Grid and Distributed Networks to handle mammoth tasks

Mrs. Radhika U Kamath
Lecturer,
V.P.M.'s Polytechnic, Thane

ABSTRACT

To analyze huge data sets instantaneously, run scenarios thousands of times, to get the results after operations always faster, accurate, with an increase in productivity the super computers can be used. But what if the same mammoth tasks can be achieved through the IT components the user already own? Mainframes, servers, databases, storage systems, desktop computers and workstations can be pulled together to realize this power of a super computer, through Grid and Distributed Computing.

Grid provides a collection of servers, clients and other resources often working collectively together to solve a problem. Much of Grid and distributed software technology addresses the issues of resource scheduling, quality of service, fault tolerance, decentralized control and security and so on. Grids are intrinsically distributed and heterogeneous, but must be perceived by the user as a single virtual platform or environment with uniform access to the resources.

The large and complex system of Grid software has to be robust, useful and provide an interoperable collection of services that support large-scale distributed computing and data management. Perspective of this paper provides a comprehensive look at the state of the art and best practices for wide areas of Grid and Distributed Computing.

INTRODUCTION

"A grid is a software framework providing layers of services to access and manage distributed hardware and software resources" [1] In 2001, Foster, Kesselman and Tuecke refined their definition of a Grid to "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations" [2]. This latest definition is the one most commonly used today to abstractly define a Grid.

Half a decade ago, Ian Foster later produced a checklist with three parts [3] that helps to understand exactly what can be identified as a Grid system. The first part to check off is that there is coordinated resource sharing with no centralized point of control that the users reside within different administrative domains. If this is not true, probably this is not a Grid system.

The second part to check off is the use of standard, open, general-purpose protocols and interfaces. If this is not the case it is likely that system components will not be able to communicate or interoperate, and it is likely that it is an application-specific system, and not the Grid.

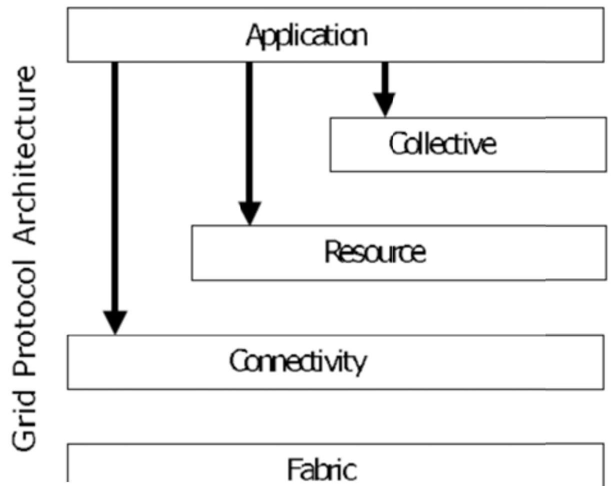
The final part to check off is that of delivering non-trivial qualities of service. The components that make up a Grid can be used in a coordinated way to deliver combined services, which are appreciably greater than the sum of the individual components. These services may be associated with throughput, response time, meantime between failure, security and many other facets.

From a commercial view point, IBM define a grid as "standards based application/resource sharing architecture that makes it possible for heterogeneous systems and applications to share, compute and storage resources transparently"[5].

ARCHITECTURE

Grids are focused on integrating existing resources with their hardware, operating systems, local resource management, and security infrastructure. In order to support the creation of the so called “Virtual Organizations”—a logical entity within which distributed resources can be discovered and shared as if they were from the same organization, Grids define and provide a set of standard protocols, middleware, toolkits, and services built on top of these protocols. Interoperability and security are the primary concerns for the Grid infrastructure as resources may come from different administrative domains, which have both global and local resource usage policies, different hardware and software configurations and platforms, and vary in availability and capacity.

Grids provide protocols and services at five different layers as identified in the Grid protocol architecture (see Figure 1). At the fabric layer, Grids provide access to different resource types such as compute, storage and network resource, code repository, etc. Grids usually rely on existing fabric components, for instance, local resource managers, General-purpose components, and specialized resource management services.



ecture

The connectivity layer defines core communication and authentication protocols for easy and secure network transactions. The resource layer defines protocols for the publication, discovery, negotiation, monitoring, accounting and payment of sharing operations on individual resources. The collective layer captures interactions across collections of resources, directory services. The application layer comprises whatever user applications built on top of the above protocols and APIs and operate in VO environments.

The GRID Models

Business model

The business model for Grids (at least that found in academia or government labs) is project-oriented in which the users or community represented by that proposal have certain number of service units (i.e. CPU hours) they can spend. For example, the TeraGrid operates in this fashion, and requires increasingly complex proposals be written for increasing number of computational power. The TeraGrid has more than a dozen Grid sites, all hosted at various institutions around the country.

What makes an institution want to join the TeraGrid? When an institution joins the TeraGrid with a set of resources, it knows that others in the community can now use these resources across the country. It also acknowledges the fact that it gains access to a dozen other Grid sites. This same model has worked rather well for many Grids around the globe, giving institutions incentives to join various Grids for access to additional resources for all the users from the corresponding institution.

There are also endeavors to build a Grid economy for a global Grid infrastructure that supports the trading, negotiation, provisioning, and allocation of resources based on the levels of services provided, risk and cost, and users' preferences; so far, resource exchange (e.g. trade storage for compute cycles), auctions, game theory based resource coordination, virtual currencies, resource brokers and intermediaries, and various other economic models have been proposed and applied in practice [6].

Compute Model:

Most Grids use a batch-scheduled compute model, in which a local resource manager (LRM) manages the compute resources for a Grid site, and users submit batch jobs to request some resources for some time. Many Grids have policies in place that enforce these batch jobs to identify the user and credentials under which the job will run for accounting and security purposes, the number of processors needed, and the duration of the allocation. For example, a job could say, stage in the input data from a URL to the local storage, run the application for 60 minutes on 100 processors, and stage out the results to some FTP server.

The job would wait in the LRM's wait queue until the 100 processors were available for 60 minutes, at which point the 100 processors would be allocated and dedicated to the application for the duration of the job. Due to the expensive scheduling decisions, data staging in and out, and potentially long queue times, many Grids don't natively support interactive applications; although there are efforts in the Grid community to enable lower latencies to resources via multi-level scheduling, to allow applications with many short-running tasks to execute efficiently on Grids [7].

Programming Model

Although programming model in Grid environments does not differ fundamentally from traditional parallel and distributed environments, it is obviously complicated by issues such as multiple administrative domains; large variations in resource heterogeneity, stability and performance; exception handling in highly dynamic (in that resources can join and leave pretty much at any time) environments, etc. Grids primarily target large-scale scientific computations, so it must scale to leverage large number/amount of resources, and we would also naturally want to make programs run fast and efficient in Grid environments, and programs also need to finish correctly, so reliability and fault tolerance must be considered.

In Grids, many applications are loosely coupled in that the output of one may be passed as input to one or more others. While such "loosely coupled" computations can involve large amounts of computation and communication, the concerns of the programmer tend to be different from those in traditional high performance computing, being focused on management issues relating to the large numbers of datasets and tasks rather than the optimization of inter-processor communication. In such cases, workflow systems [54] suit better in the specification and execution of such applications. More specifically, a workflow system allows the composition of individual (single step) components into a complex dependency graph, and it governs the flow of data and/or control through these components.

The Globus Toolkit version 4 contains Java and C implementations of WSRF, most of the Globus core services have been re-engineered to build around WSRF, these altogether will enable service oriented Grid programming model.

Application Model

Grids generally support many different kinds of applications, ranging from high performance computing (HPC) to high throughput computing (HTC). HPC applications are efficient at executing tightly coupled parallel jobs within a particular machine with low-latency interconnects and are generally not executed across a wide area network Grid; these applications typically use message passing interface (MPI) to achieve the needed interprocess communication. On the other hand, Grids have also seen great success in the execution of more loosely coupled applications that tend to be managed and executed through workflow systems or other sophisticated and complex applications. Related to HTC applications loosely coupled nature, there are other application classes, such Multiple Program Multiple Data (MPMD), capacity computing, utility computing, and embarrassingly parallel, each with their own niches [7]. These loosely coupled applications can be composed of many tasks (both independent and dependent tasks) that can be individually scheduled on many different computing resources across multiple administrative boundaries to achieve some larger application goal. Tasks may be small or large, uniprocessor or multiprocessor, compute-intensive or data-intensive. The set of tasks may be static or dynamic, homogeneous or heterogeneous, loosely or tightly coupled.

Another emerging class of applications in Grids is scientific gateways [7], which are front-ends to a variety of applications that can be anything from loosely-coupled to tightly-coupled. A Science Gateway is a community-developed set of tools, applications, and data collections that are integrated via a portal or a suite of applications. Gateways provide access to a variety of capabilities including workflows, visualization, resource discovery and job execution services through a browser-based user interface (which can arguably hide much of the complexities).

Security Model

Grids are built on the assumption that resources are heterogeneous and dynamic, and each Grid site may have its own administration domain and operation autonomy. Thus, security has been engineered in the fundamental Grid infrastructure. The key issues considered are: single sign-on, so that users can log on only once and have access to multiple Grid sites, this will also facilitate accounting and auditing; delegation, so that a program can be authorized to access resources on a user’s behalf and it can further delegate to other programs; privacy, integrity and segregation, resources belonging to one user cannot be accessed by unauthorized users, and cannot be tampered during transfer; coordinated resource allocation, reservation, and sharing, taking into consideration of both global and local resource usage policies. The public-key based GSI (Grid Security Infrastructure) protocols are used for authentication, communication protection, and authorization. Furthermore, CAS (Community Authorization Service) is designed for advanced resource authorization within and across communities.

Grids are stricter about its security. For example, although web forms are used to manage user accounts, sensitive information about new accounts and passwords requires also a person to person conversation to verify the person, perhaps verification from a sponsoring person who already has an account, and passwords will only be faxed or mailed, but under no circumstance will they be emailed. The Grid approach to security might be more time consuming, but it adds an extra level of security to help prevent unauthorized access.

Digital cancer imaging example:

Even though the grid technology could be implemented in all organizations with network infrastructure in place, organizations with the following profile are the best candidates for grid technology:

Applications take too long to finish.

Have many servers and looking to consolidate.

CPU/ Storage intensive applications.

Organization is geographically distributed.

In the cancer imaging example [5] given below, a grid is being used in medical research and diagnoses. Here is an illustration of how data resource sharing can be implemented in a grid to aid in research and potentially save lives.

Needs

A need has developed for a medical records system to capture, manage, and store patient files from any location for fast retrieval and diagnostic evaluations. These files include patient records, clinical history, and medical images, such as tomography (CT or cat scan), magnetic resonance imagery (MRI), ultra sounds, and mammograms. The main benefit for such a system is number of lives saved by early diagnosis and treatment.

In some hospitals, files have become mis-filed or otherwise make unavailable, which has left physicians and radiologists without comparative records for making diagnoses. Patients also sometimes obtained their health care services at other medical facilities, making it difficult to quickly retrieve those records to study the medical history and prior treatments. For the patient, there is the potential for complications leading to disabilities or death. For the hospital or medical practitioner, there is the possibility of litigation for misdiagnosis due to the unavailability of patients' records. As a result of missing records, additional, often costly, diagnostic studies are required.

Rising health care costs are due to the high overhead and administrative costs of maintaining paper and x-ray film file systems. Each year, the average hospital spends a lot to develop x-ray films, according to some estimates. This can be saved each year by using the medical records grid. A unique property exists for one form of cancer diagnosis in that a set of standards and protocols exist for mammography imagery. Traditionally, radiologists will scan the films with a magnifying glass to look for micro calcifications. There are then three options: Wait six months to see if the mass or calcifications increase, which suggests malignancy or Biopsy or Biopsy and removal.

It is very important that prior x-rays be taken into account when deciding on an option. To help, x-ray images can be converted to a digitized format and analytical tools used to help diagnose individual patients and quickly isolate abnormalities.

Solution

As shown in Figure 2, the use of a data grid disperses among four hospitals. The grid takes into account the following challenges:

-Data images of 160 MB per mammogram exam results in more than 5.6 PB per year.

-Daily traffic for archiving current exams and comparing past and present exams is a minimum of 28 TB.

-Network bandwidth responses range from high speed access to expert consultation and unscheduled exams down to low speed Internet access.

At each hospital, physicians, radiologists, or other authorized personnel upload images to a data repository, accessible

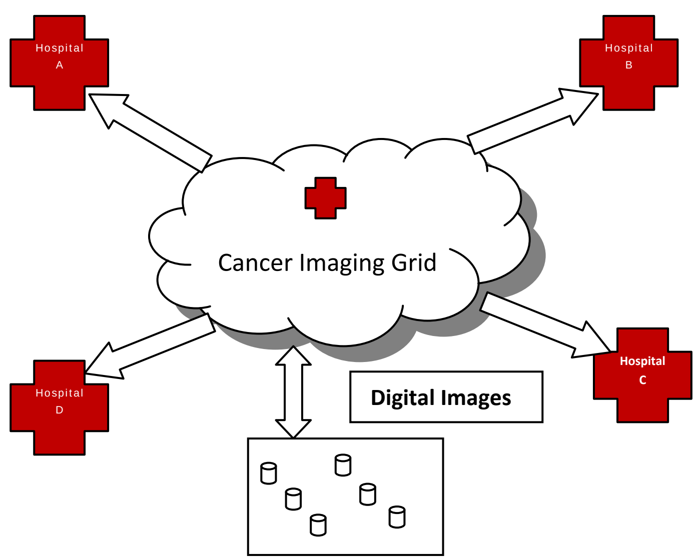


Figure 2 Conceptual view of the cancer imaging grid

by other hospitals. The medical personnel are able to query and retrieve patient records within 90 seconds.

Data grid

The data grid topology has the following characteristics

- Images are loaded at each hospital or end node. There are two servers. One acts as a temporary repository. The other is a link to the Internet or to the next generation of the Internet.
- Each hospital transmits images to a metropolitan hub.
- The metropolitan hubs funnel its images to a high-capacity regional hub for resource pooling.
- The distributed archive emulates one huge archive. Queries to the archive are handled rapidly by a secure, highly available database that indexes and catalogs the data.
- Access at local hospitals is transparent and fast.
- Management software monitors and controls the nodes and provides security and diagnostic information.

Application

Digital images of potential cancers are being shared among four hospitals. The potential exists to scale this grid to include thousands of hospitals and clinics and to share more types of patient data not only for aiding diagnoses but also for research and training. Using the grid, algorithms can uncover patterns to identify abnormal concentrations of cancer in the population. A suite of educational tools will be deployed on the grid to help doctors, medical students, and interns learn more about cancer and related diseases.

CONCLUSION

So, overall, the Grid is about resource sharing; this includes computers, storage, sensors and networks. Sharing is obviously always conditional and based on factors like trust, resource-based policies, negotiation and how payment should be considered. The Grid also includes coordinated problem solving, which is beyond simple client–server paradigm, which is in combinations of distributed data analysis, computation and collaboration. The Grid also involves dynamic, multi-institutional Virtual Organizations (VOs), where these new communities overlay classical organization structures, and these virtual organizations may be large or small, static or dynamic.

Grid Computing aims to “enable resource sharing and coordinated problem solving in dynamic, multi-institutional virtual organizations”. Grids provide a distributed computing paradigm or infrastructure that spans across multiple virtual organizations (VO) where each VO can consist of either physically distributed institutions or logically related projects/groups. The goal of such a paradigm is to enable federated resource sharing in dynamic, distributed environments. The approach taken by the de facto standard implementation – The Globus Toolkit, is to build a uniform computing environment from diverse resources by defining standard network protocols and providing middleware to mediate access to a wide range of heterogeneous resources. Globus addresses various issues such as security, resource discovery, resource provisioning and management, job scheduling, monitoring, and data management.

References:

[1] CCA, http://www.extreme.indiana.edu/ccat/glossary.html.

[2] Foster, I., Kesselman, C. and Tuecke, S., The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *International Journal of Supercomputer Applications*, 15(3), 2001.

[3] The Grid-Core Technologies, Maozhen Li, Brunel University, UK, Mark Baker, University of Portsmouth, UK
[4]R. Buyya, K. Bubendorfer. “Market Oriented Grid and Utility Computing”, Wiley Press, New York, USA, 2008.
[5] Introduction to Grid Computing with Globus -Luis Ferreira, Viktors Berstis, Jonathan Armstrong, Mike Kendzierski, Andreas Neukoetter, MasanobuTakagi, Richard Bing-Wo, Adeeb Amir, Ryo Murakawa, Olegario Hernandez, James Magowan, Norbert Bieberstein, ibm.com/redbooks.
[6] Amazon Simple Storage Service (Amazon S3), http://aws.amazon.com/s3, 2008.
[7] Cloud Computing and Grid Computing 360-Degree Compared-Ian Foster,Yong Zhao, Ioan Raicu,Shiyong Lu
[8]Y. Zhao, I. Raicu, I. Foster. “Scientific Workflow Systems for 21st Century, New Bottle or New Wine?” IEEE Workshop on Scientific Workflows, 2008.