# Building a Global Compute Grid—Two Examples Using the Sun™ ONE Grid Engine and the Globus Toolkit


*Charu Chaubal, Grid Computing Engineering*

*Radoslaw Rafinski, Sun Microsystems, Poland*



*Sun BluePrints™ OnLine—April, 2003*

**http://www.sun.com/blueprints**

**Sun Microsystems, Inc.**
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-2649-10
Revision 1.0, 5/1/03
Edition: April 2003

# Building a Global Compute Grid—Two Examples Using the Sun ONE Grid Engine and the Globus Toolkit

This Sun BluePrints™ article extends the series on compute grids. In the previous Sun BluePrints articles entitled "Introduction to the Cluster Grid" parts 1 and 2, the architecture and design of Sun's cluster grid was discussed (see next section for a description of the cluster grid). In this Sun BluePrints article, the global grid is tackled, first in general terms then by considering two case studies. In particular, we look at the compute grid and the integration of the campus or cluster level grids with global grid technologies.

While this document is technical, it should provide suitable introductory material for those interested in or considering building a global compute grid. References are given at the end of this document to where you can obtain further information about the software elements discussed.

## What is a Grid?

Grid software and middleware enhances the way that collections of geographically distributed resources can be shared. Ian Foster, a pioneer in grid computing, describes the grid as "... a flexible, secure, co-ordinated resource sharing among dynamic collections of individuals, institutions and resources.". While the term has traditionally related just to high performance computing, organizations such as Globus and Avaki have helped extend the scope so that the utility of the tools becomes apparent in other fields.

Sun divides grid computing into three logical levels of deployment: cluster grids, enterprise grids, and global grids. global grids are collections of enterprise and cluster grids as well as other geographically distributed resources, all of which have agreed upon global usage policies and protocols to enable resource sharing. Enterprise grids enable multiple projects or departments to share resources within an enterprise or campus and do not have to address the security issues associated with global grids. A cluster grid is the simplest form of a grid and provides a compute service at the local area network (LAN) level. The class of software at the heart of the cluster grid is the distributed resource management (DRM) system.

Furthermore, the term *grid* can describe different aspects of resource sharing. For example: compute, data, or access. In this Sun BluePrints article, we concentrate on the compute aspect of the grid.

# Overview of a Global Compute Grid

In this section, these basic questions are briefly addressed.

- What are the motivations for creating a grid?
- What are the major areas of difficulty that must be addressed?
- What are the currently available components of a global compute grid and how do they solve the issues?

The Globus project introduces the term *virtual organization* (VO) as a set of users in multiple network domains who wish to share some of their resources. The virtual organization might be (as is currently the norm) a group of academic institutions that wish to enhance resource sharing. The possible functional objectives of such resource sharing include:

The aggregation of compute power can bring a number of benefits:

- Increases the throughput of users' jobs by maximizing resource utilization
- Increases the range of complementary hardware available, for example, compute clusters, large shared memory servers
- Provides a *grid supercomputer* that can provide a platform for grand challenge applications
- Provides the tight integration of geographically and functionally disparate databases
- Provides the catering for a huge, dynamic data set and the processing there of.

The main pieces of a global compute grid are:

- Interface—This enables access to the grid by nonexpert users. Some straightforward interface usually based upon a web-portal must be created.

- Broker—This is automating job scheduling based upon the user's policies. Such policies could describe the user's priorities in terms of job requirements, available budget, time requirements and so on. The broker would use these policies when negotiating on the user's behalf for a resource on the grid.
- Security, data-management, job-management, and resource discovery—These are the key issues that have been addressed by the Globus project. They are described further in the discussion of the Globus Toolkit.
- Resource guarantees and accounting—These are areas of current intensive research activity, and links with the brokering technologies. In this Sun BluePrints article, the features of the DRM are used to help solve these issues at the cluster grid level.

In this article we describe the components of a simple compute grid where geographically dispersed compute and storage resources are brought together and presented to users as a unified resource. First, the general concepts are discussed and then two specific implementations are described. The middleware discussed and implemented in the two case studies is the Globus Toolkit version 2.0 or 2.2.

# Brief Description or the Globus Toolkit 2.0

The Globus Toolkit 2.0 is an open architecture, open source software toolkit developed by the Globus project. A full description of the Globus Toolkit can be found at the Globus web site. The next generation of the Globus Toolkit (GT3.0) will be available in alpha at the time of the release of this Sun BluePrints article.

FIGURE 1 shows the three *pillars* of GT2.0 and GT2.2, each of which rely upon the Globus Security Infrastructure (GSI) base layer. The following is a brief description of these components:

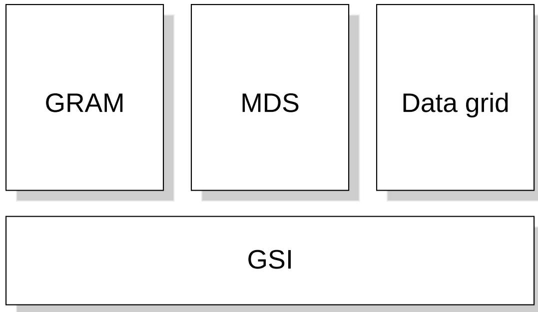


**FIGURE 1** Open Grid Services Architecture (OGSA).

## Globus Security Infrastructure (GSI)

The Globus Security Infrastructure provides the underlying security for the three pillars. GSI is based upon public key encryption. Each time any of the pillars are invoked to perform some transaction between Globus resources in that VO, GSI provides the mutual authentication between the hosts involved.

## Globus Resource Allocation Manager (GRAM)

GRAM provides the ability to submit and control jobs. GRAM includes the Resource Specification Language in which users can describe their job requirements. Once submitted, the job might be forked on some grid resource or might be passed on to a DRM such as Condor or grid engine.

## Monitoring and Discovery Services (MDS)

MDS provides the ability to discover the available resources on the grid. MDS implements a hierarchical structure of LDAP databases. Each grid resource can be configured to report to a local database and this information is aggregated in the higher-level databases. Grid users can query the high-level databases to discover up-to-date information on grid resources.

## Data Grid

Data grid provides the ability to transfer files efficiently (using grid FTP) and also provides a data replica management service.

There are a number of implementations of the programmer's interface to the Globus Toolkit. Commodity Grid (CoG) kits have been developed for Java™, Python, Web Services, and Perl among others. The Java CoG kit provides a J2SE 1.2.2 compliant interface for the Java environment.

# Distributed Resource Managers

As described in the previous Sun BluePrints article, the core of any enterprise or cluster grid is the distributed resource manager (DRM). Examples of DRMs are Sun™ ONE Grid Engine, Platform Computing's Load Sharing Facility, or Veridian's PBS Pro. In a global compute grid, it is often beneficial to take advantage of the features provided by the DRMs at the global grid level. Such features may include the ability to do the following:

- Create user sets whose access rights to the cluster grid might be controlled. This will strongly complement the limited authorization available through GT2.
- Provide resource guarantees to grid users
- Reserve portions of the local resource for local users
- Perform advanced reservation of compute resources

One of the key advantages to using DRM software is that it can simplify the implementation of the Globus layer above it. Specifically, where the underlying compute resources are heterogeneous in terms of operating platform, processor architecture, and memory, the DRM provides a virtualization of these resources, usually by means of the queue concept.

Different DRMs have different definitions of a queue, but essentially a queue and its associated attributes, represents the underlying compute resource to which jobs are submitted. If a VO chooses to implement a specific DRM at each of its cluster grids, then the concept of implementing a virtualization of all the cluster grids is relatively straightforward, despite the possibility that the underlying hardware might be quite heterogeneous. One simply aggregates all the queue information across the VO. Since the attributes of the queues will have a common definition across the VO, the interface to this grid could be designed to be analogous to that implemented at the campus level.

Integration of the DRM with GT2 can mean a number of things. Primarily it means that:

- There is an integration of the DRM with GRAM. This means that jobs submitted to Globus (using the Globus Resource Specification Language [RSL]) can be passed on to the DRMs. Evidently the key here is to provide a means of translation between RSL and the language understood by the DRM. These are implemented in Globus using GRAM job manager scripts.
- There is an integration with MDS. The use of a GRAM reporter allows information about a DRM to be gathered and published in the MDS. The reporter will run at each campus site periodically via `cron` and query the local DRM. This means that up-to-date queue information can be gathered across many cluster grids.

# Portal Software and Authentication

The portal solution may be split into two parts. First is the web-server and/or container which serves the pages. Examples include Sun™ ONE Portal Server, Tomcat/Apache, or uPortal. Second is the collection of Java servlets, web-services components, Java Beans, and so on. that make up the interface between the user and the Globus Toolkit, and runs within the server. The Grid Portal Development Kit is an example of a portal implementation that is interfaced with the Globus Toolkit.

# Global Grid Example One

## Introduction

In this section, the White Rose Grid (WRG) is discussed. The White Rose Grid, based in Yorkshire, UK, is a virtual organization comprising three universities: the Universities of Leeds, York, and Sheffield. There are four significant compute resources (cluster grids) each named after a white rose. Two cluster grids are sited at Leeds (Maxima and Snowdon) and one each at York (Pascali) and Sheffield (Titania).

The White Rose Grid is heterogeneous in terms of underlying hardware and operating platform. While Maxima, Pascali, and Titania are built from a combination of large symmetric memory Sun servers and storage/backup, Snowdon comprises a Linux/Intel based compute cluster interconnected with Myricom Myrinet.

The software architecture can be viewed as four independent Cluster Grids interconnected through global grid middleware, and accessible, optionally, through a portal interface. All the grid middleware implemented at White Rose is available in open source form.

FIGURE 2 shows the overall architecture. Each of the four WRG cluster grids have an installation of Sun™ ONE Grid Engine, Enterprise Edition. Globus Toolkit 2.0 provides the means to securely access each of the cluster grids through the portal.

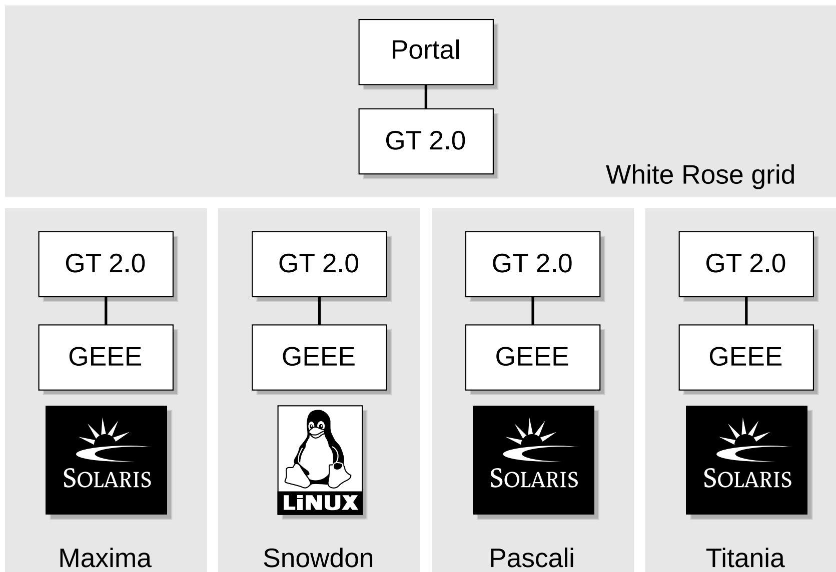


**FIGURE 2** White Rose Grid Overall Architecture

# Sun ONE Grid Engine, Enterprise Edition

Sun One Grid Engine, Enterprise Edition is installed at each of the four nodes, Maxima, Snowdon, Titania, and Pascali. The command line and GUI of Enterprise Edition is the main access point to each node for local users. The Enterprise Edition version of grid engine provides policy driven resource management at the node level. There are four policy types that might be implemented:

- **Share tree policy** – Enterprise Edition keeps track of how much usage users/projects have already received. At each scheduling interval, the scheduler adjusts all jobs' share of resources to ensure that users/groups and projects get very close to their allocated share of the system over the accumulation period.

- **Functional policy** – Functional scheduling, sometimes called priority scheduling, is a nonfeedback scheme (that is, no account taken of past usage) for determining a job's importance by its association with the submitting user/project/department.
- **Deadline policy** – Deadline scheduling ensures that a job is completed by a certain time by starting it early enough and giving it enough resources to finish on time.
- **Override policy** – Override scheduling allows the Enterprise Edition operator to dynamically adjust the relative importance of an individual job or of all the jobs associated with a user/department/project.

At White Rose, the Share Tree policy is used to manage the resource share allocation at each node. Users across the three universities are of two types: local users who have access only to the local facility and White Rose Grid users who are allowed access to any node in the WRG. Each White Rose Grid node administrator has allocated 25 percent of their node's compute resource for White Rose Grid users. The remaining 75 percent share can be allocated as required across the local academic groups and departments. The White Rose Grid administrators also agree upon the half-life associated with Sun One Grid Engine, Enterprise Edition so that past usage of the resources is taken into account consistently across the White Rose Grid.

# Globus

As shown in FIGURE 2, each White Rose Grid Cluster Grid hosts a Globus gatekeeper. The default job manager for each of these gatekeepers is set to grid engine using the existing scripts in the GT2.0 distribution. In order that the Globus job manager is able to submit jobs to the local DRM, it is necessary to ensure that the Globus gatekeeper server is a registered as a *submit host* at the local grid engine master node.

The Globus grid security file referenced by the gatekeeper servers includes the names of all WRG users. New users' grid identities must be distributed across the grid in order for them to be successfully authenticated. Additionally, at each site all WRG users are added to the user set associated with the WRG share of the Enterprise Edition controlled resource. This ensures that the sum usage by WRG users at any cluster grid does not exceed 25 percent.

# Portal Interface

The portal technology used at White Rose has been implemented using the Grid Portal Development Kit (GPDK). It has been designed as a web interface to Globus. GPDK uses Java Server Pages (JSP) and Java Beans and runs in Apache Tomcat, the open source web application server developed by Sun Microsystems. GPDK takes full advantage of the Java implementation of the Globus CoG toolkit.

GPDK Java Beans is responsible for the functionality of the portal and can be grouped into five categories; Security, User Profiles, Job Submission, File Transfer, and Information Services. For security, GPDK integrates with MyProxy. MyProxy enables the Portal server to interact with the MyProxy server to obtain delegated credentials in order to authenticate on the user's behalf.

The following development work has been done in order to port the publicly available GPDK to GT2.0:

- GPDK was modified to work with the updated MDS in GT2.0
- Information providers were written to enable grid engine queue information to be passed to the MDS. Grid users can query MDS to establish the state of the DRMs at each cluster grid.

As with many current Portal projects, the White Rose Grid uses the MyProxy Toolkit as the basis for security. FIGURE 3 shows that prior to interacting with the WRG, a user must securely pass a delegated credential to the portal server so that the portal can act upon that user's behalf subsequently. The MyProxy Toolkit enables this.

FIGURE 3 shows the event sequence up to job submission.

- Events 1-4) When users initially log on, the MyProxy Toolkit is invoked so that the portal server can securely access a proxy credential for that user.
- Events 5 and 6)The users can view the available resources and their dynamic properties via the portal. The Globus MDS pillar provides the GIIS, LDAP based hierarchical database which must be queried by the portal server.
- Event 7) Once users have determined the preferred resource, the job can be submitted. The job information is passed down to the selected cluster grid where the local Globus gatekeeper authenticates the users and passes the job information to Sun ONE Grid Engine, Enterprise Edition.
- Event 8) When the job begins, it might further use mechanisms implemented by the Globus Toolkit during the course of execution, such as retrieving up-to-date resource information, requesting and retrieving data sets from other campuses, and storing results in other locations as well. When the job finishes, users can view the results through the Grid Portal at their desktop.

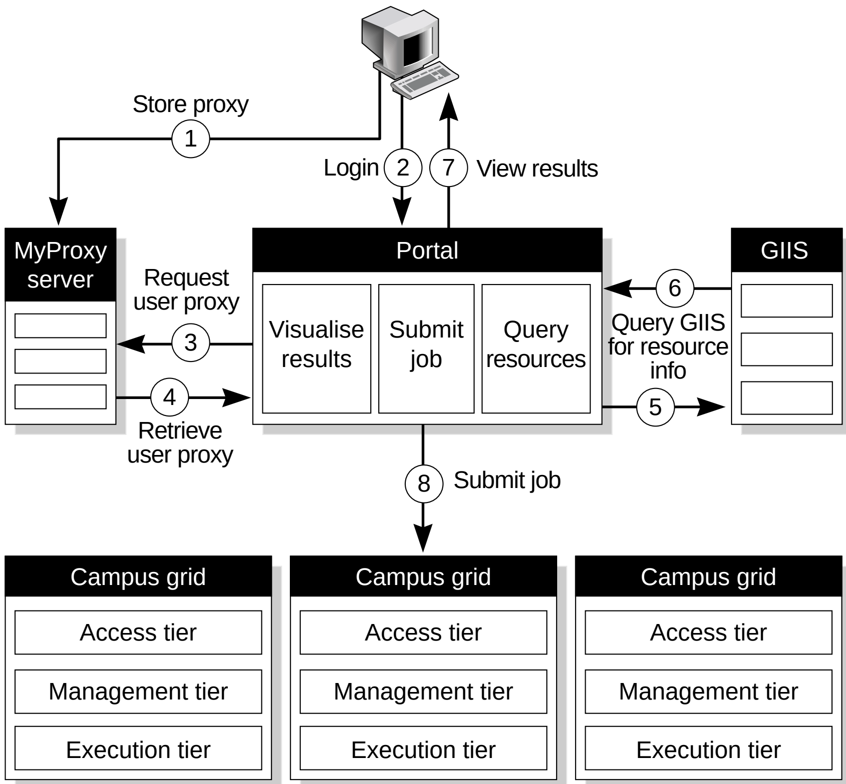


**FIGURE 3** Event Sequence

# Global Grid Example Two

## Introduction

The Poznan supercomputer project PROGRESS (Polish Research on Grid Environment for Sun Servers) involves two academic sites in Poland, Cracow and Poznan. The project was founded by the State Committee for Scientific Research.

Project partners are: Poznan Supercomputing and Networking Center; Academic Supercomputing Center of University of Mining and Metallurgy, Cracow; Technical University, Lodz, and Sun Microsystems Poland.

Currently, there are two cluster grids accessible through the PROGRESS portal: one Sun Fire 6800 (24 CPUs) in Cyfronet Cracow and two Sun Fire 6800s (24 CPUs) connected using Sun Fire™ Link sited at Poznan Supercomputing and Networking Center. The distance between the locations is about 400 km. Both locations also use Sun Fire V880s and Sun StorEdge™ 3910s as the hardware supporting the Distributed Data Management System discussed below. At the development stage, only large Sun SMP machines are used, but the architecture of the developed system allows the existing computing resources to be augmented by hardware from other vendors.

As in the White Rose implementation, Globus Toolkit 2.x has been implemented to provide the middleware functionality. Also, as in the White Rose case, Sun ONE Grid Engine, Enterprise Edition is installed to control each of the cluster grids. In this case, however, the portal interface has been built using Webservices elements based upon J2EE.

The main task of this project is to give a unified access to distributed computing recourses for the Polish scientific community.

Other aims are the following:

- Development of novel tools supporting grid-portal architecture (grid service broker, security, migrating desktop, and portal access to grid)
- Development and integration of data management and visualization modules
- Testing of tools and infrastructure using a set of bio-informatical applications (DNA assembling and prediction of protein secondary structure using Logical Analisys of Data)
- Enabling the grid-portal environment for other advanced applications (PIONIER program)

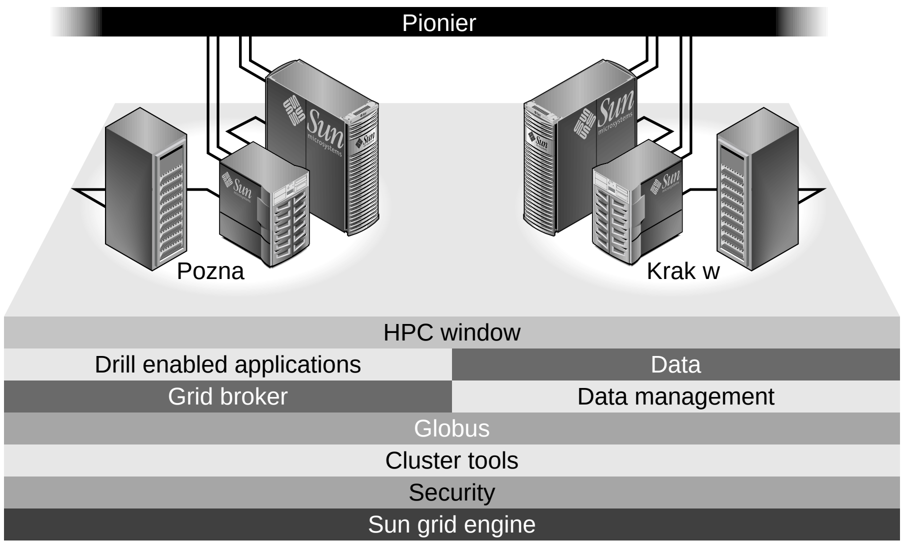


**FIGURE 4** Overview of PROGRESS Project

## System Modules

The PROGRESS architecture can be described in terms of its constituent modules. As well as using the Globus Toolkit, each of these modules provides a major piece of functionality for the PROGRESS grid. The four main modules are:

- Portal Environment
- Grid Service Provider
- Grid Infrastructure
- Data Management System.

FIGURE 5 shows how these modules interact in the PROGRESS architecture and shows the main communication protocols used between the modules.

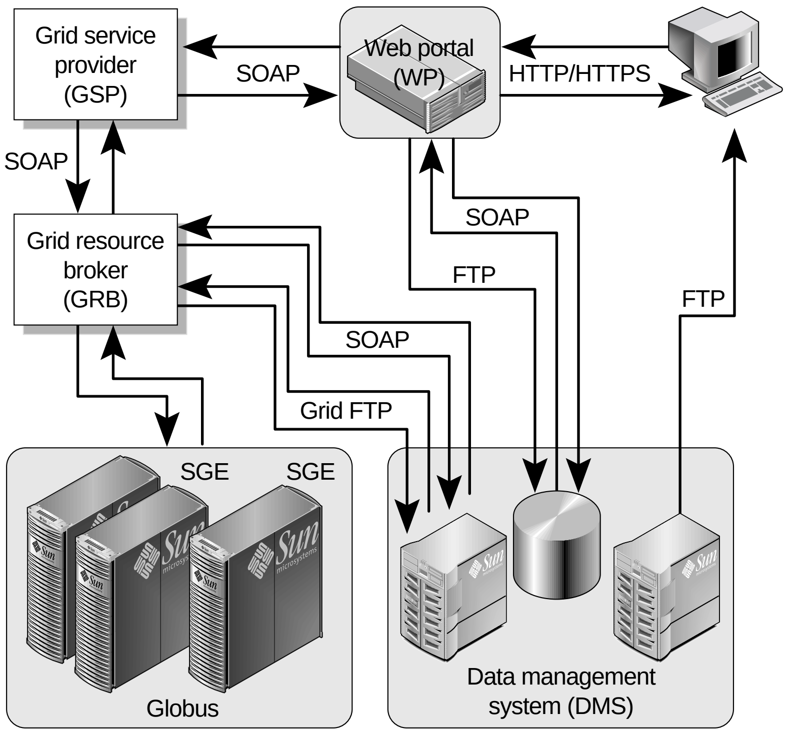


**FIGURE 5** PROGRESS Architecture

# Portal Environment

The main module of the Progress Portal Environment is the Grid Service Provider (GSP). It is a new layer introduced into the Grid Portal architecture by the PROGRESS research team. The GSP provides users with three main services: a job submission service (JS), an application management service (AM), and a provider management service (PM).

- The JS is responsible for managing creation of user jobs, their submission to the grid, and monitoring of their execution.
- The AM provides functions for storing information about applications available for running in the grid. One of its main features is the possibility of assisting application developers in adding new applications to the application factory.
- The PM allows the GSP administrator to keep up-to-date information on the services available within the provider.

The PROGRESS GSP services are accessible through two client interfaces: Web Portal (WP) and Migrating Desktop (MD). The Web Portal, which is deployed on the Sun ONE Portal Server 7.0, performs three functions:

- Grid job management: creating, building, submitting, monitoring execution, and analyzing results
- Application management
- Provider management

FIGURE 6 shows a typical user's view of the portal interface.

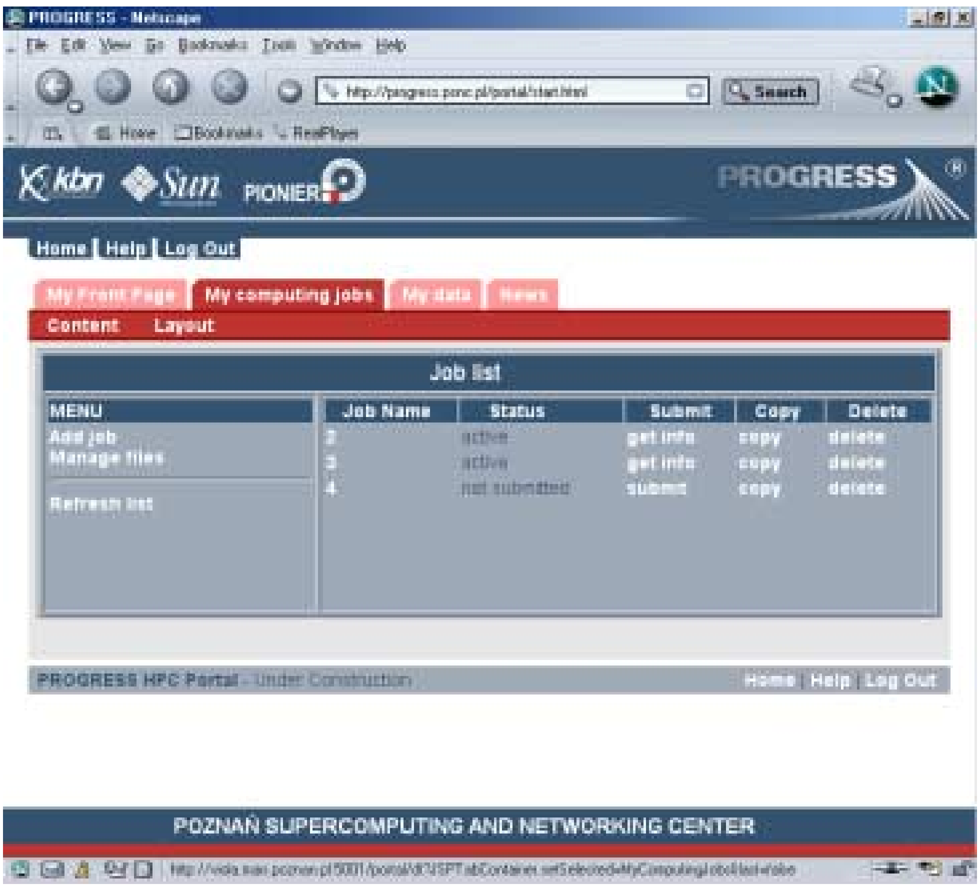


**FIGURE 6** PROGRESS Portal

The MD, which is a separate Java client application, provides user interface for grid job management and DMS file system management. Both user interfaces are installed on a Sun Fire 280R machine, which serves as the PROGRESS system front-end.

Additionally PROGRESS Portal gives access to services such as news services, calendar server, and messaging server (deployed on a Sun ONE calendar and messaging server).

## Grid Infrastructure

The Grid Resource Broker (GRB) is developed in the Poznan Supercomputing and Networking Center and enables execution of PROGRESS grid jobs in grid clusters. Clusters are managed by Sun ONE Grid Engine, Enterprise Edition software with Globus deployed on it. The GRB provides two interfaces: a CORBA interface and a web services interface. Grid job definitions are passed to the GRB in the form of an XRSL document and the GRB informs the JS about events connected with the execution of a job (start, failure, or success).

Sun ONE Grid Engine, Enterprise Edition is implemented at both sites as the local distributed resource manager. Sun ONE Grid Engine, Enterprise Edition provides policy driven resource management at the node level. The Share Tree policy is used to manage the resource share allocation at each node.

Two types of users can have access to resources:

- Local users, accessing compute resources through the grid engine GUI
- Portal users, accessing nodes using the PROGRESS portal or Migrating Desktop.

In the event of extensions to the PROGRESS grid, grid engine would also be used.

Each cluster grid also hosts Globus Gatekeeper.

## Data Management System (DMS)

PROGRESS grid jobs use the DMS to store the input and output files. The DMS provides a web services based data broker that handles all requests. The DMS is equipped with three data containers: the file system, the database system, and the tape storage system containers. A data file is referenced within the DMS with a universal object identifier, which allows for obtaining information on the location of the file.

Users can download or upload files using one of three possible protocols: FTP, GASS, or GridFTP.

# References

Home page fro the Globus Project

`www.globus.org`

Sun grid information

`wwws.sun.com/software/grid/`

Home page of the Grid Portal Development Kit

`http://doesciencegrid.org/projects/GPDK/`

Home page of the MyProxy project

`http://www.ncsa.uiuc.edu/Divisions/ACES/MyProxy/`

# About the Authors

Charu Chaubal has been an engineer in the Grid Computing Engineering group at Sun Microsystems, Inc. for the last two years. He currently works on prototyping and development of integrated grid solutions. He has also worked on customer installations and developed and delivered training courses on grid computing. Charu received a Bachelor of Science in Engineering from the University of Pennsylvania, and a Ph.D. from the University of California at Santa Barbara, where he studied the numerical modeling of complex fluids.

Radek Rafinski has been with Sun Microsystems, Poland, for three years. Radek is and SE specializing in the education sector.

# Ordering Sun Documents

The SunDocs[SM] program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

# Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`