

A Library Grid Network for Enhancing Learning in Nigerian Universities.

O.K. Boyinbode*, R.O. Akinyede, and O.S. Adeola

Department of Computer Science, Federal University of Technology, Akure, Nigeria.

*E-mail: okboyinbode@yahoo.com

ABSTRACT

Grid computing delivers on the potential in the growth and abundance of network connected systems and bandwidth: computation, collaboration and communication over the advanced web. The term 'grid' is variously used to describe a number of different, but related, ideas, including utility computing concepts, grid technologies, and grid standards.

In this paper the term 'Grid' is used in the widest sense to describe the ability to pool and share information technology (IT) resources in a global environment in a manner which achieves seamless, secure, transparent, simple access to a vast collection of many different types of hardware and software resources, (including compute nodes, software codes, data repositories, storage devices, graphics and terminal devices and instrumentation and equipment), through non-dedicated wide area networks, to deliver customized resources to specific applications. A library information grid technology is also proposed for the libraries of some prominent Nigeria Universities.

(Keywords: information technology, grid computing, library information grid technology, university education, Nigeria, virtualized resources)

INTRODUCTION

Over the last few years we have seen grid computing evolve from a technology associated with scientific and technical computing, into a business-innovating technology that is driving increased commercial adoption. Grid deployments accelerate application performance, improve productivity and collaboration, and optimize the resiliency of the IT infrastructure. By accelerating application performance, companies can more quickly deliver business results; achieving greater productivity, faster time to market, and increased customer satisfaction.

Grid technology also provides the ability to store, share and analyze large volumes of data, ensuring that people have access to information at the right time, which can improve decision making, employee productivity and collaboration. Grid technology improves resource utilization and reduces costs, while maintaining a flexible infrastructure that can cope with changing business demands, yet remain reliable, resilient and secure. At its core, grid is about virtualization, of both information and workload [3].

In a grid environment, resources are virtualized to create a pool of assets. Workload is spread across servers and data can be seamlessly retrieved. By separating applications and information from the infrastructure they run on, and providing this abstract, "virtualized" view, a new level of infrastructure flexibility can be achieved. Infrastructures can now dynamically adapt to business requirements, instead of the other way around. Resources are more fully utilized, resulting in decreased infrastructure costs, reduced processing time, increased responsiveness and faster time-to-market. This paper investigates the roots and evolution of grid computing as applied to information resource sharing.

REASONS FOR USING GRID COMPUTING

Exploiting Underutilized Resources

The easiest use of grid computing is to run an existing application on a different machine. The machine on which the application is normally run might be unusually busy due to an unusual peak in activity. The job in question could be run on an idle machine elsewhere on the grid [4]. There are at least two prerequisites for this scenario. First, the application must be executable remotely and without undue overhead. Second, the remote machine must meet any special hardware,

software, or resource requirement imposed by the application.

Parallel CPU Capacity

The potential for massive parallel CPU capacity is one of the most attractive features of a grid. In addition to pure scientific needs, such computing power is driving a new evolution in industries such as the bio-medical field, financial modeling, oil exploration, motion picture animation, and many others. The common attribute among such uses is that the applications have been written to use algorithms that can be partitioned into independently running parts.

A CPU intensive grid application can be thought of as many smaller “subjobs,” each executing on a different machine in the grid. To the extent that these sub jobs do not need to communicate with each other, the more “scalable” the application becomes. A perfectly scalable application will, for example, finish 10 times faster if it uses 10 times the number of processors.

Applications

There are many factors to consider in grid-enabling an application. One must understand that not all applications can be transformed to run in parallel on a grid and achieve scalability. Furthermore, there are no practical tools for transforming arbitrary applications to exploit the parallel capabilities of a grid. There are some practical tools that skilled application designers can use to write a parallel grid application.

However, automatic transformation of applications is a science in its infancy. This can be a difficult job and often requires top mathematics and programming talents, if it is even possible in a given situation. New computation intensive applications written today are being designed for parallel execution and these will be easily grid-enabled, if they do not already follow emerging grid protocols and standards.

Virtual Resources and Virtual Organizations for Collaboration

Another important grid computing contribution is to enable and simplify collaboration among a

wider audience. In the past, distributed computing promised this collaboration and achieved it to some extent. Grid computing takes these capabilities to an even wider audience, while offering important standards that enable very heterogeneous systems to work together to form the image of a large virtual computing

Access to Additional Resources

In addition to CPU and storage resources, a grid can provide access to increased quantities of other resources and to special equipment, software, licenses, and other services. The additional resources can be provided in additional numbers and/or capacity. For example, if a user needs to increase his total bandwidth to the Internet to implement a data mining search engine, the work can be split among grid machines that have independent connections to the Internet. In this way, the total searching capability is multiplied, since each machine has a separate connection to the Internet. If the machines had shared the connection to the Internet, there would not have been an effective increase in bandwidth.

Resource Balancing

A grid federates a large number of resources contributed by individual machines into a greater total virtual resource. For applications that are grid-enabled, the grid can offer a resource balancing effect by scheduling grid jobs on machines with low utilization.

This feature can prove invaluable for handling occasional peak loads of activity in parts of a larger organization. This can happen in two ways: _ an unexpected peak can be routed to relatively idle machines in the grid. _ If the grid is already fully utilized, the lowest priority work being performed on the grid can be temporarily suspended or even cancelled and performed again later to make room for the higher priority work.

Reliability

High-end conventional computing systems use expensive hardware to increase reliability. They are built using chips with redundant circuits that vote on results, and contain much logic to achieve

graceful recovery from an assortment of hardware failures. The machines also use duplicate processors so that when they fail, one can be replaced without turning the other off. Power supplies and cooling systems are duplicated. The systems are operated on special power sources that can start generators if utility power is interrupted. All of this builds a reliable system, but at a great cost, due to the duplication of high-reliability components.

Management

The goal to virtualize the resources on the grid and more uniformly handle heterogeneous systems will create new opportunities to better manage a larger, more dispersed IT infrastructure. It will be easier to visualize capacity and utilization, making it easier for IT departments to control expenditures for computing resources over a larger organization.

GRID CONCEPTS AND COMPONENTS

In this section, we introduce the various grid concepts, components, and terms in more detail.

Types of Resources

A grid is a collection of machines, sometimes referred to as “nodes,” “resources,” “members,” “donors,” “clients,” “hosts,” “engines,” and many other such terms. They all contribute any combination of resources to the grid as a whole. Some resources may be used by all users of the grid while others may have specific restrictions.

Computation

The most common resource is computing cycles provided by the processors of the machines on the grid. The processors can vary in speed, architecture, software platform, and other associated factors, such as memory, storage, and connectivity. There are three primary ways to exploit the computation resources of a grid. The first and simplest is to use it to run an existing application on an available machine on the grid rather than locally.

The second is to use an application designed to split its work in such a way that the separate parts

can execute in parallel on different processors. The third is to run an application that needs to be executed many times, on many different machines in the grid. “Scalability” is a measure of how efficiently the multiple processors on a grid are used. If twice as many processors makes an application complete in one half the time, then it is said to be perfectly scalable. However, there may be limits to scalability when applications can only be split into a limited number of separately running parts or if those parts experience some other contention for resources of some kind.

Storage

The second most common resource used in a grid is data storage. A grid providing an integrated view of data storage is sometimes called a “data grid.” Each machine on the grid usually provides some quantity of storage for grid use even if temporary. Storage can be memory attached to the processor or it can be “secondary storage” using hard disk drives or other permanent storage media. Memory attached to a processor usually has very fast access but is volatile. It would best be used to cache data or to serve as temporary storage for running applications.

Secondary storage in a grid can be used in interesting ways to increase capacity, performance, sharing, and reliability of data. Many grid systems use mountable networked file systems, such as Andrew File System (AFS®), Network File System (NFS), Distributed File System (DFS™), or General Parallel File System (GPFS). These offer varying degrees of performance, security features, and reliability features. Capacity can be increased by using the storage on multiple machines with a unifying file system. Any individual file or database can span several storage devices and machines, eliminating maximum size restrictions often imposed by file systems shipped with operating systems. A unifying file system can also provide a single uniform name space for grid storage. This makes it easier for users to reference data residing in the grid, without regard for its exact location. In a similar way, special database software can “federate” an assortment of individual databases and files to form a larger, more comprehensive database, accessible using database query functions. More advanced file systems on a grid can automatically duplicate sets of data, to provide redundancy for increased reliability and increased performance.

An intelligent grid scheduler can help select the appropriate storage devices to hold data, based on usage patterns. Then jobs can be scheduled closer to the data, preferably on the machines directly connected to the storage devices holding the required data. Data striping can also be implemented by grid file systems. When there are sequential or predictable access patterns to data, this technique can create the virtual effect of having storage devices that can transfer data at a faster rate than any individual disk drive.

Communications

The rapid growth in communication capacity among machines today makes grid computing practical, compared to the limited bandwidth available when distributed computing was first emerging. Therefore, it should not be a surprise that another important resource of a grid is data communication capacity. This includes communications within the grid and external to the grid. Communications within the grid are important for sending jobs and their required data to points within the grid. Some jobs require a large amount of data to be processed and it may not always reside on the machine running the job. The bandwidth available for such communications can often be a critical resource that can limit utilization of the grid. External communication access to the Internet, for example, can be valuable when building search engines. Machines on the grid may have connections to the external Internet in addition to the connectivity among the grid machines. When these connections do not share the same communication path, then they add to the total available bandwidth for accessing the internet. Redundant communication paths are sometimes needed to better handle potential network failures and excessive data traffic.

In some cases, higher speed networks must be provided to meet the demands of jobs transferring larger amounts of data. A grid management system can better show the topology of the grid and highlight the communication bottlenecks. This information can in turn be used to plan for hardware upgrades.

Software and Licenses

The grid may have software installed that may be too expensive to install on every grid machine. Using a grid, the jobs requiring this software are

sent to the particular machines on which this software happens to be installed. When the licensing fees are significant, this approach can save significant expenses for an organization. Some software licensing arrangements permit the software to be installed on all of the machines of a grid but may limit the number of installations that can be simultaneously used at any given instant. License management software keeps track of how many concurrent copies of the software are being used and prevents more than that number from executing at any given time. The grid job schedulers can be configured to take software licenses into account, optionally balancing them against other priorities or policies.

GRID CONSTRUCTION: GENERAL PRINCIPLES

This section briefly highlights some of the general principles that underlie the construction of the Grid. In particular, the idealized design features that are required by a Grid to provide users with a seamless computing environment are discussed. Four main aspects characterize a Grid.

- *Multiple administrative domains and autonomy.* Grid resources are geographically distributed across multiple administrative domains and owned by different organizations. The autonomy of resource owners needs to be honored along with their local resource management and usage policies.
- *Heterogeneity.* A Grid involves a multiplicity of resources that are heterogeneous in nature and will encompass a vast range of technologies.
- *Scalability.* A Grid might grow from a few integrated resources to millions. This raises the problem of potential performance degradation as the size of Grids increases. Consequently, applications that require a large number of geographically located resources must be designed to be latency and bandwidth tolerant.
- *Dynamicity or adaptability.* In a Grid, resource failure is the rule rather than the exception. In fact, with so many resources in a Grid, the probability of some resource failing is high. Resource managers or applications must tailor their behavior dynamically and use the available resources and services efficiently and effectively.

The steps necessary to realize a Grid include:

- integration of individual software and hardware components into a combined networked resource (e.g. a single system image cluster);
- deployment of: – low-level middleware to provide a secure and transparent access to resources;–User-level middleware and tools for application development and the aggregation of distributed resources;
- development and optimization of distributed applications to take advantage of the available resources and infrastructure.

The components that are necessary to form a Grid (shown in Figure 1) are as follows:

- *Grid fabric.* This consists of all the globally distributed resources that are accessible from anywhere on the Internet. These resources could be computers (such as PCs or Symmetric Multi-Processors) running a variety of operating systems (such as UNIX or Windows), storage devices, databases, and special scientific instruments such as a radio telescope or particular heat sensor.
- *Core Grid middleware.* This offers core services such as remote process management, co-allocation of resources, storage access, information registration and discovery, security, and aspects of Quality of Service (QoS) such as resource reservation and trading.
- *User-level Grid middleware.* This includes application development environments, programming tools and resource brokers for managing resources and scheduling application tasks for execution on global resources.
- *Grid applications and portals.* Grid applications are typically developed using Grid-enabled languages and utilities such as HPC++ or MPI.

An example application, such as parameter simulation or a grand-challenge problem, would require computational power, access to remote data sets, and may need to interact with scientific instruments. Grid portals offer Web-enabled application services, where users can submit and collect results for their jobs on remote resources through the Web. In attempting to facilitate the collaboration of multiple organizations running diverse autonomous heterogeneous resources, a number of basic principles should be followed so that the Grid environment:

- does not interfere with the existing site administration or autonomy;
- does not compromise existing security of users or remote sites;
- does not need to replace existing operating systems, network protocols, or services;
- allows remote sites to join or leave the environment whenever they choose;
- does not mandate the programming paradigms, languages, tools, or libraries that a user wants;
- provides a reliable and fault tolerant infrastructure with no single point of failure;
- provides support for heterogeneous components;
- uses standards, and existing technologies, and is able to interact with legacy applications;
- provides appropriate synchronization and component program linkage.

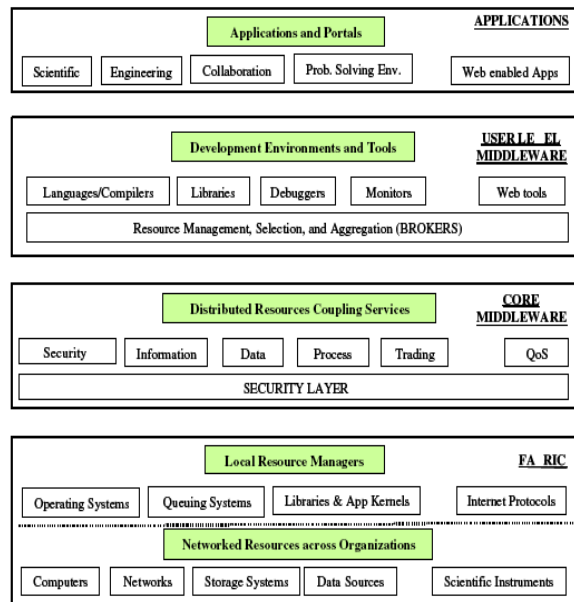


Figure 1: A Layered Grid Architecture and Components [1].

As one would expect, a Grid environment must be able to interoperate with a whole spectrum of current and emerging hardware and software technologies. An obvious analogy is the Web. Users of the Web do not care if the server they

are accessing is on a UNIX or Windows platform. From the client browser's point of view, they 'just' want their requests to Web services handled quickly and efficiently. In the same way, a user of a Grid does not want to be bothered with details of its underlying hardware and software infrastructure. A user is really only interested in submitting their application to the appropriate resources and getting correct results back in a timely fashion. An ideal Grid environment will therefore provide access to the available resources in a seamless manner such that physical discontinuities, such as the differences between platforms, network protocols, and administrative boundaries become completely transparent. In essence, the Grid middleware turns a radically heterogeneous environment into a virtual homogeneous one.

The following are the main design features required by a Grid environment:

- *Administrative hierarchy.* An administrative hierarchy is the way that each Grid environment divides itself up to cope with a potentially global extent. The administrative hierarchy determines how administrative information flows through the Grid.
- *Communication services.* The communication needs of applications using a Grid environment are diverse, ranging from reliable point-to-point to unreliable multicast communications. The communications infrastructure needs to support protocols that are used for bulk- data transport, streaming data, group communications, and those used by distributed objects. The network services used also provide the Grid with important QoS parameters such as latency, bandwidth, reliability, fault-tolerance, and jitter control.
- *Information services.* A Grid is a dynamic environment where the location and types of services available are constantly changing. A major goal is to make all resources accessible to any process in the system, without regard to the relative location of the resource user. It is necessary to provide mechanisms to enable a rich environment in which information is readily obtained by requesting services. The Grid information (registration and directory) services components provide the mechanisms for registering and obtaining information about the Grid structure, resources, services, and status.
- *Naming services.* In a Grid, like in any distributed system, names are used to refer to a wide variety of objects such as computers, services, or data objects. The naming service provides a uniform name space across the complete Grid environment. Typical naming services are provided by the international X.500 naming scheme or DNS, the Internet's scheme.
- *Distributed file systems and caching.* Distributed applications, more often than not, require access to files distributed among many servers. A distributed file system is therefore a key component in a distributed system. From an applications point of view it is important that a distributed file system can provide a uniform global namespace, support a range of file I/O protocols, require little or no program modification, and provide means that enable performance optimizations to be implemented, such as the usage of caches.
- *Security and authorization.* Any distributed system involves all four aspects of security: confidentiality, integrity, authentication, and accountability. Security within a Grid environment is a complex issue requiring diverse resources autonomously administered to interact in a manner that does not impact the usability of the resources or introduces security holes/lapses in individual systems or the environments as a whole. A security infrastructure is the key to the success or failure of a Grid environment.
- *System status and fault tolerance.* To provide a reliable and robust environment it is important that a means of monitoring resources and applications is provided. To accomplish this task, tools that monitor resources and application need to be deployed.
- *Resource management and scheduling.* The management of processor time, memory, network, storage, and other components in a Grid is clearly very important. The overall aim is to efficiently and effectively schedule the applications that need to utilize the available resources in the Grid computing environment. From a user's point of view, resource management and scheduling should be transparent; their interaction with it being confined to a manipulating mechanism for submitting their application. It is important in a Grid that a resource management and scheduling service can interact with those that may be installed locally.

- *Computational economy and resource trading*; As a Grid is constructed by coupling resources distributed across various organizations and administrative domains that may be owned by different organizations, it is essential to support mechanisms and policies that help in regulate resource supply and demand [8,9] An economic approach is one means of managing resources in a complex and decentralized manner. This approach provides incentives for resource owners, and users to be part of the Grid and develop and using strategies that help maximize their objectives.
- *Programming tools and paradigms*. Grid applications (multi-disciplinary applications) couple resources that cannot be replicated at a single site even or may be globally located for other practical reasons. A Grid should include interfaces, APIs, utilities, and tools to provide a rich development environment. Common scientific languages such as C, C++, and FORTRAN should be available, as should application-level interfaces such as MPI and PVM. A variety of programming paradigms should be supported, such as message passing or distributed shared memory. In addition, a suite of numerical and other commonly used libraries should be available.
- *User and administrative GUI*. The interfaces to the services and resources available should be intuitive and easy to use. In addition, they should work on a range of different platforms and operating systems. They also need to take advantage of Web technologies to offer a view of portal supercomputing. The Web-centric approach to access supercomputing resources should enable users to access any resource from anywhere over any platform at any time. That means, the users should be allowed to submit their jobs to computational resources through a Web interface from any of the accessible platforms such as PCs, laptops, or Personal Digital Assistant, thus supporting the ubiquitous access to the Grid. The provision of access to scientific applications through the Web (e.g. RWCPs parallel protein information analysis system [10] leads to the creation of science portals.

SOME EXISTING GRID COMPUTING PROJECTS

Globus: Globus [13] provides a software infrastructure that enables applications to handle distributed heterogeneous computing resources as a single virtual machine. The Globus project is a U.S multi-institutional research effort that seeks to enable the construction of computational Grids. A computational Grid, in this context, is a hardware and software infrastructure that provides dependable, consistent, and pervasive access to high-end computational capabilities, despite the geographical distribution of both resources and users. Globus provides basic services and capabilities that are required to construct a computational Grid.

Legion: Legion [14] is an object-based metasystem developed at the University of Virginia. Legion provides the software infrastructure so that a system of heterogeneous, geographically distributed, high-performance machines can interact seamlessly.

GridSim: GridSim [15] is a toolkit for modeling and simulation of Grid resources and application scheduling. It provides a comprehensive facility for the simulation of different classes of heterogeneous resources, users, applications, resource brokers, and schedulers. It has facilities for the modeling and simulation of resources and network connectivity with different capabilities, configurations, and domains. It supports primitives for application composition, information services for resource discovery, and interfaces for assigning application tasks to resources and managing their execution. These features can be used to simulate resource brokers or Grid schedulers to evaluate performance of scheduling algorithms or heuristics.

Gridbus: The Gridbus (GRID computing and BUSiness) toolkit project is engaged in the design and development of cluster and grid middleware technologies for service-oriented computing [15]. It provides end-to-end services to aggregate or lease services of distributed resources depending on their availability, capability, performance, cost, and users' QoS requirements. The key objective of the Gridbus project is to develop fundamental, next- generation cluster and grid technologies that support utility computing.

Information Power Grid

The NAS Systems Division is leading the effort to build and test NASA's IPG [2], a network of high performance computers, data storage devices, scientific instruments, and advanced user interfaces. The overall mission of the IPG is to provide NASA's scientific and engineering communities with a substantial increase in their ability to solve problems that depend on the use of large-scale and/or distributed resources. The project team is focused on creating an infrastructure and services to locate, combine, integrate, and manage resources from across the NASA centers. An important goal of the IPG is to produce a common view of these resources, and at the same time provide for distributed management and local control.

INFORMATION GRID: AN INFORMATION INFRASTRUCTURE FOR GRID

Every year, data volumes increase 800 MB per user. The sheer quantity of information can overwhelm the IT systems that must collect, store, retrieve, manage and protect it. Nearly one third of an IT staff's time is spent searching for relevant data. Although the data may be timely, is it easy to use? Is it integrated? Is it tailored to business needs, and is it cost-effective to manage and retrieve? With a dynamic infrastructure, enabled by grid computing technology, customers can deploy capabilities that allow them to capture and analyze customer information to increase the speed and accuracy of business decisions. "Information on Demand"

Information Infrastructure

Establishing an information infrastructure for grid is a core component of the grid computing model [11, 12]. It allows end users and applications secure access to any information source – regardless of where it exists – over a local and/or distributed network in intranet, internet or even extranet environments. It provides access to heterogeneous files, databases, storage systems and supports data sharing for processing and/or large-scale collaboration. The initial focus of a grid implementation may be on shortening the processing time of a single application. However, as more applications and system resources become associated with a grid environment, the need to consider how data is accessed and

managed must be taken into consideration. While a grid may be optimally constructed to intelligently schedule and manage workloads, a poor information-access and storage-system deployment scheme could significantly reduce the benefits that can be derived from a grid implementation.

In a grid environment, applications are not necessarily dedicated to running on specific processors (or nodes). Applications can be provisioned onto different processors within the grid at any given time, depending on their business priority. Moreover, nodes in the grid can be geographically dispersed. The challenge is making sure that data is easily accessible and doesn't create network bandwidth problems as a result of transporting it to computing locations in a distributed environment. Customers would like to be able to have an easier method of pulling together data and information from multiple "business" areas within and beyond the enterprise, without disturbing the original format of the data (or how the data is managed at its source). Therefore, it is necessary to ensure that any node in the grid can access the data ubiquitously, without having to build a new path to access it. Physically consolidating data can be incredibly expensive, time consuming and negatively impact performance of applications. Information-grid solutions may also address the following customer challenges:

- Fragmentation of data resources and assets due to a heterogeneous environment or underutilized compute and storage resources
- Cumbersome data access and poor integration
- Data security and protection
- Complex management of decentralized systems and resources

Challenges and Solutions

The information grid [17] solves the problem of managing information, which may include databases, files, storage spanning across heterogeneous resources and software and hardware. The following are some common computing challenges and their solutions.

- **Challenge No. 1:** Accessing "heterogeneous data" stored in different formats across multiple "business" areas. The application must perform multiple I/O requests to retrieve the data that

slows down the execution of the job. Programmers that build and maintain these applications must be aware of different formats and determine how to transform and join the data within their applications.

Solution: Data-access virtualization technology across diverse data formats is instrumental in helping solve the challenge of reading data stored in different formats. Programmers can simplify access to data that are stored in mixed formats (e.g., multi vendor relational databases, flat files) by enabling these data to be accessed with a single structured query language instruction. Such access also helps reduce the need to move remote files. The data's virtual view is also known as federated access to the data. That is, making the data appear as one source even though the data are distributed and stored in mixed formats. In the event that large volumes of data need to be tailored for an application, specific extraction, transformation and loading functions can be preformed on the grid. Once the data has been prepared into a proper format for an application, the data can be temporally "transported" to and cached at a location where the processing will take place. The ability to cleanse, transform, federate and analyze data across "heterogeneous data" sources is crucial to gaining information insight and making more effective business decisions.

- **Challenge No. 2:** Data discovery and information delivery in a grid environment with mixed file system types. It is difficult for application developers and end-users to locate and access data because data are stored under multiple directories that are associated with each file system type.

Solution: Poor storage-resource utilization can be resolved through the use of SAN technology. Optimal solutions would include SAN software that enables system administrators to create a virtual view of all of the SAN storage, making them appear to be one homogeneous set of files with a common name space. Also, it is necessary to move large data volumes across a network to facilitate remote processing. A software solution that addresses this challenge should enable data to be cached close to where distributed processing occurs. An ideal solution would include global naming, secure wide-area access to consistent, current data and distributed data access including POSIX/NFS interface, access control and remote-data caching. Similarly,

virtualization of heterogeneous file systems can help manage a complex SAN environment. Creating a single name space for the file system helps programmers and administrators locate and access data more easily versus having to identify files individually and determine what access path is required to reference the data. IBM's General Parallel File System or SAN File System can be used as foundations for this solution

- **Challenge No. 3:** Mixed vendor storage is common within any given enterprise. It is costly for administrators to manually manage data placement across heterogeneous sets of storage devices. In many cases, bottlenecks occur in retrieving data from these devices due to congestion of over utilized devices even though there may be space available on underutilized storage resources.

Solution: Frequently, customers have heterogeneous (multi-vendor) storage devices installed. Each vendor's storage device comes with its own management console, which makes it difficult to efficiently manage data placement across the various devices and ensure that there isn't uneven data loading. Uneven data distribution could cause some of the devices to be over utilized while others remain underutilized. This unbalanced condition could lead to bottlenecks when attempting to retrieve data, thus slowing the application processing. A virtualization portal that consolidates the view across all of the SAN devices allows a single administrator to see how data is being loaded on these devices. Administrators are then able to shift data from over utilized devices to those that are underutilized without disrupting how applications access the data.

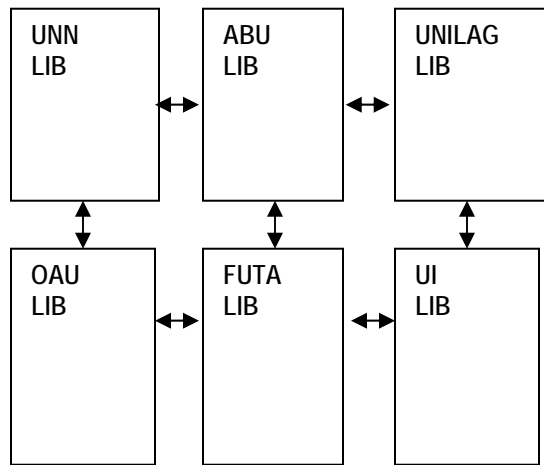
The Optimal Information Infrastructure

Once the aforementioned solutions have been applied, the information grid has been established. These solutions address many potential problems of accessing data, managing heterogeneous file and storage systems and removing the network effect of supplying data for remote processing. By applying each of these solutions to create a virtual environment, distributed computing in a grid environment can achieve its maximum benefits.

The enterprise has the flexibility to utilize all of its computing assets by enabling information to be virtually managed and presented

THE LIBRARY INFORMATION GRID TECHNOLOGY

As we have discussed earlier the core function of a Grid technology is to share resources. We are proposing a library information grid that will connect and ensure resource sharing among some Nigeria Universities Libraries. We are starting with six Universities, which can be increased with time. Figure 2 shows the proposed architecture for Library information Grid.



UNN LIB: University of Nigeria Nsukka Library
 ABU LIB: Ahmadu Bello University Zaria Library
 UNILAG LIB: University of Lagos Library
 OAU LIB: Obafemi Awolowo University Library
 FUTA LIB: Federal University of Technology Akure Library
 UI LIB: University of Ibadan Library

Figure 2: A Proposed Library Information Grid for Six Universities in Nigeria.

CONCLUSION

Grid technology has the ability to store, share and analyze large volumes of data, ensuring that people have access to information at the right time, which can improve decision making, employee productivity and collaboration. Grid technology improves resource utilization and reduces costs, while maintaining a flexible infrastructure that can cope with demands, yet remain reliable, resilient and secure. At its core,

grid is about virtualization, of both information and workload. Java, with its related technologies and growing repository of tools and utilities, is having a huge impact on the growth and development of Grid environments. From a relatively slow start, the developments in Grid computing are accelerating fast with the advent of these new and emerging technologies. The proposed Library information Grid Technology when implemented in the nearest future will help our universities in sharing their available information.

REFERENCES

1. Foster, I. and Kesselman, C. 1999. *The Grid: Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann: San Francisco, CA.
2. Johnston, W., Gannon, D., and Nitzberg, B. 1999. "Grids as Production Computing Environments: The Engineering Aspects of NASA's Information Power Grid". *Eighth IEEE International Symposium on High Performance Distributed Computing*. Redondo Beach, CA, August 1999. IEEE. Computer Society Press: Los Alamitos, CA
3. Buyya, R. 2009. The World-Wide Grid. <http://www.buyya.com/ecogrid/wwg/>.
4. NSF. 2009. "Tera-Grid". <http://www.teraGrid.org/>.
5. Hoschek, W., Jaen-Martinez, J., Samar, A., Stockinger, H., and Stockinger, K. 2000. "Data Management in an International Data Grid Project". *Proceedings of the 1st IEEE/ACM International Workshop on Grid Computing (Grid'2000)*. Bangalore, India. 17-20 December 2000. Springer: Berlin.
6. W3C. 2009. "Web Services Activity". <http://www.w3.org/2002/ws/>.
7. Buyya, R., Giddy, J., and Abramson, D. 2001. "A Case for Economy Grid Architecture for Service-Oriented Grid Computing". *10th IEEE International Heterogeneous Computing Workshop (HCW 2001), in Conjunction with IPDPS 2001*. San Francisco, CA, April 2001. IEEE Computer Society Press: Los Alamitos, CA.
8. Buyya, R., Abramson, D., and Giddy, J. 2000. "Economy Driven Resource Management Architecture for Computational Power Grids". *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2000)*. Las Vegas, NV, 2000. CSREA Press: Athens, GA.
9. Buyya, R. 2002. "Economic- Based Distributed Resource Management and Scheduling for Grid

Computing". Ph.D. Thesis. Monash University: Melbourne, Australia.

10. PAPIA 1999. "Parallel Protein Information Analysis System". <http://www.rwcp.or.jp/papia/>. *Computer Systems*. 1999: 15.
11. Buyya, R. (ed.). 2008. Grid Computing Info Centre. <http://www.Gridcomputing.com/>.
12. Baker, M. (ed.). 2008. "Grid Computing". IEEE DS Online. <http://computer.org/dsonline/gc/>.
13. Poster, I., Kesselman, C., and Globus. 1997. "A Metacomputing Infrastructure Toolkit". *International Journal of Supercomputer Applications*. 11(2):115–128.
14. Grimshaw, A. and Wulf, W. 1997. "The Legion Vision of a Worldwide Virtual Computer". *Communications of the ACM 1997*. 40(1) with HPDC 2001, 1 August 2000. Kluwer Academic Press, Pittsburgh, PA. 2000. buyya.com/gridsim/.
15. Buyya, R. 2008. "The Gridbus Toolkit: Enabling Grid Computing and Business". <http://www.gridbus.org>.
16. Sherwani, J., Ali, N., Lotia, N., Hayat, Z., Buyya, R. and Libra. 2002. "An Economy Driven Job Scheduling System for Clusters". Technical Report. The University of Melbourne: Melbourne, Australia.
17. Leinberger, W. and Kumar, V. 1999. "Information Power Grid: The New Frontier in Parallel Computing?". *IEEE Concurrency*. 7(4).

SUGGESTED CITATION

Boyinbode, O.K., R.O. Akinyede, and O.S. Adeola. 2009. A Library Grid Network for Enhancing Learning in Nigerian Universities". *Pacific Journal of Science and Technology*. 10(1):240-250.



[Pacific Journal of Science and Technology](http://www.pacificjournalofscienceandtechnology.com)