system of systems: such as a cloud of clouds, a grid of clouds, or a cloud of grids, or inter-clouds as a basic SOA architecture.

Assignment Questions:

1. Explain the evolution of grid computing ?

2. Describe the multicore and multithreading architecture ?

3. Briefly write about GPU computing?

4. Write in detail about VMs & its operations , structures.

5. Describe in detail about the Grid infrastructure.

6. Describe the Cluster architecture in detail.

7. Write briefly about SOA and its evolution.

8. Explain in detail about grid architecture, its standards,and its elements.

UNIT II

Open Grid Services Architecture	Open Grid Services Architecture (OGSA) is a set of standards defining the way in which information is shared among diverse components of large, heterogeneous grid systems. In this context, a grid system is a scalable wide area network (WAN) that supports resource sharing and distribution. OGSA is a trademark of the Open Grid Forum.
Peer to Peer Computing.	Peer to Peer computing is a relatively new computing discipline in the realm of distributed computing. P2P system defines collaboration among a larger number of individuals and/or organizations, with a limited set of security requirements and a less complex resource-sharing topology.
SOA	A service-oriented architecture is intended to define loosely coupled and interoperable services/applications, and to define a process for integrating these interoperable components.
GRAM	GRAM provides resource allocation, process creation, monitoring, and management services. The most common use of GRAM is the remote job submission and control facility. GRAM simplifies the use of remote systems.
SOAP.	SOAP is a simple and lightweight XML-based mechanism for creating structured data packages that can be exchanged between network applications. SOAP provides a simple enveloping mechanism and is proven in being able to work with existing networking services technologies such as HHTP.SOAP is also flexible and extensible. SOAP is based on the fact that it builds upon the XML info set.
WSDL	WSDL is an XML Info set based document, which provides a model and XML format for describe web services. This enables services to be described and enables the client to consume these services in a standard way without knowing much on the lower level protocol exchange binding including SOAP and HTTP.

	This high level abstraction on the service limits human interaction and enables the automatic generation of proxies for web services, and these proxies can be static or dynamic. It allows both document and RPC - oriented messages.

Grid Architecture and Service Modeling

The grid is a meta computing infrastructure that brings together computers (PCs,workstations, server clusters, supercomputers, laptops, notebooks, mobile computers, PDAs, etc.) to form a large collection of compute, storage, and network resources to solve large-scale computation problems or to enable fast information retrieval by registered users or user groups. The coupling between hardware and software with special user applications is achieved by leasing the hardware, software, middleware, databases, instruments, and networks as computing utilities. Good examples include the renting of expensive special-purpose application software on demand and transparent access to human genome databases.

The goal of grid computing is to explore fast solutions for large-scale computing problems. This objective is shared by computer clusters and massively parallel processor (MPP) systems . However, grid computing takes advantage of the existing computing resources scattered in a nation or internationally around the globe. In grids, resources owned by different organizations are aggregated together and shared by many users in collective applications. Grids rely heavy use of LAN/WAN resources across enterprises, organizations, and governments. The virtual organizations or virtual supercomputers are new concept derived from grid or cloud computing. These are virtual resources dynamically configured and are not under the full control of any single user or local administrator.

Grid History and Service Families

Network-based distributed computing becomes more and more popular among the Internet users. Recall that the Internet was developed in the 1980s to provide computer-to-computer connections using the telnet:// protocol. The web service was developed in the 1990s to establish direct linkage between web pages using the http:// protocol. Ever since the 1990s, grids became gradually available to establish large pools of shared resources. The approach is to link many Internet applications across machine platforms directly in order to eliminate isolated resource islands. We may invent upgraded protocols in the future like ¯grid:// and ¯cloud:// to realize this dream of a socialized cyberspace with greater resource sharing.

The idea of the grid was pioneered by Ian Foster, Carl Kesselman and Steve Tuecke in a 2001 paper . With is ground work, they are often recognized as the fathers of the grids. The Globus Project supported by DARPA has promoted the maturity of grid technology with a rich collection of software and middleware tools for grid computing. In 2007, the concept of cloud computing was thrown out, which in many ways was extending grid computing through virtualized data centers. In this beginning section, we introduce major grid families and review the grid service evolution over the past 15 years.

Grids differ from conventional HPC clusters. Cluster nodes are more homogeneousmachines that are better coordinated to work collectively and cooperatively. The grid nodes are heterogeneous computers that are more loosely coupled together over geographically dispersed sites. In 2001, Forbes Magazine advocated the emergence of the great global grid (GGG) as a new global infrastructure. This GGG evolved from the World Wide Web (WWW) technology we have enjoyed for many years.

Four Grid Service Families

Most of today's grid systems are called computational grids or data grids. Good examples are the NSF TeraGrid installed in the United States and the DataGrid built in the European Union. Information or knowledge grids post another grid class dedicated to knowledge management and distributed ontology processing. The Semantic web, also known as semantic grids, belongs to this faimly. Ontology platform falls into information or knowledge grids. Other information/knowledge grids include the Berkeley BOINC and NASA's Information Power Grid.

In the business world, we see a family, called business grids, built for business data/information processing. These are represented by the HP eSpeak, IBM WebSphere, Microsoft .NET, and Sun One systems. Some business grids are being transformed into Internet clouds. The last grid class includes several grid extensions such as P2P grids and parasitic grids. This will concentrate mainly in computational or data grids. Business grids are only briefly introduced.

Grid Service Protocol Stack

To put together the resources needed in a grid platform, a layered grid architecture . The top layer corresponds to user applications to run on the grid system. The user applications demand collective services including collective computing and communications. The next layer is formed by the hardware and software resources aggregated to run the user applications under the collective operations. The connectivity layer provides the interconnection among drafted resources. This connectivity could be established directly on physical networks or it could be built with virtual networking technology.

The layered grid service protocols and their relationship with the Internet service protocols. Courtesy of Foster, Kesselman, and Tuecke .The connectivity must support the grid fabric, including the network inks and virtual private channels. The fabric layer includes all computational resources, storage systems, catalogs, network resources, sensors, and their network connections. The connectivity layer enables the exchange of data between fabric layer resources. The five-layer grid architecture is closely related to the layered Internet protocol stack . The fabric layer corresponds to the link layer in the Internet stack. The connectivity layer is supported by the network and transport layers of the Internet stack. The Internet application layer supports the top three layers.

Grid Resources

It summarizes typical resources that are required to perform grid computing.Many existing protocols (IP, TCP, HTTP, FTP, and DNS) or some new communication protocols can be used to route and transfer data. The resource layer is responsible for sharing single resources. An interface is needed to claim the static structure and dynamic status of local resources. The grid should be able to accept resource requests, negotiate the Quality of Service (QoS), and perform the operations specified in user applications.

The collective layer handles the interactions among a collection of resources. This layer implements functions such as resource discovery, co-allocation, scheduling, brokering, monitoring, and diagnostics. Other desired features include replication, grid-enabled programming, workload management, collaboration, software discovery, access authorization, and community accounting and payment. The application layer comprises mainly user applications. The applications interact with components in other layers by using well-defined APIs (application programming interfaces) and SDKs (software development kits).

CPU Scavenging and Virtual Supercomputers

The process of grid resource aggregation from local and remote sources. Then we link the grid to the concept of virtual organizations in a dynamic sense. In fact, the distinction between grids and clouds becomes blurred in recent years. Traditionally, grids were formed with allocated resources statically, while clouds were formed with provisioned resources dynamically. As virtualization is applicable to grid components, some grids involving data centers become more like clouds.

Foster, et al. [15] have compared the grid problem with the anatomy problem in biology. The application users expect grids to be designed as flexible, secure, and coordinated resources shared by individuals, institutions, and virtual organizations. The grid resources could come from two possible sources. On the one hand, large-scale HPC grids can be formed with computers from resource-rich supercomputer centers owned by government agencies and research institutions. Alternatively, one could form ¯virtual grids, casually, out of a large number of small commodity computers owned by ordinary citizens, who volunteer to share their free cycles with other users for a noble cause.

CPU Scavenging and Virtual Supercomputers

Both public and virtual grids can be built over large or small machines, that are loosely coupled together to satisfy the application need. Grids differ from the conventional supercomputers in many ways in the context of distributed computing. Supercomputers like MPPs in the Top-500 list are more homogeneously structured with tightly coupled operations, while the grids are built with heterogeneous nodes running non-interactive workloads. These grid workloads may involve a large number of files and individual users. The geographically dispersed grids are more scalable and fault-tolerant with significantly lower operational costs than the supercomputers.The concept of creating a ¯grid from the unused resources in a network of computers is known as CPU scavenging. In reality, virtual grids are built over large number of desktop computers by using their free cycles at night or during inactive usage periods. The donors are ordinary citizens on a voluntary participation basis. In practice, these client hosts also donate some disk space, RAM, and network bandwidth in addition to the raw CPU cycles. At present, many volunteer computing grids are built using the CPU scavenging model. The most famous example is the SETI@Home , which applied over 3 million computers to achieve 23.37 TFlpos as of Sept. 2001. More recent examples include the BOINC and Folding@Home , etc. In practice, these virtual grids can be viewed as virtual supercomputers.

Grid Resource Aggregation

During the resource aggregation process for grids or clouds, several assumptions are made. First, the compute nodes and other necessary resources for grids do not join or leave the system incidentally, except when some serious faults occur in the grid. Second, cloud resources are mostly provisioned from large data centers. Since security and reliability are very tight in these data centers, resource behavior is not predictable. Third, although resources in P2P systems are casually allocated, we can build P2P grids for distributed file sharing, content delivery, gaming, and entertainment applications. The joining or leaving of some peers has little impact on the needed functions of a P2P grid system.

We envision the grid resource aggregation process in a global setting. Hardware, software, database, and network resources are denoted by R's and are scattered all over the world. The availability and specification of these open resources is provided by Grid Information Service (GIS) agencies. The grid resource brokers assist users with fees to allocate available resources. Multiple brokers could compete to serve users. Also, multiple GISes may overlap in their resource coverage. New grid applications are enabled after the coupling of computer databases, instruments, and human operators needed in their specific applications. It should be noted that today's grid computing applications are no longer restricted to using HPC systems. HTC systems, like clouds, are even more in demand in business services.

Virtual Organization

The grid is a distributed system integrated from shared resources to form a virtual organization(VO). The VO offers dynamic cooperation built over multiple physical organizations. The virtual resources contributed by these real organizations are managed autonomously. The grid must deal with the trust relationship in a VO. The applications in a grid vary in terms of workload and resource demand. A flexible grid system should be designed to adapt to varying workloads. In reality, physical organizations include a real company, a university, or a branch of government. These real organizations often share some common objectives.

For example, several research institutes and hospitals may undertake some joint research challenges together to explore a new cancer drug. Another concrete example is the joint venture among IBM, Apple, and Motorola to develop PowerPC processors and their supporting software in the past. The joint venture was based on the VO model. Grids definitely can promote the concept of VOs. Still, joint ventures demand resources and labor from all participants. The following example shows how two VOs or grid configurations can be formed out of three physical organizations.

Open Grid Services Architecture (OGSA)

The OGSA is an open source grid service standard jointly developed by academia and the IT industry under coordination of a working group in the Global Grid Forum (GGF). The standard was specifically developed for the emerging grid and cloud service communities. The OGSA is extended from web service concepts and technologies. The standard defines a common framework that allows businesses to build grid platforms across enterprises and business partners. The intent is to define the standards required for both open source and commercial software to support a global grid infrastructure.

OGSA Framework

The OGSA was built on two basic software technologies: the Globus Toolkit widely adopted as a grid technology solution for scientific and technical computing, and web services (WS 2.0) as a popular standards-based framework for business and network applications. The OGSA is intended to support the creation, termination, management, and invocation of stateful, transient grid services via standard interfaces and conventions . The OGSA framework specifies the physical environment, security, infrastructure profile, resource provisioning, virtual domains, and execution environment for various grid services and API access tools.

A service is an entity that provides some capability to its client by exchanging messages. We feel that greater flexibility is needed in grid service discovery and management. The service-oriented architecture (SOA) presented serves as the foundation of grid computing services. The individual and collective states of resources are specified in this service standard. The standard also specifies interactions between these services within the particular SOA for grids. An important point is that the architecture is not layered, where the implementation of one service is built upon modules that are logically dependent. One may classify this framework as object-oriented. Many web service standards, semantics, and extensions are applied or modified in the OGSA.

OGSA Interfaces

The OGSA is centered on grid services. These services demand special well-defined application interfaces. These interfaces provide resource discovery, dynamic service creation, lifetime management, notification, and manageability. The conventions must address naming and upgradeability. The interfaces proposed by the OGSA working group. While the OGSA defines a variety of behaviors and associated interfaces, all but one of these interfaces (the grid service) is optional. Two key properties of a grid service are transience and statefulness. These properties have significant implications regarding how a grid service is named, discovered, and managed. Being transient means the service can be created and destroyed dynamically; statefulness refers to the fact that one can distinguish one service instance from another.

OGSA Grid Service Interfaces Developed by the OGSA Working Group

Grid Service Handle

A GSH is a globally unique name that distinguishes a specific grid service instance from all others. The status of a grid service instance could be that it exists now or that it will exist in the future. These instances carry no protocol or instance-specific addresses or supported protocol bindings. Instead, these information items are encapsulated along with all other instance-specific information. In order to interact with a specific service instance, a single abstraction is defined as a GSR. Unlike a GSH, which is time-invariant, the GSR for an instance can change over the lifetime of the service. The OGSA employs a ¯handle-resolution mechanism for mapping from a GSH to a GSR. The GSH must be globally defined for a particular instance. However, the GSH may not always refer to the same network address. A service instance may be implemented in its own way, as long as it obeys the associated semantics. For example, the port type on which the service instance was implemented decides which operation to perform

.Grid Service Migration

This is a mechanism for creating new services and specifying assertions regarding the lifetime of a service. The OGSA model defines a standard interface, known as a factor, to implement this reference. Any service that is created must address the former services as the reference of later services. The factory interface is labeled as a Create Service operation . This creates a requested grid service with a specified interface and returns the GSH and initial GSR for the new service instance. It should also register the new service instance with a handle resolution service. Each dynamically created grid service instance is associated with a specified lifetime.

Grid Service Migration Using GSH and GSR shows how a service instance may migrate from one location to another during execution. A GSH resolves to a different GSR for a migrated service instance before (on the left) and after (on the right) the migration at time T. The handle resolver simply returns different GSRs before and after the migration. The initial lifetime can be extended by a specified time period by explicitly requesting the client or another grid service acting on the client's behalf.

A GSH resolving to a different GSR for a migrated service instance before (shown on the left) and after (on the right) the migration at time T. If the time period expires without having received a reaffirmed interest from a client, the service instance can be terminated on its own and release the associated resources accordingly. The lifetime management enables robust termination and failure detection. This is done by clearly defining the lifetime semantics of a service instance. Similarly, a hosting environment is guaranteed to consume bounded resources under some system failures. If the termination time of a service is reached, the hosting environment can reclaim all resources allocated.

OGSA Security Models

The OGSA supports security enforcement at various levels. The grid works in a heterogeneous distributed environment, which is essentially open to the general public. We must be able to detect intrusions or stop viruses from spreading by implementing secure conversations, single logon, access control, and auditing for nonrepudiation. At the security policy and user levels, we want to apply a service or endpoint policy, resource mapping rules, authorized access of critical resources, and privacy protection. At the Public Key Infrastructure (PKI) service level, the OGSA demands security binding with the security protocol stack and bridging of certificate authorities (CAs), use of multiple trusted intermediaries, and so on. Trust models and secure logging are often practiced in grid platforms.

The OGSA security model implemented at various protection levels. Courtesy of I. Foster, et al., http://www.ogf.org/documents/GFD.80.pdf

DATA-INTENSIVE GRID SERVICE MODELS

Applications in the grid are normally grouped into two categories: computation-intensive and data-intensive. For data-intensive applications, we may have to deal with massive amounts of data. For example, the data produced annually by a Large Hadron Collider may exceed several petabytes (1015 bytes). The grid system must be specially designed to discover, transfer, and manipulate these massive data sets. Transferring massive data sets is a time-consuming task. Efficient data management deman ds low-cost storage and high-speed data movement. Listed in the following paragraphs are several common methods for solving data movement problems.

Data Replication and Unified Namespace

This data access method is also known as caching, which is often applied to enhance data efficiency in a grid environment. By replicating the same data blocks and scattering them in multiple regions of a grid, users can access the same data with locality of references.

Furthermore, the replicas of the same data set can be a backup for one another. Some key data will not be lost in case of failures. However, data replication may demand periodic consistency checks. The increase in storage requirements and network bandwidth may cause additional problems. Replication strategies determine when and where to create a replica of the data. The factors to consider include data demand, network conditions, and transfer cost. The strategies of replication can be classified into method types: dynamic and static. For the static method, the locations and number of replicas are determined in advance and will not be modified. Although replication operations require little overhead, static strategies cannot adapt to changes in demand, bandwidth, and storage vailability. Dynamic strategies can adjust locations and number of data replicas according to changes inconditions (e.g., user behavior). However, frequent data-moving operations can result in much more overhead than in static strategies. The replication strategy must be optimized with respect to the status of data replicas. For static replication, optimization is required to determine the location and number of data replicas. For dynamic replication, optimization may be determined based on whether the data replica is being created, deleted, or moved. The most common replication strategies include preserving locality, minimizing update costs, and maximizing profits.

Grid Data Access Models

Multiple participants may want to share the same data collection. To retrieve any piece of data, we need a grid with a unique global namespace. Similarly, we desire to have unique file names. To achieve these, we have to resolve inconsistencies among multiple data objects bearing the same name. Access restrictions may be imposed to avoid confusion. Also, data needs to be protected to avoid leakage and damage. Users who want to access data have to be authenticated first and then authorized for access. In general, there are four access models for organizing a data grid, as listed here and shown in Figure 7.5. Four architectural models for building a data grid. Monadic model: This is a centralized data repository model. All the data is saved in a central data repository. When users want to access some data they have to submit requests directly to the central repository. No data is replicated for preserving data locality. This model is the simplest to implement for a small grid. For a large grid, this model is not efficient in terms of performance and reliability. Data replication is permitted in this model only when fault tolerance is demanded.

Hierarchical model: The hierarchical model, is suitable for building a large data grid which has only one large data access directory. The data may be transferred from the source to a second-level center. Then some data in the regional center is transferred to the third-level center. After being forwarded several times, specific data objects are accessed directly by users. Generally speaking, a higher-level data center has a wider coverage area. It provides higher bandwidth for access than a lower-level data center. PKI security services are easier to implement in this hierarchical data access model. The European Data Grid (EDG) adopts this data access model.

Federation model: This data access model is better suited for designing a data grid with multiple sources of data supplies. Sometimes this model is also known as a mesh model. The data sources are distributed to many different locations. Although the data is shared, the data items are still owned and controlled by their original owners. According to predefined access policies, only authenticated users are authorized to request data from any data source. This mesh model may cost the most when the number of grid institutions becomes very large.

Hybrid model: This is data access model . The model combines the best features of the hierarchical and mesh models. Traditional data transfer technology, such as FTP, applies for networks with lower bandwidth. Network links in a data grid often have fairly high bandwidth, and other data transfer models are exploited by high-speed data transfer tools such as GridFTP developed with the Globus library. The cost of the hybrid model can be traded off between the two extreme models for hierarchical and mesh-connected grids.

Overview of Grid'5000 located at nine resource sites in France.

Parallel versus Striped Data Transfers

Compared with traditional FTP data transfer, parallel data transfer opens multiple data streams for passing subdivided segments of a file simultaneously. Although the speed of each stream is the same as in sequential streaming, the total time to move data in all streams can be significantly reduced compared to FTP transfer. In striped data transfer, a data object is partitioned into a number of sections, and each section is placed in an individual site in a data grid. When a user requests this piece of data, a data stream is created for each site, and all the sections of data objects are transferred simultaneously. Striped data transfer can utilize the bandwidths of multiple sites more efficiently to speed up data transfer.

Grid Projects and Grid Systems Built

Grid computing provides promising solutions to contemporary users who want to effectively share and collaborate with one another in distributed and self-governing environments. Apart from volunteer grids, most large-scale grids are national or international projects funded by public agencies. This section reviews the major grid systems developed in recent years. In particular, we describe three national grid projects that have been installed in the U.S., EU, and China.

National Grids and International Projects

Like supercomputers, national grids are mainly funded through government sources.These national grids are developed to promote research discovery, middleware products, and utility computing in grid-enabled applications.

National Grid Projects