual organizations can be formed that could use a subset of the resources. The key aspect of a virtual organization is its formation for a specific project.

The distributed team may be involved in a scientific experiment requiring data collection for experiment equipment. Not only are computers shared by the virtual organization formed for the project but also the experimental equipment, or the data that comes from sensors of the experimental equipment. Members of the team contribute to the overall mission of the project. A well-known example of a Grid computing project that involves experimental equipment and a wide distributed Grid is CERN grid that centers around the experimental Large Hadron Collider facility at European Center for Nuclear Research at CERN near Geneva Switzerland (World-wide LHC Computing Grid). In the search for new fundamental particles, the data from this experimental facility is sent to researchers around the world for analysis and collective research using Grid technology.
Standards. An extremely important aspect of Grid computing that has emerged is the use of open agreed standards. It is critically important to have agreed standards to encourage widespread adoption so that software can be developed by different groups and interoperate. Grid computing uses standard Internet standards and technology such as HTTP and SOAP. However, standards for Grid computing have to go much further than just the network protocols. Agreement must be in place for many aspects of Grid computing including security, data management, resource discovery, job submission etc. As we shall see, the underlying implementation technology used is now based upon Web services, the basis of recent web-based solutions. Later we will look at some actual Grid projects. Grid computing is evolving and so is the underlying technology. We will look at these standards in detail in subsequent chapters. Let us first look at the history of distributed computing.

Fosters’ check list. Ian Foster who is credited for the development of Grid computing, and sometimes called the father of Grid computing, proposed a simple checklist of aspects that are common to most true Grids (Foster 2002):

- No centralized Control
- Standard open protocols
- Non-trivial quality of service (QoS)

1.2 HISTORY OF DISTRIBUTED COMPUTING

Certainly one can go back a long way to trace the history of distributed computing. Types of distributed computing existed in the 1960s. Many people were interested in connecting computers together for high performance computing in the 1970s (including the author), and in particular forming multicomputer or multiprocessor systems. From connecting processors and computers together locally that began in earnest in the 1960s and 1970s, distributed computing now extends to connecting computers that are geographically distant. The development of the distributed computing technologies that underpin Grid computing were developed concurrently and rely upon each other. There are three concurrent interrelated paths:

- Networks
- Computing platforms
- Software techniques

Networks: Grid computing relies on distributed high performance networks. The history of such networks began in the 1960s with networks such as ARPNET introduced in 1969 providing 50 Kbs speeds. Network ports, addresses, and the TCP protocol appeared. The Ethernet network was developed in the early 1970’s. The Internet began to be formed in early 1980’s using the TCP/IP protocol. A brief review these underlying network technologies is given in Appendix A. During the 1980’s, the Internet grow at a phenomenal rate (1000 nodes in 1984, 10,000 nodes in

1 A brief review these underlying network technologies is given in Appendix A.
Network continued to improve and become more pervasive throughout the world. In the 1990, the Internet developed into the World-Wide Web. The browser and the HTML mark-up language was introduced. (Mark-up languages had been conceived earlier notably as a way of making documents machine readable.

**Computing Platforms:** Several projects using networked computers in laboratories for high performance computing began in the 1980s as laboratories of network computers became prevalent. A very important project in relation to Grid computing is called Condor (Condor), which started in the mid-1980s with goal to harness “unused” cycles of networked computers for high performance computing. In the Condor project, a collection of computers could be given over to remote access automatically when they were not been used locally. The collection of computers (called Condor pool) then formed a high-performance multicomputer. Multiple users could use such physically distributed computer systems. Some very important ideas were developed in Condor including matching the job with the available resources automatically using a description of the job and a description of the available resources. A job workflow could be described in which the output of one job could automatically fed into another job. Condor has been continually developed into the 2000s such that the distributed computers only need be networked and could be geographically distributed. It is very stable free software and widely used as a job scheduler. We will come back to its relation and application to Grid computing later.

In the 1990s, it was recognized that commodity computers (PC’s) provided the ideal cost-effective solution for constructing multi-computers, and the term cluster computing emerged. In cluster computing, computers are connected together through network connections, but the computer systems are physically close. Specialized high speed interconnections were developed for cluster computing. However, many chose to use Ethernet as a cost-effective solution although Ethernet was not developed for cluster computing applications and incurred a higher latency. The term Beowulf cluster was coined to describe using off-the-shelf computers and other commodity components to form a cluster, named after the Beowulf project at the NASA Goodard Space Flight Center started in 1993 (Sterling 2002). This project used Intel processors, the free Linux operating system and Ethernet connections. As clusters were being constructed, work was done of how to program them. The dominant programming paradigm for cluster computing is message passing in which information is passed between processes running on the computers in the form of messages. These messages are specified by the programmer using message-passing routines. The most notable library of message-passing routines was PVM (Sunderam 1990), which was started in the late 1980s and became the defacto standard in the early-mid 1990s. A true somewhat different but extensive standard definition for message passing libraries called MPI (Message Passing Interface) was subsequently established (Snir et al. 1998), which laid down what the routines do and how they are invoked but not the implementation. Several implementations were developed. Both PVM (now historical) and MPI routines are called from C/C++ or Fortran and course data to be

---

2 Mark-up languages including the XML language are described in Appendix B
passed via messages between processes running on the computers.

**Software Techniques:** Apart from the development of distributed computing platforms, software techniques were being developed to harness truly distributed systems. The remote procedure call (RPC) was conceived in the mid 1980’s as a way of invoking execution of a procedure on a remote computer as an extension of executing procedures locally. The remote procedure call was subsequently developed into object oriented versions in the 1990’s. One was CORBA (Common Request Broker Architecture) and another was the Java Method Invocation (RMI). The remote procedure call introduced the very important concept for Grid computing, that of a service registry which is used to locate remote services. We shall describe service registries in relation to discovering services in a Grid computing environment in Chapter 2. This includes the mechanism of discovering their method of invocation.

During the development of the world-wide web, the HTML mark-up language had been adopted for making Web pages machine readable to display purposes. Soon the Web page became more than simply displaying information; it became an interactive tool whereby information could be entered and processed at either the client side or the server side. The programming language Javascript was introduced in 1995 mostly for client-side processing, whereas other technologies were being developed for server-side processing such as ASP first released in 1996. The XML mark-up language was introduced in 2000 and the concept of Web services arrived. The Web service has its roots in the remote procedure call but based around the XML language, which provides interoperability. Web services were adopted into Grid computing soon after their introduction, as a flexible interoperable way of implementing the Grid infrastructure and potentially useful for Grid applications.

**Grid Computing:** The first large-scale Grid computing demonstration that involved geographically distributed computers and the start of Grid computing proper was the Information Wide-Area Year (I-WAY) demonstration at the Supercomputing 1995 conference (SC’95). This demonstration involved networked sites across the US. Seventeen supercomputer sites were involved including five DOE supercomputer centers, four NSF supercomputer centers, three NASA supercomputer sites, and other large computing sites. Ten existing ATM networks were interlinked with the assistance of several major network service providers. A focus was on virtual reality environments in such a networked system. Virtual reality components included an immersive 3D environment (CAVE™ Automatic Virtual Environment). Over 60 applications demonstrated in areas including astronomy and astrophysics, atmospheric science, biochemistry, molecular biology and structural biology, biological and medical imaging, chemistry, distributed computing, earth science, education, engineering, geometric modeling, material science, mathematics, microphysics and macrophysics, neuroscience, performance analysis, plasma physics, teleoperations/telepresence, and visualization (DeFanti 1996). Separate papers in the 1996 special issue of *International Journal of Supercomputer Applications* described nine of these applications.

I-Way was perhaps the largest collection of networked computing resources ever assembled for such a significant demonstration purpose at that time. It explored
many of the aspects of Grid computing now regarded as central such as security, job
submission and distributed resource scheduling. It came face-to-face with the “political
and technical constraints” that made it infeasible to provide single scheduler.
(DeFanti 1996). Each site had its own job scheduler, which have to be married
together. The I-way project also marked roughly the start of the Globus project
(Globus project), which developed defacto software for Grid computing. The Globus
project is led by Ian Foster, a co-developer of the I-way demonstration, and a founder
of Grid computing concept. The Globus project has developed a toolkit of middle-
ware software components for Grid computing infrastructure including for basic job
submission, security, and resource management. We will describe Globus a little
more detail later.

Although the Globus software has been widely adopted and is the basis of the
coursework described in this book, there are other early software infrastructure
projects. The Legion project also envisioned a distributed Grid computing environ-
ment. Legion was conceived in 1993 although work on the Legion software did not
begin in 1996 (Legion WorldWide Virtual Computer). Legion used an object based
approach to grid computing. Users could create objects in distant locations. The first
public release of Legion was at Supercomputing 97 conference in November 1997.
The work led to the Grid computing company and software called Avaki in 1999. The
company was subsequently taken over by Sybase Inc. UNICORE (UNiform Interface
to COmputing REsources) is a European grid computing project initially funded by
the German ministry for education and research (BMBF) and continued with other
European funding during the period 1997–2002 (Unicore). UNICORE is the basis of
several of the European efforts in Grid computing and elsewhere, including in Japan.
It has many similarities to Globus for example in its security model (X509 certificates
and certificate authorities, see Chapter 5) and a service based OGSA standard (see
Chapter 3) but is more complete solution than Globus and includes a graphical inter-
face. An example project using UNICORE is EUROGRID, a Grid computing testbed
developed in the period 2000–2004. (EUROGRID) An application project using
UNICORE and EUROGRID is OpenMolGRID Open Computing GRID for
Molecular Science and Engineering developed during the period 2002–2005 to
“speed up, automatize, and standardize the drug-design using Grid technology.”
OpenMolGRID)

With the development of Grid computing tools such as Globus and UNICORE,
a growing number of Grid projects began to develop applications. Originally, these
focused on computational applications. They can be categorized as:

- Computationally intensive
- Data intensive
- Experimental collaborative projects

The computationally intensive category is traditional high performance computing
addressing large problems. Sometimes it is not necessarily one big problem but a
problem that has to be solved repeatedly with different parameters (parameter sweep
problems). The data intensive category include computational problems but the
emphasis on large amounts of data to store and process. Experimental collaborative
projects often requiring collecting data from experimental apparatus, often very large amounts of data and studying and using this data.

The term e-Science was coined by John Taylor, the Director General of the United Kingdom's Office of Science and Technology in 1999 to describe conducting such research using distributed networks and resources, i.e. using Grid computing infrastructures (Wikipedia entry: E-science). Another more recent European term is e-Infrastructure, which refers to creating a Grid-like research infrastructure.

The potential of Grid computing was soon recognized by the business community for so-called e-Business applications to improve business models and practices, sharing corporate computing resources and databases and commercialisation of the technology through on-demand computing as mentioned early (Wikipedia entry: E-business). For e-Business applications the driving motive was reduction of costs whereas for e-Science applications, the driving motive was obtaining research results. That is not to say cost was not a factor in e-Science Grid computing. Large-scale research has very high costs and Grid computing offers distributed efforts and cost sharing of resources. There are projects that focus upon accounting such as GridBus mentioned earlier.

Figure 1.1 shows the time lines for computing platforms, underlying software techniques, and networks as we have been discussing. Some see Grid computing as an extension of cluster computing and it is true in the development of high performance computing, Grid computing has followed on from cluster computing in connecting computers together to form a multi-computer platform but Grid computing offers much more. We will take the approach of describing Grid computing as involving geographically distributed sites. The term cluster computing is limited for computing using computers interconnected locally to form a computing resource where the communication is invoked generally through explicit message passing routines within user programs. There is certainly a fine line in the continuum of interconnected computers from locally interconnected computers in a small room, through interconnected systems in a large computer room, then in multiple rooms and in different departments within a company, through to computers interconnected on the Internet in one area, in one country and across the world. The early hype of grid computing and marketing ploys in the late 1990 and early 2000s caused some to call configurations Grid computing when they were just large computational clusters or they were laboratory computers whose idle cycles are being used.

One classification that embodies the collaborative feature of Grid computing is:

- Enterprise Grids - Grids formed within an organization for collaboration
- Partner Grids - Grids set up between collaborative organizations or institutions

Enterprise Grid still might cross administrative domains of departments and requires departments to share their resources. Some of the key features we regard as indicative of Grid computing are:

- Shared multi-owner computing resources
- Used Grid computing software such as Globus, with security and cross-man-
We will describe Grid computing software such as Globus later but such software provides the tools to bring together geographically distributed computers owned by others together.

Grid Computing versus Cluster Computing. It is important not to think of traditional Grid computing simply as large cluster because the potential and challenges are different. Courses on Grid computing and on cluster computing are quite different. In cluster computing one learns about message passing programming using tools such as MPI, and shared memory programming using threads and OpenMP, given that most computers in a cluster today are now multi-core shared memory systems. Network security in cluster computing is not a big issue that the user usually needs to be directly concerned with. Usually an ssh connection to the
cluster is sufficient. The user is logging onto a single compute resource.

In Grid computing one looks at how to manage and use the geographically distributed sites (distributed resources). Each site is typically a high performance cluster. Being distributed, one looks at distributing computing techniques such as Web services and Internet protocols and network security as well as how to actually take advantage of the distributed resource.

In Grid computing, the user needs to be able to access multiple distributed resources. Of course, there are things in common with both Grid computing and cluster computing. Both are hands-on with programming experiences. Both require job schedulers to place jobs onto the best platform. A Grid computing scheduler has to manage the geographically disturbed resources owned by others and typically interacts with local cluster job schedulers.

**Grid Computing verse Cloud Computing.** Commercialization of Grid computing is driven by a business model that will make profits. The first widely publicized attempt was on-demand and utility computing in the early 2000’s which attempted to sell computer time on a Grid platform constructed using Grid technologies such as Globus. More recently Cloud computing is a business model in which services are provided on servers that can be accessed through the Internet, the “cloud” referring to the servers, as illustrated in Figure 1.2.

The common thread between Grid computing and cloud computing is the use of the Internet to access the resources. Cloud computing is driven by the widespread access that the Internet provides and newer Internet technologies. However cloud computing is quite distinct from the original purpose of Grid computing. Whereas Grid computing focuses on collaborative and distributed shared resources, cloud computing concentrates upon placing resources in one place for users to access and share. The technology for cloud computing emphases the use of services (software as a service, SaaS), and possibly the use of virtualization (the process of separating the
particular user’s software environment from the underlying hardware, which can take different forms but provides users with their own environment abstracted from the hardware.)

A number of companies entered the cloud computing space in the mid-late 2000’s. IBM was an early promoter of on-demand Grid computing in the early 2000’s and moved into cloud computing in a significant way, opening “cloud computing centers” in Dublin, Ireland in March, 2008 (the first cloud center in Europe), and in Amsterdam, the Netherlands, Beijing, China and Johannesburg, South Africa in June 2008. Other major players include Amazon and Google with their massive number of servers available. Amazon has the Amazon Elastic Compute Cloud (Amazon E2) project for users to buy time and resources through Web services and virtualization. The cloud computing business model is one step further than simply renting servers to clients, which became popular in the early-mid 2000’s with many start-up companies.

1.3 COMPUTATIONAL GRID APPLICATIONS

By the early 2000s, there were a very large number of computational Grid projects around the world, many more than we can describe here. They spans all science and engineering disciplines. There are Grid projects in areas such as:

- Biomedical research
- Industrial research
- Engineering research
- Studies in high energy Physics
- Studies in Chemistry

A representative large scale grid computing project involving experimental equipment mentioned earlier is the Large Hadron Collider experimental facility being constructed at CERN for research into particle Physics mentioned earlier. This project was fully operation with data being processed from the collider in 2007.

Another representative large scale grid computing project involving experimental equipment includes the NSF Network for Earthquake Engineering Simulation (NEES) for research to reduce vulnerabilities to catastrophic earthquakes. The work is also applicable to affects of other natural disasters and from terrorism. This large scale collaborative project brought together experimental networked facilities across the US interconnected using very high speed Internet 2 connections. It was funded by the National Science Foundation for ten years (2004-2010), and involves 15 universities (Cornell University, Lehigh University, Oregon State University, Rensselaer Polytechnic Institute, University at Buffalo, University of California, Berkeley, University of California, Davis, University of California, Los Angeles, University of California, San Diego, University of California, Santa Barbara, University of Colorado at Boulder, University of Illinois at Urbana-Champaign, University of Min-
Sites were provided with new or enhanced experimental apparatus for research into earthquakes (e.g. shaking tables, Tsunami wave basin, Geotechnical centrifuges), and included field monitoring. Sites also provided distributed computing resources. The work included the unusual aspect of merging together the measurements from physical apparatus such as shaking tables with computing models performed elsewhere to study the effects of earthquakes where the shaking table or computer model alone cannot represent the complete environment. More information on this project can be found at the NEES home page.

Another project focusing upon earth sciences is the Earth System Grid for climate modeling and research. This project is concerned with providing the climate research community access to climate data and as such can be classified as a data-intensive grid application. The project was funded by US Department of Energy. The first phase of this project (Earth System Grid I) in the 200-2001 period was pilot project, which was continued with a follow-up 5-year project called Earth System Grid II project (2001-2006) and involved Argonne National Laboratory, Lawrence Berkeley National Laboratory, Los Alamos National laboratory, National Center for Atmospheric Research, Oak Ridge National Laboratory, and University of South California (Earth System Grid). The ESG II data Grid connected various very large climate data repositories and developed Grid computing solutions to access to these data repositories to enable researchers to perform climate research. The Grid technologies including identifying, transferring at very high speed or replicating where appropriate very large amounts of data, security aspects, and ease of access through a grid portal. We will discuss these Grid technologies later in the book. The project managed terabytes of data, in a discipline where the amount of data is growing at a very fast pace with the data coming from Earth observing satellites and other sources. Details of the project can be found in the ESG II final report (Earth System Grid II).

A grid computing project in the medical arena is the UK eDiaMoND project conducted over the period 2002-2005 (EDiaMoNd Grid computing project). The objective of this project was to build a national database of mammographic images (digital Mammography) to aid screening and diagnosis of breast cancer. The project could be categorized as data-intensive in that medical images are stored and transferred to sites, but as most data-intensive grid projects, it also includes experimental equipment for obtaining the data (images in this case). The collaborating organizations were Churchill Hospital Oxford, Edinburgh Ardmillan Hospital, Guy's Hospital, London, IBM, King's College London, Mirada Solutions, Oxford e-Science Centre, School of Informatics (University of Edinburgh), St Georges Hospital, University College London, University of Oxford. A statement by the British prime minister Tony Blair in May 2002:

"The emerging field of e-science should transform this kind of work. It's significant that the UK is the first country to develop a national e-science grid, which intends to make access to computing power, scientific data repositories and experimental facilities as easy as the web makes access to information. One of the pilot e-science projects is to develop a digital mammographic archive, together with an intelligent medical decision
support system for breast cancer diagnosis and treatment. An individual hospital will not have supercomputing facilities, but through the grid it could buy the time it needs. So the surgeon in the operating room will be able to pull up a high-resolution mammogram to identify exactly where the tumor can be found." (124 words - check if permission needed)

describes the goals of this projects succinctly. Note his use of the key aspects of many Grid computing, “computing power,” “data repositories,” and “experimental equipment.” An aspect of this project, in common with other medical Grid computing projects involving human data, are legal and ethical considerations of maintaining confidentiality of personnel records.

There many more Grid projects. A key aspect of many of the computational grid projects are:

- State-of-the-art interconnection networks.
- Sharing resources.
- Community of scientists.

Resources can be much more than just computers. They can be:

- Storage
- Sensors for experiments at particular sites in the grid
- Application Software
- Databases

Let us reiterate that Grid computing is about collaborating and resource sharing as much as it is about high performance computing.

1.4 GRID COMPUTING INFRASTRUCTURE DEVELOPMENT

Grid computing application projects can create their own Grid, that is, a network of resources for that particular project. But just as the Internet provides a universal communication platform, Grid networks can be created to support many future projects and not tied to particular project or application. Grid networks have been set up at a local level, national level, and international level throughout the world to promote Grid computing.

1.4.1 Large-Scale US Grids

One of the most notable high speed networks constructed for Grid computing is the TeraGrid, which was funded by the National Science Foundation in 2001 initially to link five supercomputer centers. Hubs were established at Chicago and Los Angeles and interconnected using a 40 Gigabit/sec optical backplane network. The five centers were connected to one of the hubs using 30 Gigabit/sec connections:

- Argonne National Laboratory (ANL) (Chicago hub)
State-of-the-art optical lines could reach 10 Gigabit/sec in the early 2000s, and four such lines were used to achieve 40 Gigabit/sec. Three lines were used to achieve 30 Gigabit/sec.

TeraGrid was further funded by NSF for the period 2005-2010 and has developed into a platform for a wide range of Grid applications and is described as “the world’s largest, most comprehensive distributed cyberinfrastructure for open scientific research.” (TeraGrid). The 2008 configuration of TeraGrid is shown in Figure 1.3 As of 2008, TeraGrid consists to eleven resource provider sites, each with very significant compute data storage resources: Indiana University, the Louisiana Optical Network Initiative, the National Center for Atmospheric Research, National Center for Supercomputing Applications, the National Institute for Computational Sciences, Oak Ridge National Laboratory, Pittsburgh Supercomputing Center, Purdue University, San Diego Supercomputer Center, Texas Advanced Computing Center, and University of Chicago/Argonne National Laboratory, up from nine resource provider sites in 2007. As of 2008, there were hubs at Chicago Denver, Los Angeles, and Atlanta. Resource providers make connections to a hub with 10+ Gigabit/sec connections.

TeraGrid provides an open access for scientific research. Users make requests for an allocation reminiscent to getting access on a single supercomputer in the 1980’s and 1990, only now the users have access to broad range of extremely powerful
Open Science Grid (OSG) is another large scale grid computing infrastructure initiative, funded by the US National Science Foundation and US Department of Energy’s Office of Science (Open Science Grid). There is a very large number of participants in Open Science Grid, too long to list here. Consortium members have interests in particle and nuclear physics, astrophysics, bioinformatics, gravitational-wave science, and Computer Science aspects of grid computing. Multiple virtual organizations exist across the Grid. They exist for specific groups. In addition, a general purpose virtual organization group is provided for those who want to use OSG for individual or small group research. New members can contribute resources or even connect their own Grid to OSG, spreading out the size of the grid infrastructure. OSG provide “grid schools” for grid computing education and training.

The Southeastern Universities Research Association (SURA) established the SURAGrid, a collaborative venture between universities to provide a Grid computing facility. This participates have been growing steadily of the last few years now with a very large number of participant institutions across the US, mainly the South East, see Figure 1.4. SURAGrid is not focussed on any application domain at all and has applications that include storm surge modeling, multiple genome alignment, simulation-optimization for threat management in urban water systems, bioelectric simulator for whole body tissue, dynamics BLAST, and petroleum simulation. It also has interest in Grid computing education. (More on Grid computing education later.)

For infrastructure projects such as OSG and SURAGrid, which can take on new members working on new and distinct projects, the joining mechanism and gover-
nance policies must be easy and uniform. New members need to know the required 
software they need (the software stack) and the hardware. Obtaining and installing 
the software should be easy. Grid computing software is notoriously difficult to 
maintain because of its immaturity. Accounts need to be organized in an efficient 
manner. Accounts usually need to be provided on every resource a user wishes to 
access, either individual accounts or a group account. Infrastructure projects such as 
SURAGrid provide a centralized database of information to simplify account set-up. 
Security in all grid projects using the Internet use Internet security mechanisms, (cer-
tificates, etc.). Such security mechanisms require certificate authorities and it is still 
an open research problem the best way to organize a dynamically growing Grid infra-
structure with multiple organizations. SURAGrid accommodates institutions main-
taining their own certificate authority, which is cross-certified with a central bridge 
certificate authority. (More on security in Chapters 4 and 5.) Generally access to Grid 
resources is through a Grid portal (a Web-based user interface), and this portal will 
display the current status of available resources, but this requires all resources to 
communicate with the portal behind the scenes.

Physical grid infrastructures centers around high speed networks, which are 
being deployed everywhere. Many states in the US have deployed state-of-the-art 
high-speed fiber optic networks that are the basis for high performance Grid comput-
ing. Examples include the LONI network (Louisiana Optical Network Initiative) 
which boasts a 870 Gb/s aggregate transport capacity with connections operating 
between 20 and 60 Gb/s in 2006, the Florida Lambda Rail and the North Carolina 
NCREN network.

1.4.2 National Grids

Many countries have initiated national Grid computing projects around their high 
speed networks. The UK e-Science program began in November 2000 with £98 
million 3-year funding for new UK e-Science program. Funded quickly increased to 
£120 million, with £75 million devoted to large scale pilot projects in science and 
engineering and £35 million for the so-called “core e-Science program” that focussed 
on Grid middleware in collaboration with industry. This led to the formation of the 
UK e-Science Grid. Nine e-Science centers were created across the country. These 
centers (Southhampton, London, Cardiff, Oxford, Cambridge, Manchester, Newcas-
tle, Edinburgh/Glasgow, and Belfast.) and a couple of other sites/laboratories (a site 
Hinxton, and the Rutherford and Daresbury Laboratory) were connected together to 
form original UK e-Science grid as illustrated in Figure 1.5. The project was 
described in a paper by Hey and Trefethen (Hey and Trefethen 2002). The network 
used the existing UK university network, which was upgraded to 10 Gps by 2002. 
Later, seven “Centers of Excellence” were added to the nine regional grid centers. 
The purpose of the centers of excellence was to add regional presence and add 
expertise and applications. Funding over the five years 2001-2006 was quoted at 
£250 million for more than 100 projects (Highlights from the UK e-Science 
program). A feature of the UK e-Science program was the use of a single certificate 
authority for issuing certificates. This certainly greatly simplifies the process of 
issuing certificates. Such an approach has not been adopted in the US.
A follow-up UK activity to the original UK e-Science Grid was the establishment of UK National Grid Service, founded in 2004 to provide distributed access to computational and database resources, with four core sites, the universities of Manchester, Oxford and Leeds, and Rutherford Appleton Laboratory. By 2008, it had grown to 16 sites. Access is free to any academic with a legitimate need.

During the period 2000-2005, many other countries have seen the need for a national Grid. Other national Grid networks include Grid-Ireland (Ireland), NorduGrid (Scandinavian grid), DutchGrid (Netherlands), PIONIER (Poland), ACI (France). A national Grid provides collective national computing resources to address major scientific and engineering problems and also problems of national interest such as studying or predicting earthquakes, storms, major environmental disasters, global warming, and terrorism.

### 1.4.3 Multi-National Grids

Also in the period 2000-2005, several efforts were started to create Grids that spanned across many countries. For example, ApGrid, a partnership for Grid computing in the Asia Pacific region involved Australia, Canada, China, Hong Kong, India, Japan, Malaysia, New Zealand, Philippines, Singapore, South Korea, Taiwan, Thailand, USA, and Vietnam.

There have been several initiatives for European countries to collaborate in forming Grid-like infrastructures to share compute resources funded by European programs. For example, the DEISA (Distributed European Infrastructure for Supercomputing Applications) project to connecting major supercomputing facilities
across Europe, as illustrated in Figure 1.6. DEISA has the unique aspect of providing a global file systems that eliminates the need to move input files to the location of the executable and moving output files back to the user (so-called input and output staging). The DEISA-1 project was from 2004 - 2008. DEISA-2 started in 2008 with funding of 12,237,000 EUR (DEISA) to continue to 2011, with the partners: Barcelona Supercomputing Centre Spain (BSC), Consortio Interuniversitario per il Calcolo Automatico Italy (CINECA), Finnish Information Technology Centre for Science Finland (CSC), University of Edinburgh and CCLRC UK (EPCC), European Centre for Medium-Range Weather Forecast UK (ECMWF), Research Centre Juelich Germany (FZJ), High Performance Computing Centre Stuttgart Germany (HLRS), Institut du Développement et des Ressources en Informatique Scientifique - CNRS France (IDRIS), Leibniz Rechenzentrum Munich Germany (LRZ), Rechenzentrum Garching of the Max Planck Society Germany (RZG), Dutch National High Performance Computing Netherlands (SARA), Kungliga Tekniska Högskolan Sweden (KTH), Swiss National Supercomputing Centre Switzerland (CSCS), Joint Supercomputer Center of the Russian Academy of Sciences Russia (JSCC).

The vision of a single universal international Grid such as the Internet/World Wide Web may never be achieved though. More likely is that Grids will connect to other Grids but will maintain their identity.

Figure 1.6  DEISA (2008)
1.4.4 Campus Grids

Educational institutions also recognized the advantages of Grid technology in the early 2000s for cost-effectively sharing computing resources and several institutions set up campus wide grids. Examples include the University of Houston Campus Grid, Oxford University campus Grid, OxGrid, University of Texas at Austin Grid UTGRid, University of Michigan MGrid, University of Florida campus research Grid, and University of Virginia Campus Grid. Since most campuses nowadays provide wireless connectivity, a wireless Grid is possible for harnessing wireless resources. Note that forming a Grid on a campus faces political hurdles especially if the resources are not originally under one control, i.e., are controlled by individual faculty perhaps through research grants for particular purposes. Everybody has to agree in the collective advantages of sharing resources. Some campus Grids only embody those computing resources that are collectively controlled by the universities’ computing services.

1.5 GRID COMPUTING COURSES

Grid computing first appeared in graduate Computer Science curriculum in the late 1990s and early 2000’s as special topics courses within a single university, usually at those universities that already had Grid computing research projects. Very few, if any, of these early courses involved more than one university in their teaching. The Grid computing platform used was also often limited. Grid computing entered the Computer Science undergraduate curriculum with the introduction of Grid computing courses in the 2003–2004 period. One of the first advanced undergraduate courses was a collaborative project between SUNY at Buffalo and SUNY College at Geneseo, which was funded by NSF in 2003 (Ramamurthy and Jayaraman 2003). NSF funded two further independent projects on undergraduate computing course development in 2004. In one project, investigators at the University of Arkansas and at Lewis and Clark College developed a collaborative undergraduate Grid computing course (Apon and Mache 2004). In the other project (our first Grid computing course project) investigators at UNC-Charlotte, Western Carolina University, and UNC-Wilmington developed a collaborative undergraduate Grid computing course. These courses were targeted towards computer science majors. Our Grid computing course was taught across the state of North Carolina starting in 2004, and repeated in 2005, 2007, and 2008 (Wilkinson and Ferner 2006, 2008). This course was one of the first at the senior undergraduate level/first year graduate level to use both geographically distributed computers (a physical Grid) and geographically distributed students. A total of fifteen institutions across North Carolina shown in Figure 1.7 participated in the offerings between Fall 2004 and Fall 2008. Over this period, first Globus version 2.4 was used, then Globus version 3.2 and finally Globus version 4.0.

Computer systems (clusters) were set up at different universities depending upon the actual participants. For example in Fall 2005, clusters were used at six sites as illustrated in Figure 1.8. The Microelectronics Center of North Carolina provided...
a cluster as a back-up facility in case there were problems with the other systems, but it was not actually needed. Each site had its own certificate authority for signing certificates of users at that site and those of users at sites without contributing computer systems (see Chapters 4 and 5).

The early undergraduate Grid computing courses generally took a bottom-up approach to Grid computing education starting with network protocols, client-server concepts, creating Web and Grid services, and then progressing through the underlying Grid computing middleware, security mechanisms, and job submission to a Grid platform, all using Linux a command-line interface. The UNC-Charlotte course moved away from a Linux command line interface as much as possible although for some tasks, a user/programmer might wish to do still require a command line interface. While command-line interfaces are still used to access Grid resources in some
grid courses, it is more desirable to have a web-base GUI interface, a so-called Grid Portal. Whether a command-line or a portal, the user has *single sign-on* for all Grid resources, that is, once logged in with a password, it is unnecessary to supply any passwords subsequently to reach any grid resources, local or distant. (The actual way this is done will be described in Chapters 4 and 5 on Security.) In the next section, we will describe a typical user interface and then continue in later chapters with the underlying mechanisms.

### 1.6 GRID COMPUTING SOFTWARE INTERFACE

The primary objective of Grid computing infrastructure (middleware) software is to make a seamless environment for users to be able to access distributed resources. Key aspects include:

- Secure envelop over all transactions
- Single sign-on - being able with a single log on procedure to access all available resources and run jobs without having to supply additional passwords or account information.
- Data management tools
- Information services providing characteristics of resources and their status (including dynamic load)
- APIs and services that enable applications themselves to take advantage of the Grid platform
- Convenient user interface

**Globus.** One of the most influential Grid middleware project is the Globus project mentioned earlier. Globus has gone though several versions and bug fixes as illustrated in Figure 1.9. Version 1.0.0 was released in 1998 and was subsequently

![Figure 1.9: Some Globus Toolkit versions (appropriate time line)](image-url)
updated with version 2.0 released in early 2002. Version 2 became widely adopted, especially versions 2.2 and 2.3 which were released later. Foster, Kesselman, Nick, and Tuecke introduced an overall Grid architecture called OGSA (Open Grid Service Architecture) in 2002, which called for a service approach to Grid components. The first Globus implementation of this architecture was Globus version 3. Version 3 had a short life because of the way these services were implemented using the now defunct OGSI (Open Grid Service Infrastructure) standard. Version 4.0 released in April 2005 after a number of pre-releases (3.9.x) from May 2004 onwards and implemented a new standard called WSRF (Web Service Resource Framework), which was more widely accepted. Subsequent releases include version 4.1.0 in June 2006, 4.1.1 in March 2007 and version 4.2.0 in July 2008. The Globus versioning scheme is <major>.<minor>.<incremental> where <major> is a complete redesign, <minor> might possibly include changes to API’s although infrequent, and <incremental> are generally bug fixes. Stable releases have even minor numbers.

The Globus toolkit has five major parts:

- Common run time
  - Libraries and services
- Security
  - Components to provide secure access
- Execution management
  - Executing, monitoring and management of jobs
- Data Management
  - Discovery access and transfer of data
- Information
  - Discovery and monitoring of resources and services

We shall fully describe the components of Globus and how it developed in later chapters, but let us introduce some central components here.

**Security.** First security is required. The distributed resources must be protected from unauthorized access. The Globus components for creating the security envelop is called GSI (Grid Security Infrastructure), which used public key cryptography. We shall describe in detail how security is implemented in Chapters 4 and 5. It requires each user to be authenticated (their identity vouched) which is done by each user having a so-called (digital) certificate, signed by a trusted certificate authority. This technique is the basic of Internet security. Users will also need to be able to give their authority to Grid components to act on their behalf. Users also generally require accounts on resources that they intend to use.

**Resource Discovery.** Next the user often needs to know information about the available Grid resources. The basic Globus component for this is called MDS (Monitoring and Discovery System). The users might access MDS to discover
the status of the compute resources. Resource discovery is still very primitive and in research, but the ideal is to be able to submit a job and the system find the best resources for that job based upon the job description and resource descriptions across the whole Grid. In practice, users often know what resources are there but not the dynamic load.

**Executing a Job.** Next the user typically would want to submit a job. The basic Globus component for running a job is GRAM (Globus or Grid Resource Allocation Management). It may be necessary beforehand to transfer files to the resources and afterwards to transfer files to other locations including back to the user. The user might use the data management component called GridFTP for that.

The above activities are illustrated in Figure 1.10. It is important to note that Globus is a “toolkit” of components and not a complete solution for Grid computing infrastructure nor was it ever intended to be. Other higher level components are needed in a sophisticated grid computing infrastructure. Issues not addressed in the basic Globus toolkit include account management.

**User interfaces.** Grid computing environments are mostly Linux-based and originally accessed through the command line. So once you have established your security credentials (Chapter 4 and Chapter 5) to run a job, one might issue the GRAM command:

```
globusrun-ws -submit -c prog1
```

where `prog1` is the executable of the job. Transferring files could be done with the GridFTP command such as:

```
globus-url-copy
gsiftp://www.coitgrid02.uncc.edu/~abw/proglout
```
where in this case the file \texttt{www.coit-grid02.uncc.edu/~abw/prog1out} is transferred to \texttt{home/abw/}. Transferring files can be also done in a more power way of specifying the job in a job description language, a topic of Chapter 5.

As one can see this is a very primitive way of interacting with the Grid resources. A more desirable way is to have a Web-based GUI interface, a so-called Grid portal or gateway. Perhaps the most successful Grid portal project is the Gridsphere portal project, which is discussed in Chapter 8 with other GUI interfaces.

The Grid portal used for the UNC-Charlotte course is shown in Figure 1.11. The Grid portal is hosted on one server (in this case coit-grid02.uncc.edu:8080/gridsphere), which can be reached from anywhere on the Internet. The course portal was based upon the Gridsphere grid portal toolkit (The OGCE Portal Toolkit), which adheres to the JSR 168portlet standard and can interface to the defacto-standard Globus toolkit. It allows customized portlets (windows within the portal selected by menu) to be created and deployed within the portal. In fact, later on the course, students created customized portlets as front-ends to applications. The layout of a Portlet are actually defined using simple HTML/Javascript or JSP (Javaserver pages) or similar technology. We will discuss creating customized portals in Chapter 8.

Before users can login do anything on a grid platform, they must have security credentials and accounts on the resources they wish to access. The PURSe (Portal-based User Registration Service) portal was incorporated into the portal as a portlet and reached by selecting the “Register” menu item from the main course portal page. Figure 1.12 shows the PURSE registration portlet. The user submits the required information (name, email address, institution, etc.) This information is then forwarded to the course system administrator to set up accounts and credentials. A
A series of exchanges occur with the user by email confirming their intentions as shown in Figure 1.13. Note that communication is required with system administrators of remote resources. It is difficult to fully automate the process without communication.
between administrators because, apart from the technical matters that need to be set up, approval is needed to use resources owned by others. A number of software projects and tools have focussed on the very important matter of account management. However, account management is still often a human-centered process.

Finally, once everything is in place, the user will be able to login to the grid portal. The user will see a number of tabs across the top (Grid Information, Proxy Management, File Management, Job submission etc.) which enable the user to perform many basic tasks.

**Grid Information Tab:** The grid information portlet will display the status of the various computing resources available (i.e. whether up or down, the current load, etc.). Figure 1.14 shows a typical grid information portlet. For this to work, communication is needed between the resources and the portal. The GPIR (Grid Portal Information Repository) service works with the portal to gather the information into a database. Resources send an XML document in clear text (without security) at intervals to update the information. We will describe XML later. Details on exactly how GRIP does this can be found in the GPIR home page (GPIR). Notice that the information can be both static information and dynamic information and Grid resource can be more than just computing resources, for example, data servers and visualization resources could be displayed.

**Proxy Management Tab:** In order to use many services on the Portal, you are required to have a proxy. Proxies are part of the Grid security infrastructure, which we will discuss later in Chapter 5. For now, it is sufficient to say that a proxy is an electronic document called a certificate that enables resources to be accessed on
the user’s behalf. Usually PURSe automatically loads a proxy for you when you log in. Figure 1.15 show a typical display when selecting the Proxy Management tab. The default lifetime of this proxy is shown as 2 hours. (Proxies have limited lifetimes for security reasons.) However if you are running a job that takes longer than 2 hours to complete, you will need to create a proxy with a longer lifetime. One can click the “Get New Proxy” tab to do this. This will require you to provide information such as the host name of the myProxy server, which is a credential management service used to manage security credentials (MyProxy Credential Management Service). Users still need to know something about the inner working of the Grid infrastructure even when interacting with a portal. Ideally, users should need to know the minimum about this.

**File Management Tab.** Once you have your proxy, you can do grid-related activities such as transfer files or submitting a job. Figure 1.16 shows a file management tool. From the File Management tab, it will be possible to display the contents of your directories, display directories on two resources, and transfer files between resources. The Grid version of FTP called GridFTP [54] has been developed for efficient high speed transfers. Grid FTP comes under data management and will be discussed with other data management tools in Chapter ???

**Job Submission Tab.** Under the job submission one has the choice of submitting an interactive job or a batch job to a grid resource. Figure 1.17 shows the batch job submission portlet.
Figure 1.16  File management tab

Figure 1.17  Batch job submission tab
1.7 SUMMARY

This chapter introduced the following:

- A brief history leading to Grid computing
- Grid computing infrastructure examples, local, national and international
- A brief introduction to the purpose of Grid computing infrastructure software
- Grid user interfaces

In the next chapter, we will cover Web services. Web services are the basic of much of the Grid infrastructure.

FURTHER READING

The following are suitable as reading assignments at this stage:

Two important seminal papers:


Links to these papers can be found at http://www.cs.uncc.edu/~abw/GridComputingBook/

Selection of book and book chapters:

computing worthy of reading.


von Laszewski wrote a nice paper describing the evolution of Grid computing from the perspective of a Globus team member (von Laszewski 2005).

**REFERENCES**

About UK e-Science program, http://www.rcuk.ac.uk/escience/default.htm
DEISA Distributed European Infrastructure for Supercomputing Applications. http://www.depisa.eu/
EDiaMoNd grid computing project. http://www.ediamond.ox.ac.uk/whatis.html
Foster, I. 2002 What is the Grid? A three point checklist, Grid today, 1: 6 (July).
GPIR. http://gridport.net/services/gpir/index.html
GridFTP. http://dev.globus.org/wiki/GridFTP
MGrid. http://www.mgrid.umich.edu/
MPICH. http://www-unix.mcs.anl.gov/mpi/mpiich/
National Science Foundation grant, “Introducing Grid Computing into the Undergraduate Curri-
culuma,” ref. DUE #0410667/0533334, PI: A. B. Wilkinson, co-PI’s M. Holliday and D. Lug-
Network for Earthquake Engineering Simulation (NEES). http://www.nees.org/About_NEES/
OpenMolGRID Open Computing GRID for Molecular Science and Engineering. ht-
 tp://www.openmolgrid.org/
OxGrid. http://www.mc.manchester.ac.uk/research/seminars/past_seminars/OxGrid.pdf
Ramamurthy, B., and B. Jayaraman. “Collaborative: A Multi-Tier Model for Adaptation of
Grid Technology to CS-based Undergraduate Curriculum,” National Science Foundation
Complete Reference, Vol 1 The MPI Core, Cambridge, MA: MIT Press.
Cambridge.
Sunderam, V. 1990. PVM: A Framework for Parallel Distributed Computing, Concurrency
TeraGrid. http://www.teragrid.org/about/
University of Houston Campus Grid. http://www.grid.uh.edu/
University of Virginia campus grid. http://vgrid.cs.virginia.edu/campus_wide_grid/main.htm-
land
UTGRid. http://www.utgrid.utexas.edu/
Utility Grid Project: Autonomic and Utility-Oriented Global Grids for Powering Emerging e-
10.1524/itit.2005.47.6.319
Part II. IEEE Distributed Systems Online, 7(6-7). http://www.cs.uncc.edu/~abw/pa-
Wilkinson, B., and C. Ferner. 2008. Towards a Top-Down Approach to Teaching an Under-
graduate Grid Computing Course. SIGCSE 2008 Technical Symp. on Computer Science Ed-

© 2008 by CRC Press LLC
SELF-ASSESSMENT QUESTIONS

The following questions are multiple choice questions. Unless otherwise noted, there is only one correct question for each question.

1. What is a virtual organization?
   a) An imaginary company.
   b) A web-based organization.
   c) A group of people geographically distributed that come together from different organizations to work on a Grid project.
   d) A group of people that come together to work on a virtual reality Grid project.

2. What is a remote procedure call?
   a) A procedure call that is not very friendly
   b) A procedure call which is executed on a remote computer
   c) A procedure called from a remote computer
   d) A procedure call made from one remote computer to another remote computer.

3. What is meant by the term cloud computing?
   a) Atmospheric Computing
   b) Computing using geographically distributed computers
   c) A central facility providing services and software applications
   d) A distributed facility providing services and software applications.

4. In addition to computers, which of the following resources can be shared on a Grid?
   a) Storage
   b) Application Software
   c) Specialized equipment (such as sensors)
   d) Databases
   e) All of the above

5. What is meant by “single sign-on”?
   a) Allowing only one person to sign onto a computer
   b) None of the other answers
   c) Not allowing a person to log onto a computer more than once in any one period
   d) A mechanism in which a user does not need to sign again to acquire additional resources.

6. Which of the following is not provided in Globus version 4?
   a) Execution management
   b) Accounting
   c) Network security
d) Resource discovery.

ASSIGNMENTS

A suitable assignment at this stage is to register and login to a Grid platform, and perform some simple tasks through the portal. One such assignment can be found at the textbook home page, http://www.cs.uncc.edu/~abw/GridComputing-Book/. The actual details of logging onto the Grid platform may differ, depending upon the Grid platform you are using, but afterwards, perform the tasks:

1-1 Execute the Linux command (program) `echo` with suitable arguments from the (interactive) job submission portlet, redirecting standard output to a file called `echo_output`. Go to the file transfer portlet, and find this file. Download the file to your computer and take a screenshot of its contents.

Note: The Linux program `echo` simply sends its command line arguments to standard output. This program comes with the standard Linux distribution. It’s full path is `/bin/echo`.

1-2 In this task, you are to execute your own program rather than a preexisting program. The standard features of the portal may not provide for compiling programs directly on a Grid resource. Hence, we will compile a Java program remotely to obtain a platform-independent class file which will be uploaded and executed on the Grid resource with a java virtual machine.

Write and compile a java program to compute 10! (factorial 10) on your own computer (or a lab computer). Upload the program onto a server and execute the program there. The path to java interpreter is typically `/usr/java/jdk1.5.0_08/bin/java`. This is the executable to be specified in the portlet. The arguments will include `-classpath` flag and the classpath, and the name of the java class file. Notice that you will need to specify `CLASSPATH`, which is your home directory.