Grid computing: What are the key components?

Contents:

High-level overview of Grid computing
Globus Project
Grid components: a high-level perspective
Job flow in a grid environment
Summary
Resources
About the author
Rate this article

Related content:

Develop a Grid information provider with the Monitoring and Discovery Service
Grid job submission using the Java CoG Kit
Querying the Grid with the Globus Toolkit Monitoring and Discovery Service
Transferring files with GridFTP
Use C++ classes and Globus Toolkit 2.4 to submit Grid jobs
Use C++ with Globus Toolkit 2.4 MDS to develop a job broker

Taking advantage of Grid computing for application enablement

Level: Introductory

Bart Jacob (bjacob@us.ibm.com)
ITSO Redbooks Project Leader, IBM
June 2003

Grid computing is gaining a lot of attention within the IT industry. Although it has been used within the academic and scientific community for some time, standards, enabling technologies, toolkits, and products are becoming available that allow businesses to use and reap the advantages of Grid computing. As with many emerging technologies, you will find almost as many definitions of Grid computing as people you ask. However, one of the most used toolkits for creating and managing a grid environment is the Globus Toolkit. In this article, we give you an overview of the key components that make up a Grid environment, and we present most of our information and concepts within the context of the Globus Toolkit.

High-level overview of Grid computing

The most common description of Grid computing includes an analogy to a power grid. When you plug an appliance or other object requiring electrical power into a receptacle, you expect that there is power of the correct voltage available, but the actual source of that power is not known. Your local utility company provides the interface into a complex network of generators and power sources and provides you with (in most cases) an acceptable quality of service for your energy demands. Rather than each house or neighborhood having to obtain and maintain its own generator of electricity, the power grid infrastructure provides a virtual generator. The generator is highly reliable and adapts to the power needs of the consumers based on their demand.

The vision of Grid computing is similar. Once the proper kind of infrastructure is in place, a user will have access to a virtual computer that is reliable and adaptable to the user's needs. This virtual computer will consist of many diverse computing resources. But these individual resources will not be visible to the user, just as the consumer of electric power is unaware of how their electricity is being generated. To reach this vision, there must be standards for Grid computing that will allow a secure and robust infrastructure to be built. Standards such as the Open Grid Services Architecture (OGSA) and tools such as those provided by the Globus

Toolkit provide the necessary framework. Initially, businesses will build their own infrastructures (what we might call intra-grids), but over time, these grids will become interconnected. This interconnection will be made possible by standards such as OGSA and the analogy of Grid computing to the power grid will become real.

Subscribe to the developerWorks newsletter

developerWorks Toolbox subscription

Also in the Grid computing zone:

Tools and products

Articles

Types of grids

Grid computing can be used in a variety of ways to address various kinds of application requirements. Often, grids are categorized by the type of solutions that they best address. The three primary types of grids are summarized below. Of course, there are no hard boundaries between these grid types and often grids may be a combination of two or more of these. However, as you consider developing applications that may run in a grid environment, remember that the type of grid environment that you will be using will affect many of your decisions.

Computational grid

A computational grid is focused on setting aside resources specifically for computing power. In this type of grid, most of the machines are high-performance servers.

Scavenging grid

A scavenging grid is most commonly used with large numbers of desktop machines. Machines are scavenged for available CPU cycles and other resources. Owners of the desktop machines are usually given control over when their resources are available to participate in the grid.

Data grid

A data grid is responsible for housing and providing access to data across multiple organizations. Users are not concerned with where this data is located as long as they have access to the data. For example, you may have two universities doing life science research, each with unique data. A data grid would allow them to share their data, manage the data, and manage security issues such as who has access to what data.

Another common distributed computing model that is often associated with or confused with Grid computing is peer-to-peer computing. In fact, some consider this another form of Grid computing. For a more detailed analysis and comparison of Grid computing and peer-to-peer computing, see the article *On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing*, by Ian Foster and Adriana Iamnitchi. (See Resources.)

Globus Project

The Globus Project (see Resources) is a joint effort on the part of researchers and developers from around the world that are focused on the concept of Grid computing. It's organized around four main activities:

- Research
- Software tools
- Testbeds
- Applications

Globus Toolkit V2.2

The Globus Toolkit V2.2 provides:

- A set of basic facilities needed for Grid computing:
  - Security: single sign-on, authentication, authorization, and secure data transfer
  - Resource Management: remote job submission and management
  - Data Management: secure and robust data movement
  - Information Services: directory services of available resources and their status

- Application Programming Interfaces (APIs) to the above facilities
- C bindings (header files) needed to build and compile programs

These elements are considered the core of the toolkit. In addition, other components are available that complement or build on top of these facilities. For instance, Globus provides a rapid development kit known as CoG (Commodity Grid) which supports technologies such as Java, Python, Web services, CORBA, and so on. The facilities provided by Globus can be used to build grids and grid-enabled applications today. Many such environments have been built. However, when building such an infrastructure that is suitable for use in business environments, there are other considerations that have not been fully addressed by the Globus Toolkit V2.2. For instance, you might need services such as life-cycle management, accounting, and charge back systems. Another consideration when building a grid environment today is the ability to interconnect with other grids in the future. To enable the interconnection between grids developed by different organizations that may be using different technologies requires standards to be put in place and adopted.

OGSA and Globus Toolkit V3

The Open Grid Services Architecture (OGSA) is an evolving standard for which there is much industry support. Globus Toolkit V3 (available now in beta) will be the reference implementation for OGSA. OGSA addresses both issues we discussed in the previous section. First, it changes the programming model to one that supports the concept of the various facilities becoming available as Web services. This will provide multiple benefits, including:

- A common and open standards-based set of ways to access various grid services using standards such as SOAP and XML.
- The ability to add and integrate additional services such as life cycle management in a seamless manner
- A standard way to find, identify, and utilize new grid services as they become available

In addition to benefits such as these, OGSA will provide for interoperability between grids that may have been built using different underlying toolkits. As mentioned, Globus Toolkit V3 will be the reference implementation for OGSA. Although the programming model will change, most of the actual APIs that are available with Globus Toolkit V2.2 will remain the same. Therefore, work done today to implement a grid environment and enable applications will not necessarily be lost.

OGSA and OGSI

OGSA defines a standard for the overall structure and services to be provided in grid environments. The Open Grid Services Interface (OGSI) specification is a companion standard that defines the interfaces and protocols that will be used between the various services in a grid environment. The OGSI is the standard that will provide the interoperability between grids designed using OGSA.

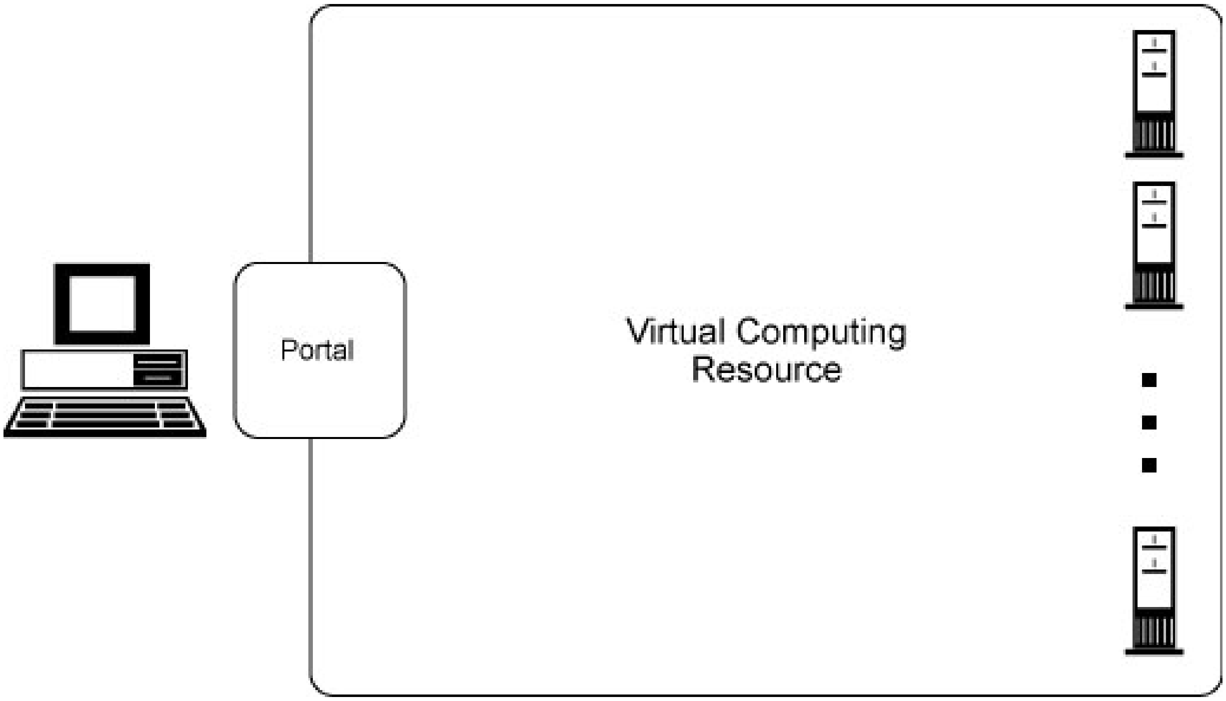
Grid components: a high-level perspective

In this section, we describe at a high level the primary components of a grid environment. Depending on the grid design and its expected use, some of these components may or may not be required, and in some cases they may be combined to form a hybrid component. However, understanding the roles of the components as we describe them here will help you understand the considerations when developing grid-enabled applications.

Portal/user interface

Just as a consumer sees the power grid as a receptacle in the wall, a grid user should not see all of the complexities of the computing grid. Although the user interface can come in many forms and be application-specific, for the purposes of our discussion, let's think of it as a portal. Most users today understand the concept of a Web portal, where their browser provides a single interface to access a wide variety of information sources. A grid portal provides the interface for a user to launch applications that will use the resources and services provided by the grid. From this perspective, the user sees the grid as a virtual computing resource just as the consumer of power sees the receptacle as an interface to a virtual generator.

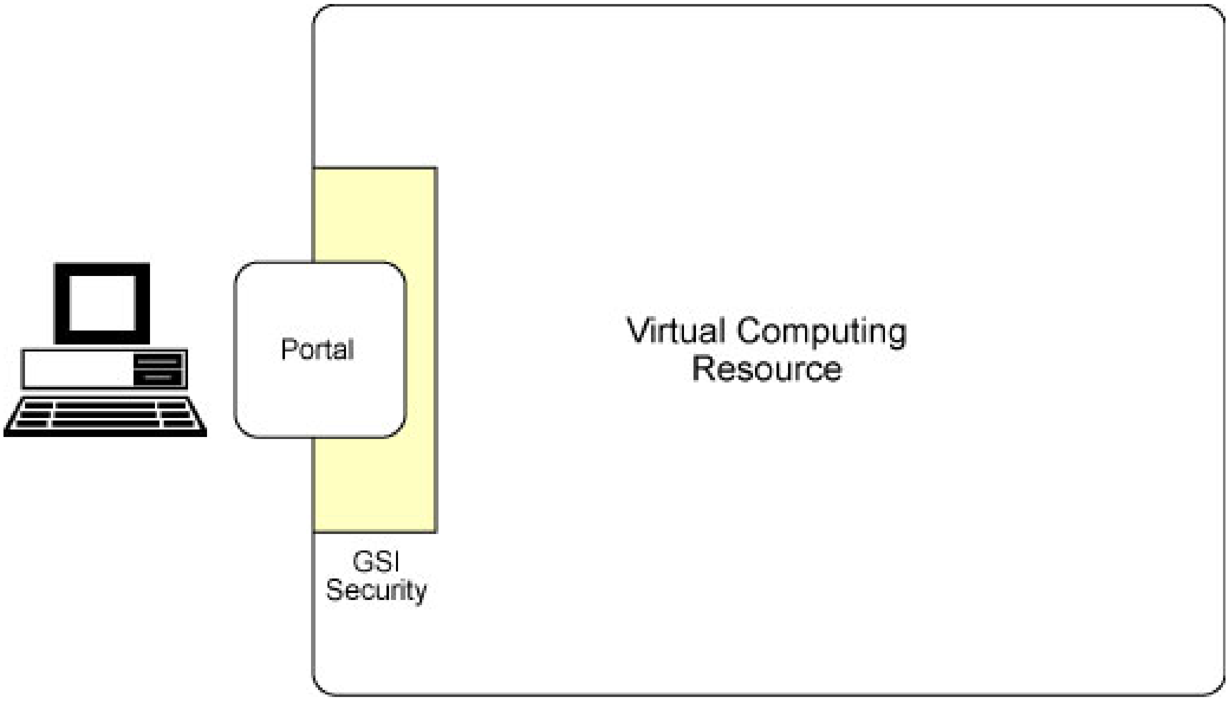
Figure 1. Possible user view of a grid



The current Globus Toolkit does not provide any services or tools to generate a portal, but this can be accomplished with tools such as WebSphere Portal and WebSphere Application Server.

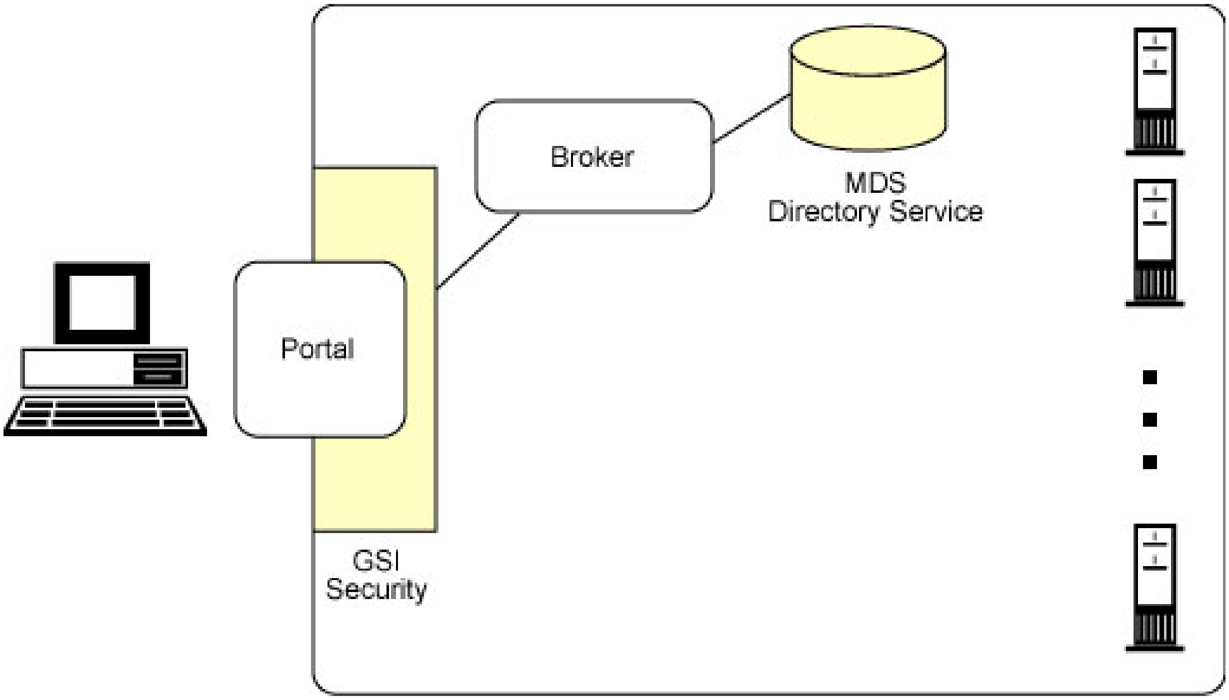
Security
A major requirement for Grid computing is security. At the base of any grid environment, there must be mechanisms to provide security, including authentication, authorization, data encryption, and so on. The Grid Security Infrastructure (GSI) component of the Globus Toolkit provides robust security mechanisms. The GSI includes an OpenSSL implementation. It also provides a single sign-on mechanism, so that once a user is authenticated, a proxy certificate is created and used when performing actions within the grid. When designing your grid environment, you may use the GSI sign-in to grant access to the portal, or you may have your own security for the portal. The portal will then be responsible for signing in to the grid, either using the user's credentials or using a generic set of credentials for all authorized users of the portal.

Figure 2. Security in a grid environment



Broker
Once authenticated, the user will be launching an application. Based on the application, and possibly on other parameters provided by the user, the next step is to identify the available and appropriate resources to use within the grid. This task could be carried out by a broker function. Although there is no broker implementation provided by Globus, there is an LDAP-based information service. This service is called the Grid Information Service (GIS), or more commonly the Monitoring and Discovery Service (MDS). This service provides information about the available resources within the grid and their status. A broker service could be developed that utilizes MDS.

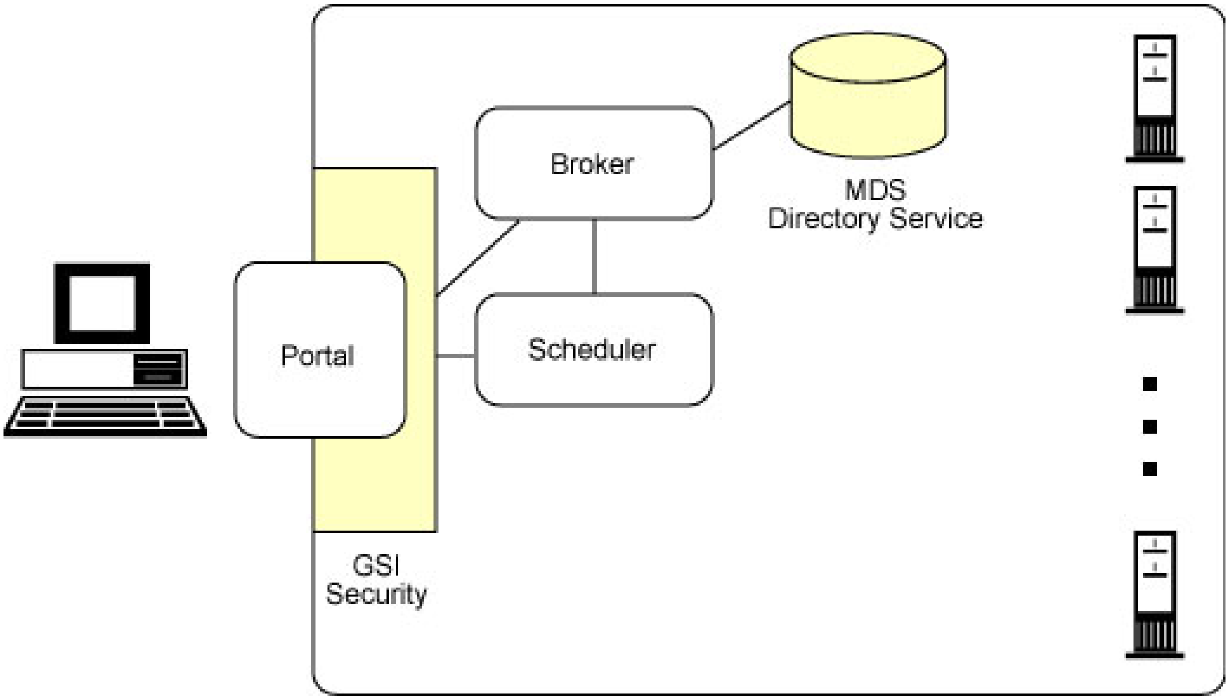
Figure 3. Broker service



Scheduler
Once the resources have been identified, the next logical step is to schedule the individual jobs to run on them. If a set of stand-alone jobs are to be executed with no interdependencies, then a specialized scheduler may not be required. However, if you want to reserve a specific resource or ensure that different

jobs within the application run concurrently (for instance, if they require inter-process communication), then a job scheduler should be used to coordinate the execution of the jobs. The Globus Toolkit does not include such a scheduler, but there are several schedulers available that have been tested with and can be used in a Globus grid environment. It should also be noted that there could be different levels of schedulers within a grid environment. For instance, a cluster could be represented as a single resource. The cluster may have its own scheduler to help manage the nodes it contains. A higher level scheduler (sometimes called a meta scheduler) might be used to schedule work to be done on a cluster, while the cluster's scheduler would handle the actual scheduling of work on the cluster's individual nodes.

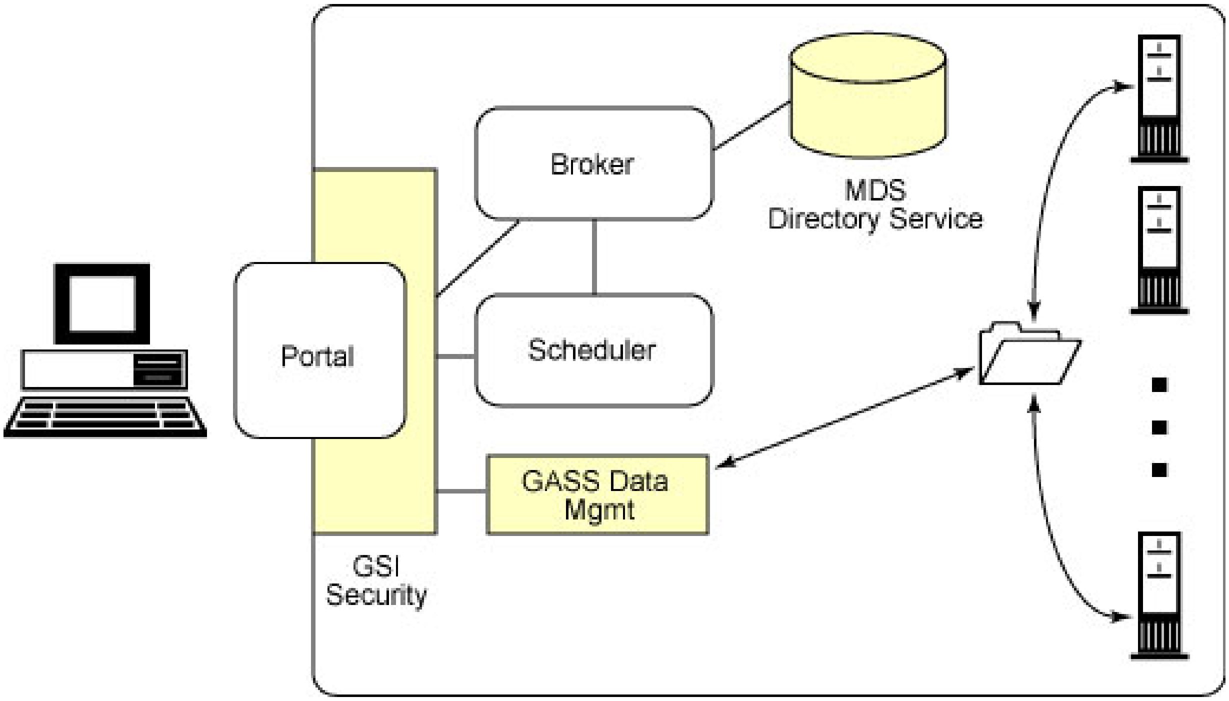
Figure 4. Scheduler



Data management

If any data -- including application modules -- must be moved or made accessible to the nodes where an application's jobs will execute, then there needs to be a secure and reliable method for moving files and data to various nodes within the grid. The Globus Toolkit contains a data management component that provides such services. This component, know as Grid Access to Secondary Storage (GASS), includes facilities such as GridFTP. GridFTP is built on top of the standard FTP protocol, but adds additional functions and utilizes the GSI for user authentication and authorization. Therefore, once a user has an authenticated proxy certificate, he can use the GridFTP facility to move files without having to go through a login process to every node involved. This facility provides third-party file transfer so that one node can initiate a file transfer between two other nodes.

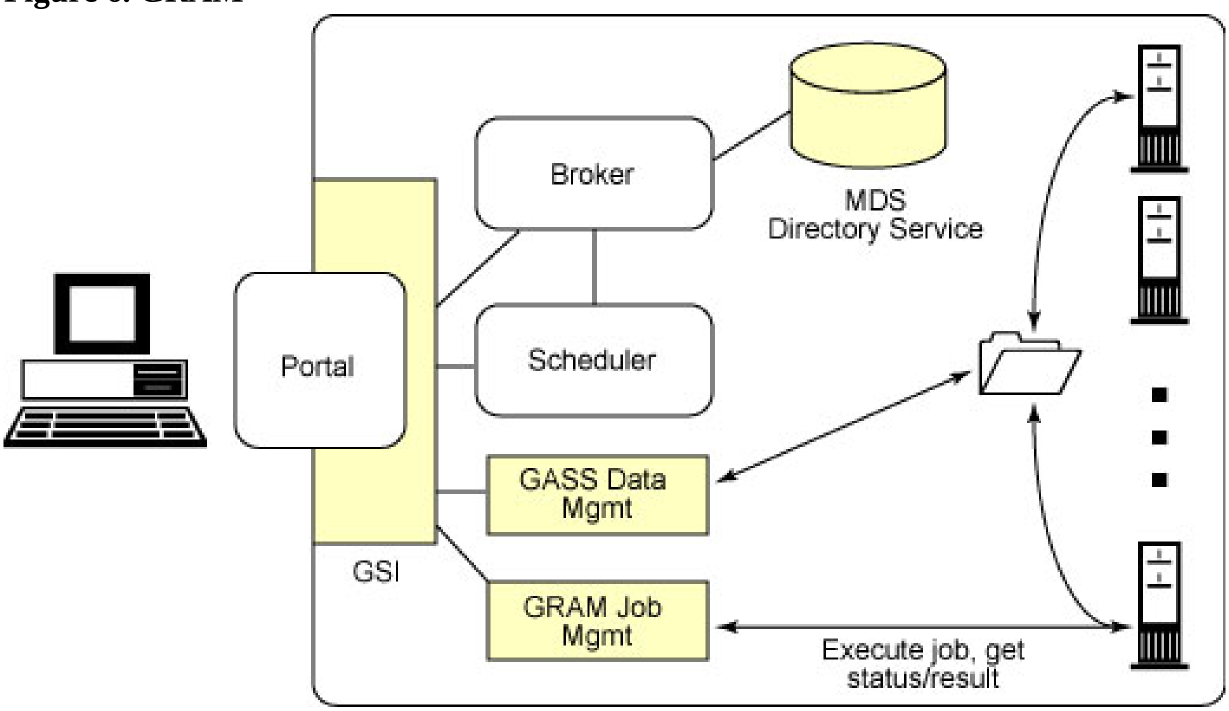
Figure 5. Data management



Job and resource management

With all the other facilities we have just discussed in place, we now get to the core set of services that help perform actual work in a grid environment. The Grid Resource Allocation Manager (GRAM) provides the services to actually launch a job on a particular resource, check its status, and retrieve its results when it is complete.

Figure 6. GRAM



Other facilities

There are other facilities that may need to be included in your grid environment and considered when designing and implementing your application. For instance, inter-process communication and accounting/chargeback services are two common facilities that are often required.

Job flow in a grid environment

In the previous section, we provided a brief and high-level view of the primary components of a grid environment. As you start thinking about enabling an application for a grid environment, it is important to keep in mind these components and how they relate and interact with one another. Depending on your grid implementation and application requirements, there are many ways in which these pieces can be put together to create a solution.

Summary

Grid computing is becoming a viable option in enterprises with the emergence and maturation of key technologies and open standards such as OGSA and OGSI. In this article, we have provided a high-level overview of the key facilities that make up grid environments.

Resources

- For more information and a detailed discussion about enabling applications for grid environments, see the IBM Redbook *Enabling Applications for Grid Computing with Globus* at http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/SG246936.html.
- The IBM Grid computing site can be found at http://www.ibm.com/grid
- For an analysis on peer-to-peer versus Grid computing, see the article "On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing," by Ian Foster and Adriana Iamnitchi at http://people.cs.uchicago.edu/~anda/papers/foster_grid_vs_p2p.pdf..
- For detailed information on the Globus Project, go to http://www.globus.org.

About the author

Bart Jacob is a Senior Consulting IT Specialist at the IBM International Technical Support Organization Center in Austin, Texas. He has 23 years of experience providing technical support across a variety of IBM products and technologies, including communications, object-oriented software development, and systems management and emerging technologies. He has over 10 years of experience at the ITSO, where he has been writing IBM redbooks and creating and teaching workshops around the world on a variety of topics. He holds a Masters degree in Numerical Analysis from Syracuse University.