GRID COMPUTING

By Sunil Shankar

Abstract

In the last few years there has been a rapid exponential increase in computer processing power, data storage and communication. But still there are many complex and computation intensive problems, which cannot be solved by supercomputers. These problems can only be met with a vast variety of heterogeneous resources. The increased use and popularity of the Internet and the availability of high-speed networks have gradually changed the way we do computing. These technologies have enabled the cooperative use of a wide variety of geographically distributed resources as a single more powerful computer. This new method of pooling resources for solving large-scale problems is called as grid computing. This paper describes the concepts underlying grid computing.

1. Introduction

The major goal of distributed computing research was to give users an easy, simple and transparent method of access to a vast set of heterogeneous resources. This is generally known as metacomputing. Metacomputing done on local area networks (LAN) are typically known as Cluster Computing Environments and those, which are done on wide area networks (WAN), are known as Grid Computing. This paper deals with the later one Grid Computing.

A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to computational capabilities. [1] Grid computing concepts were first studied and explored in the 1995 I-WAY experiment, in which high-speed networks were used to connect, for a short time, high-end resources at 17 sites throughout the USA. From this experiment a number of Grid research projects emerged that developed the core basic technologies for Grids in various communities and scientific disciplines. For example, the US National Science Foundation's National Technology Grid and NASA's Information Power Grid are both creating Grid infrastructures to serve university and NASA researchers, respectively. Across Europe and the United States, the closely related European Data Grid, Particle Physics Data Grid and Grid Physics Network (GriPhyN) projects plan to analyze data from frontier physics experiments. [2]

1.1. Characteristics of a Computational Grid [3][4]

There are many desirable properties and features that are required by a grid to provide users with a computing environment. They are as follows:

- Heterogeneity

  The grid involves a number of resources that are varied in nature and can encompass a large geographical distance through various domains.

- Scalability

  The grid should be tolerant to handle a large number of nodes without any performance degradation.

- Adaptability or Fault Tolerant

  In a grid unexpected computational aborts, hardware or software faults etc are high. These faults are generally handled by Resource Managers.

- Security

  All the user participating computers should be protected from any malicious manipulations or interventions.

1.2 Grid Components [7]

The major components that are necessary to form a grid as are shown in the Figure1. The components are as follows:

- User Level

  This layer houses the Application and High level Interfaces. Applications can be varied and encompass a vast variety of problems from chemistry to Nuclear Engineering. The high level interfaces implement an interface and protocols allowing the applications and users to access the middleware services.

- Middleware Level

  The major functionalities of grid systems normally occur in this layer. This layer provides many services like Resource discovery, resource scheduling and allocation, fault tolerance, security mechanisms and load balancing. It should provide the users a transparent view of the resources available.

- Resource Level

  This layer typically provides local services that render computational resources like CPU cycles, storage, computers, Network infrastructure, software etc.

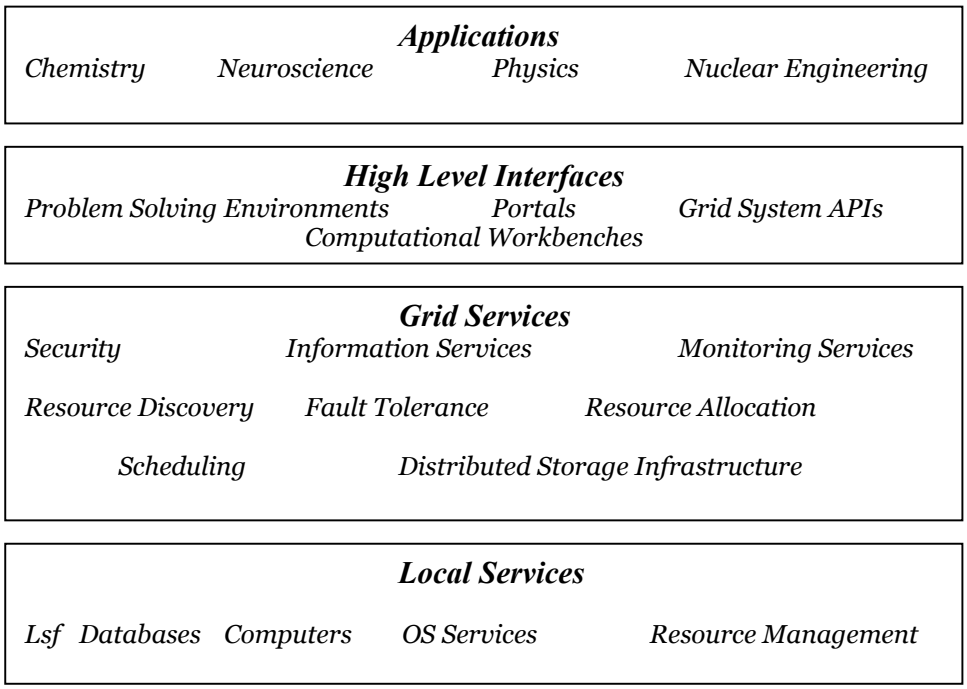


Figure 1: Grid Components [7]

2. Grid Architecture

Computational grids have to be designed so as to serve different communities with varying characteristics and requirements. Because of this reason we cannot have a uniform single architecture. But in general we can identify basic services that almost all the grids will provide although different grids will use different approaches for the realization of these services. [6]

This description of grid architecture does not provide a complete enumeration of all the required protocols and services but it identifies the requirements for general class of components. This architecture organizes the components into layers as shown in Figure 2.

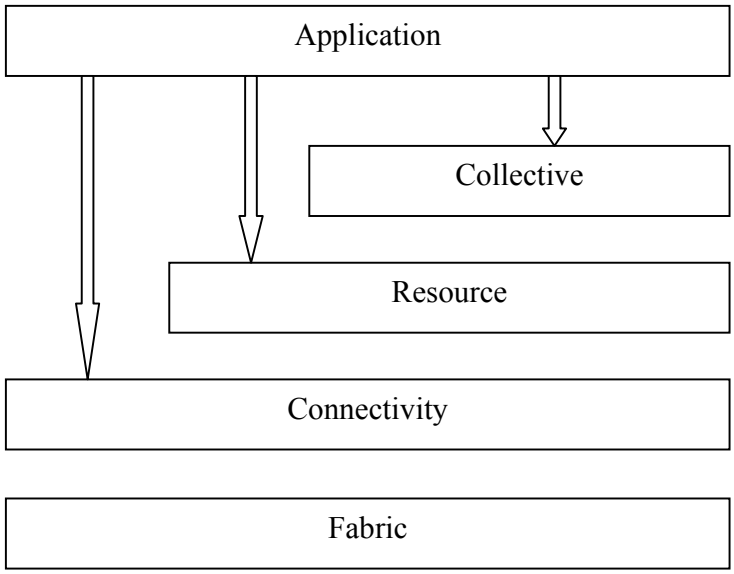


Figure 2: Grid Architecture

The Layers of the grid are as follows: [5]

- Fabric Layer

  This layer provides the resources, which could comprise computers (PCs running Windows NT or UNIX), storage devices and databases. The resource could also be a logical entity such as a distributed file system or computer pool. Excellent fabric functionality could mean that sophisticated sharing operations can be accomplished. For this, it should support enquiry mechanisms to discover their state, structure and capabilities. It should also have resource management mechanisms that provide some control of delivered quality of service.

- Connectivity Layer

  This layer consists of the core communication and authentication protocols required for transactions. Communication protocols enable the exchange of data between fabric layer resources. Authentication protocols provide secure cryptographic mechanisms for identifications of users and resources. For communication transport, naming and routing are required. These protocols can be drawn from TCP/IP protocol stack.

- Resource Layer

  This layer builds on the Connectivity layer communication and authentication protocols to define Application Program Interfaces (API) and Software Development Kit (SDK) for secure negotiation, initiation, monitoring, control, accounting and payment of sharing operations. The protocols, which the resource layers implement to achieve the above functionality are implemented with the

help of functions provided by the Fabric layer. Resource layer protocols can be distinguished primarily into two classes, which are ***Information Protocols*** and ***Management Protocols***

1. Information Protocol

   This protocol is used to obtain the necessary information about the structure and the state of the resource

2. Management Protocol

   In order to negotiate the access to the shared resources this protocol is used.

- Collective Layer

  This layer is different from the resource layer in the sense, while resource layer concentrates on interactions with single resource; this layer helps in coordinating multiple resources. Its tasks can be varied like Directory Services, Co-allocation and scheduling, monitoring, diagnostic services, and software discovery services.

- Application Layer

  This layer consists of the user applications and programs and which call upon another layer.

3. Grid Computing Projects

Grid computing is an active research area and is being worked on worldwide. There are many projects that are being carried out and they can be classified accordingly as:

- Mix and Match approach

  Ex: Globus

- Problem Solving Environments approach

  Ex: Netsolve

- Internet / WWW approach

  Ex: SETI, Distributed.net, Entropia, Charlotte, Javelin, Webflow,

- Object Oriented approach

  Ex: Legion

Mix and Match Approach

The Globus project is a multi-institutional research project, which is involved with the construction of computational grids. The key element of the Globus system is the Globus Metacomputing Toolkit, which defines the basic core services and capabilities required for the construction of a computational grid. Groups around the world are using the Globus Toolkit to build Grids and to develop Grid applications. [3][13]

The toolkit consists of a set of components that implement basic services for security, resource allocation, resource management, communication etc. This falls under a mix and match approach because Globus toolkit provides a "bag of services" from which grid designers and developers can select to meet their requirements. [14]

Brief components of globus toolkit:

Resource Management:

- *Globus Resource Allocation Manager (GRAM)*

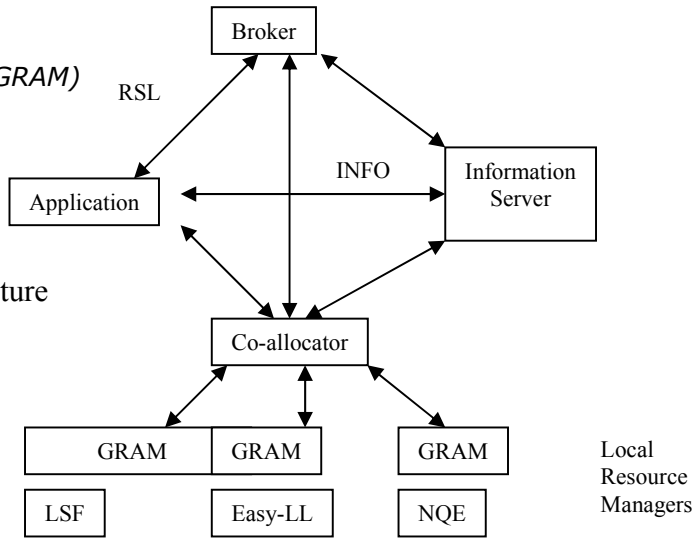


Fig 3: globus resource management architecture

The *Globus Resource Allocation Manager (GRAM)* provides resource allocation and process creation, monitoring, and management services. Each GRAM is responsible for a set of resources operating under the same site-specific allocation policy, which is often implemented by a local resource manager such as load sharing facility (LSF). A single manager can provide access to the nodes of a parallel computer, a cluster of workstations or a set of machines working within a pool. A computational grid, which is built with globus, will contain many GRAMs, which is each responsible for a particular "local" set of resources. The resource requirements are expressed by an application in terms of a high level RSL expression. A variety of resource brokers then implement domain specific resource discovery and selection polices by transforming abstract RSL expressions into progressively more specific requirements until a specific set of resources is identified.

The final step in the resource allocation process is then to decompose the RSL into a set of separate resource allocation requests and to dispatch each request to the appropriate GRAM. In high performance computations, it is necessary to collocate resources at this point, ensuring that a given set of resources are available for use simultaneously. Within Globus, a resource coallocator is responsible for providing this service: breaking the RSL into pieces, distributing it to the GRAMS, and coordinating the return values. Different coallocators can be constructed to implement different approaches to the problems of allocating and managing ensembles of resources. If any of the requested resources are unavailable for some reason, the entire coallocation request fails. [14]

Communication:

- *Nexus* provide communication services for heterogeneous environments, supporting multimethod communication, multithreading, and single-sided operations.

Information:

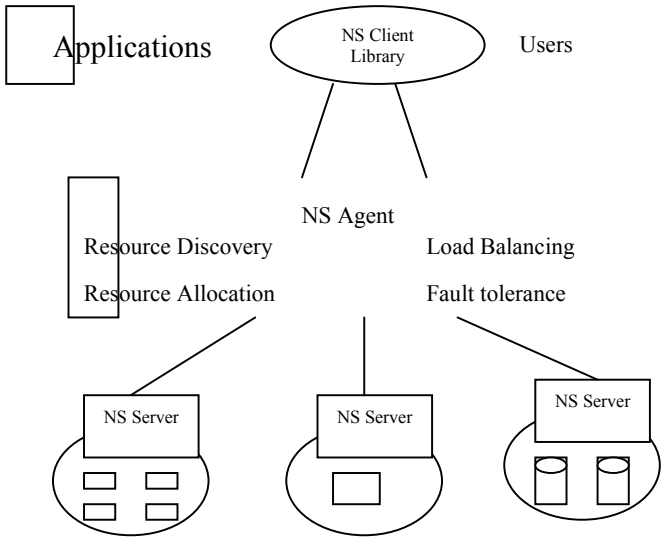
- The dynamic nature of grid environments means that toolkit components, programming tools, and applications must be able to adapt their behaviors in response to changes in system structure and state. The *globus metacomputing directory service* (MDS) is used for this type of adaptation by providing an information rich environment in which information about system components is always available. MDS stores and makes accessible information such as the architecture type, operating system version and amount of memory on a computer, network bandwidth and latency, available communication protocols, the mapping between IP addresses and network technology.

Security:

- Security in computational grids includes authentication , authorization, privacy and also many other concerns. The *globus security interface* (GSI) which is developed for globus toolkit provides security mechanisms currently.

Problem Solving Environments Approach

The Netsolve project uses this type of approach and is designed to be a simple to use middleware system. Netsolve allows the users to access additional software or hardware resources available remotely. In this manner this promotes the sharing of resources between research communities in the computational sciences.[15]



Architectural overview of the Netsolve System

Netsolve acts as a glue/middleware layer and brings the application/user together with the hardware/software it needs to complete useful tasks. The Netsolve client library is linked into the users application. The application then makes calls to NetSolve API for specific services. Through the API Netsolve client-users gain access to an aggregate of resources. Netsolve provides the user with a pool of computational resources. These resources are computational servers that can be running on single workstations, network of workstations that can collaborate for solving a problem. The user sends requests to the netsolve system asking for his numerical computation to be carried out. The main role of the NetSolve agent is to process this request and to choose the most suitable server for this particular computation. So once the server has been chosen, it is assigned the computation, uses its available numerical software and eventually returns the results to the user. [17]

NetSolve Examples:

Sub-surface modeling:

The implicit parallel accurate reservoir simulator, IPARS, developed at the University of Texas' Institute for Computational and Applied Mathematics, is a framework for developing parallel models of subsurface flow and fluid transport through porous media. It simulates single phase (water only), two phase (water and oil) or three phase (water, oil and gas) flow through a multi-block 3D porous medium. IPARS can be applied to model water table decline due to over-production near urban areas, or enhanced oil and gas recovery in industrial applications.

A NetSolve interface to the IPARS system allows users to access IPARS. The interface is primarily used from handy machines like laptop computers to run real-time simulations on clusters of workstations that allow for much quicker execution. IPARS runs primarily on LINUX. NetSolve makes it readily accessible from any platform. In addition, it has a problem-solving environment interfaced by a web browser, which one can use to enter input parameters and submit a request for execution of the IPARS simulator to a NetSolve system. The output images are then brought back and displayed by the web browser. This interaction shows how the NetSolve system can be used to create a robust grid computing environment in which powerful modeling software, like IPARS, becomes both easier to use and administrate.[17]

Internet/WWW Approach

This type of computing utilizes the idle time of Internet connected computers, which are geographically distributed around the world to run a huge distributed application. All the computing power required for the distributed application is provided from volunteer computers, which offer some of their idle time for execution. With over 100 million computers interconnected around the world this global computing approach utilizes theses computing resources to build a Very Large Scale Parallel Computer. [9]

One example is SETI@Home is a scientific effort seeking to determine if there is intelligent life outside Earth. SETI stands for the Search for Extra Terrestrial Intelligence and the project is dedicated to searching for patterns that may be signs of intelligent life amongst the mostly random mass of radio signals that reach the Earth from space. Each member of the project offers some of his or her computer's time to the cause. Membership is open to everyone with access to a computer and the Internet. [10]

Users typically install a screensaver program which will not only provide the usual graphics when their computer is idle, but will also perform sophisticated analysis of SETI data using the host computer. The data are tapped off Project Serendip IV's receiver and SETI survey operating on the 305-meter diameter Arecibo radio. [11] Here is how the computing works: [12]

1. The signal data are collected from the Arecibo dish in Puerto Rico.
2. The data are stored on tape along with observations, such as date, time, sky coordinates and notes about the receiving equipment.
3. The data are divided into small chunks that desktop user PCs can utilize.
4. The SETI@home program can then download a chunk data from the computer servers at UC-Berkeley.
5. The Home PC then analyzes the chunk of downloaded data according to the algorithms in the SETI@home program.
6. When finished, the PC uploads its results to the UC-Berkeley servers and flags any possible hits in the analysis.

7. After the upload, the PC requests another chunk of data from the server, and the process continues.

4. Discussions

Relationship and comparison with other technologies:

World Wide Web

The Web technologies such as HTTP, TCP/IP, and XML etc do an excellent job of supporting the browser-client-to-web-server interactions that are foundation of today's web. But these technologies lack the features required for richer interaction models. They do not provide integrated approaches to the coordinated use of resources at multiple sites for computation. So steps can be taken to integrate the Web and Grid Technologies.

Enterprise Computing Systems

Enterprise development technologies such as CORBA, Enterprise Java Beans, Java 2 Enterprise Edition, and DCOM are all systems designed to enable the construction of distributed applications. They provide standard resource interfaces, remote invocation mechanisms, and trading services for discovery and hence make it easy to share resources within a single organization only. Sharing arrangements are typically relatively static and restricted to occur within a single organization. The primary form of interaction is client-server, rather than the coordinated use of multiple resources.

Internet and peer-to-peer Computing
There are many Peer-to-peer computing systems like Napster, Gnutella, and Freenet file sharing systems and Internet computing systems like SETI@home, Parabon, and Entropia. These 2 types of systems are the example of the more general sharing modalities and computational structures. They have a lot of common characteristics with Grid computing technologies.

5. Grid Computing Applications

Grid Resources can be used to solve complex problems in many areas like high-energy physics, biophysics, nuclear simulations, weather monitoring and prediction, financial analysis, chemical engineering etc.

Projects, such as SETI@Home and Distributed.Net, build grids by linking multiple low-end computational resources, like PCs, from the Internet to detect extraterrestrial intelligence and crack security algorithms respectively.

Today large scale parameter study applications are using computational grid resources to crack algorithms and search for extraterrestrial intelligence.

6.Conclusion

There are many grid computational projects like globus, netsolve, entropia, SETI,condor,legion which are constantly improving the grid architecture and application interface.Grid computing has serious consequences and its implications are enormous in the field of computing.

7. References:

[1] Wolfgang Gentzsch: DOT-COMing the GRID: Using Grids for Business. GRID 2000

[2] A news article web page in "Nature"
http://www.nature.com/nature/webmatters/grid/grid.html

[3] M.A. Baker, R. Buyya, and D. Laforenza**, The Grid: International Efforts in Global Computing**, SSGRR 2000 The Computer & eBusiness Conference, l`Aquila, Italy July 31. 2000 - August 6. 2000

[4] Cécile Germain, Vincent Néri, Gilles Fedak and Franck Cappello ***Xtrem Web : building an experimental platform for global computing*** Grid2000 December 2000, IEEE Press.

[5] Foster, I., Kesselman, C. and Tuecke, S. ***The Anatomy of the Grid: Enabling Scalable Virtual Organizations.*** International Journal of High Performance Computing Applications, 15 (3). 200-222. 2001

[6] I. Foster and C. Kesselman. ***Computational grid****s*. In I. Foster and C. Kesselman, editors, The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 1998

[7] Dorian C. Arnold, Sathish S. Vadhiyar, and Jack Dongarra, **On the Convergence of Computational and Data Grids** Parallel Processing Letters, Volume 11, Numbers 2 and 3, June and September 2001, pp 187-202. ISSN 0129-6264.

[8] This is a news article on SETI.
http://biz.howstuffworks.com/seti.htm?printable=1

[9] Cecile Germain, Gilles Fedak, Vincent Neri, Franck Cappell, **Global Computing Systems** SciCom01, in cooperation with SIAM, to be published in LNCS, 2002.

[10] News article on SETI
http://www.ariadne.ac.uk/issue27/seti/#fn1

[11] W. T. Sullivan, III, D. Werthimer, S. Bowyer, J. Cobb, D. Gedye, D. Anderson. Published in: "Astronomical and Biochemical Origins and the Search for Life in the Universe", Proc. of the Fifth Intl. Conf. on Bioastronomy. 1997.

[12] News Article on SETI
http://biz.howstuffworks.com/seti.htm

[13] The Globus Web site and FAQs
http://www.globus.org/about/faq/general.html#globus

[14] I. Foster and C. Kesselman, "**The Globus project: A status report**," 7th IEEE Heterogeneous Computing Workshop (HCW '98), Mar. 1998, pp. 4--18.

[15] M. Miller, C. Moulding, J. Dongarra and C. Johnson, "**Grid-Enabling Problem Solving Environments: A Case Study of SCIRun and NetSolve**," *Proceedings of the High Performance Computing Symposium (HPC 2001) in 2001 Advanced Simulation Technologies Conference*, pp. 98-103, Society for Modeling and Simulation International, Apr. 2001

[16] Dorian C. Arnold and Jack Dongarra, **The NetSolve Environment: Progressing Towards the Seamless Grid**, 2000 International Conference on Parallel Processing (ICPP-2000), Toronto Canada, August 21-24, 2000

[17]Network enabled server: Examples and Applications', IEEE Computational Science & Engineering 5(3), 57-67.