GRID ARCHITECTURE

A new architecture model and technology was developed for the establishment, management, and cross-organizational resource sharing within a virtual organization. This new architecture, called grid architecture, identifies the basic components of a grid system, defines the purpose and functions of such components and indicates how each of these components interacts with one another (Foster, Kesselman, & Tuecke). The main attention of the architecture is on the interoperability among resource providers and users to establish the sharing relationships. This interoperability means common protocols at each layer of the architecture model, which leads to the definition of a grid protocol architecture as shown in Figure 3.1. This protocol architecture defines common mechanisms, interfaces, schema, and protocols at each layer, by which users and resources can negotiate, establish, manage, and share resources.

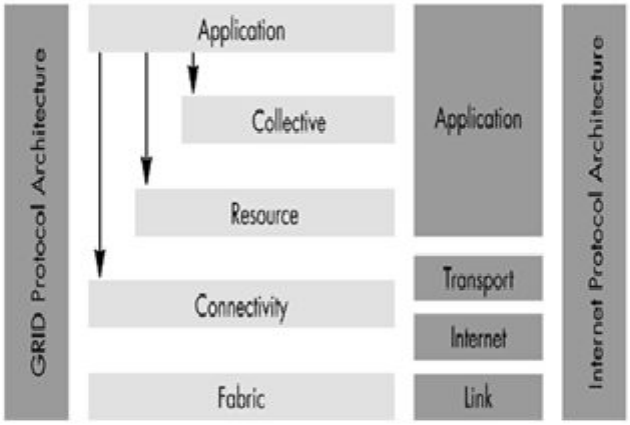

Figure 3.1 illustrates the component layers of the architecture with specific capabilities at each layer.

Each layer shares the behavior of the component layers described in the next discussion. As we can see in this illustration, each of these component layers is compared with their corresponding Internet protocol layers, for purposes of providing more clarity in their capabilities.
Now let us explore each of these layers in more detail.

Fabric Layer: Interface to Local Resources

The Fabric layer defines the resources that can be shared. This could include computational resources, data storage, networks, catalogs, and other system resources. These resources can be physical resources or logical resources by nature.

Typical examples of the logical resources found in a Grid Computing environment are distributed file systems, computer clusters, distributed computer pools, software applications, and advanced forms of networking services. These logical resources are implemented by their

own internal protocol (e.g., network file systems [NFS] for distributed file systems, and clusters using logical file systems [LFS]). These resources then comprise their own network of physical resources.

Although there are no specific requirements toward a particular resource that relates to integrating itself as part of any grid system, it is recommended to have two basic capabilities associated with the integration of resources. These basic capabilities should be considered as "best practices" toward Grid Computing disciplines. These best practices are as follows:

1. Provide an "inquiry" mechanism whereby it allows for the discovery against its own resource capabilities, structure, and state of operations. These are value-added features for resource discovery and monitoring.
2. Provide appropriate "resource management" capabilities to control the QoS the grid solution promises, or has been contracted to deliver. This enables the service provider to control a resource for optimal manageability, such as (but not limited to) start and stop activations, problem resolution, configuration management, load balancing, workflow, complex event correlation, and scheduling.

Connectivity Layer: Manages Communications

The Connectivity layer defines the core communication and authentication protocols required for grid-specific networking services transactions. Communications protocols, which include aspects of networking transport, routing, and naming, assist in the exchange of data between fabric layers of respective resources. The authentication protocol builds on top of the networking communication services in order to provide secure authentication and data exchange between users and respective resources.

The communication protocol can work with any of the networking layer protocols that provide the transport, routing, and naming capabilities in networking services solutions. The most commonly used Network layer protocol is the TCP/IP Internet protocol stack; however, this concept and discussion is not limited to that protocol. The authentication solution for virtual organization environments requires significantly more complex characteristics. The following describes the characteristics for consideration:

Single sign-on

This provides any multiple entities in the grid fabric to be authenticated once; the user can then access any available resources in the grid Fabric layer without further user authentication intervention.

Delegation

This provides the ability to access a resource under the current users permissions set; the resource should be able to relay the same user credentials (or a subset of the credentials) to other resources respective to the chain of access.

Integration with local resource specific security solutions

Each resource and hosting has specific security requirements and security solutions that match the local environment. This may include (for example) Kerberos security methods, Windows security methods, Linux security methods, and UNIX security methods. Therefore, in order to provide proper security in the grid fabric model, all grid solutions must provide integration with the local environment and respective resources specifically engaged by the security solution mechanisms.

User-based trust relationships

In Grid Computing, establishing an absolute trust relationship between users and multiple service providers is very critical. This accomplishes the environmental factor to which there is then no need of interaction among the providers to access the resources that each of them provide.

Data security

The data security topic is important in order to provide data integrity and confidentiality. The data passing through the Grid Computing solution, no matter what complications may exist, should be made secure using various cryptographic and data encryption mechanisms. These mechanisms are well known in the prior technological art, across all global industries.

Resource Layer: Sharing of a Single Resource

The Resource layer utilizes the communication and security protocols defined by the networking communications layer, to control the secure negotiation, initiation, monitoring,

metering, accounting, and payment involving the sharing of operations across individual resources.

The way this works is the Resource layer calls the Fabric layer functions in order to access and control the multitude of local resources. This layer only handles the individual resources and, hence, ignores the global state and atomic actions across the other resource collection, which in the operational context is the responsibility of the Collective layer. There are two primary classes of resource layer protocols. These protocols are key to the operations and integrity of any single resource. These protocols are as follows:

Information Protocols

These protocols are used to get information about the structure and the operational state of a single resource, including configuration, usage policies, service-level agreements, and the state of the resource. In most situations, this information is used to monitor the resource capabilities and availability constraints.

Management Protocols

The important functionalities provided by the management protocols are:

- Negotiating access to a shared resource is paramount. These negotiations can include the requirements on quality of service, advanced reservation, scheduling, and other key operational factors.
- Performing operation(s) on the resource, such as process creation or data access, is also a very important operational factor.
- Acting as the service/resource policy enforcement point for policy validation between a user and resource is critical to the integrity of the operations.
- Providing accounting and payment management functions on resource sharing is mandatory.
- Monitoring the status of an operation, controlling the operation including terminating the operation, and providing asynchronous notifications on operation status, is extremely critical to the operational state of integrity.

I t is recommended that these resource-level protocols should be minimal from a functional overhead point of view and they should focus on the functionality each provides from a utility aspect.

The Collective Layer: Coordinating Multiple Resources

While the Resource layer manages an individual resource, the Collective layer is responsible for all global resource management and interaction with a collection of resources. This layer of protocol implements a wide variety of sharing behaviors (protocols) utilizing a small number of Resource layer and Connectivity layer protocols.

Some key examples of the common, more visible collective services in a Grid Computing system are as follows:

Discovery Services

This enables the virtual organization participants to discover the existence and/or properties of that specific available virtual organization's resources.

Coallocation, Scheduling, and Brokering Services

These services allow virtual organization participants to request the allocation of one or more resources for a specific task, during a specific period of time, and to schedule those tasks on the appropriate resources.

Monitoring and Diagnostic Services

These services afford the virtual organizations resource failure recovery capabilities, monitoring of the networking and device services, and diagnostic services that include common event logging and intrusion detection. Another important aspect of this topic relates to the partial failure of any portion of a Grid Computing environment, in that it is critical to understand any and all business impacts related to this partial failure are well known, immediately, as the failure begins to occur—all the way through its corrective healing stages.

Data Replication Services

These services support the management aspects of the virtual organization's storage resources in order to maximize data access performance with respect to response time, reliability, and costs.

Grid-Enabled Programming Systems

These systems allow familiar programming models to be utilized in the Grid Computing environments, while sustaining various Grid Computing networking services. These networking services are integral to the environment in order to address resource discovery, resource

allocation, problem resolution, event correlation, network provisioning, and other very critical operational concerns related to the grid networks.

Workload Management Systems and Collaborative Frameworks

This provides multistep, asynchronous, multicomponent workflow management. This is a complex topic across several dimensions, yet a fundamental area of concern for enabling optimal performance and functional integrity.

Software Discovery Services

This provides the mechanisms to discover and select the best software implementation(s) available in the grid environment, and those available to the platform based on the problem being solved.

Community Authorization Servers

These servers control resource access by enforcing community utilization policies and providing these respective access capabilities by acting as policy enforcement agents.

Community Accounting and Payment Services

These services provide resource utilization metrics, while at the same time generating payment requirements for members of any community.

As we can observe based on the previous discussion, the capabilities and efficiencies of these Collective layer services are based on the underlying layers of the protocol stack. These collective networking services can be defined as general-purpose Grid Computing solutions to narrowed-domain and application-specific solutions. As an example, one such service is accounting and payment, which is most often very specific to the domain or application. Other notable and very specialized Collective layer services include schedulers, resource brokers, and workload managers (to name a few).

Application Layer: User-Defined Grid Applications

These are user applications, which are constructed by utilizing the services defined at each lower layer. Such an application can directly access the resource, or can access the resource through the Collective Service interface APIs (Application Provider Interface).

Each layer in the grid architecture provides a set of APIs and SDKs (software developer kits) for the higher layers of integration. It is up to the application developers whether they

should use the collective services for general-purpose discovery, and other high-level services across a set of resources, or if they choose to start directly working with the exposed resources. These user-defined grid applications are (in most cases) domain specific and provide specific solutions.

Solution : http://elearningatria.files.wordpress.com/2013/10/ise-viii-grid-computing-06is845-notes.pdf