INTRODUCTION TO PARALLEL AND DISTRIBUTED COMPUTING

The simultaneous growth in availability of big data and in the number of simultaneous users on the Internet places particular pressure on the need to carry out computing tasks "in parallel," or simultaneously. Parallel and distributed computing occurs across many different topic areas in computer science, including algorithms, computer architecture, networks, operating systems, and software engineering. During the early 21st century there was explosive growth in multiprocessor design and other strategies for complex applications to run faster. Parallel and distributed computing builds on fundamental systems concepts, such as concurrency, mutual exclusion, consistency in state/memory manipulation, message-passing, and shared-memory models.

Parallel Computing:

In parallel computing multiple processors performs multiple tasks assigned to them simultaneously. Memory in parallel systems can either be shared or distributed. Parallel computing provides concurrency and saves time and money.

Distributed Computing:

In distributed computing we have multiple autonomous computers which seems to the user as single system. In distributed systems there is no shared memory and computers communicate with each other through message passing. In distributed computing a single task is divided among different computers.

Difference between Parallel Computing and Distributed Computing:

S.NO	PARALLEL COMPUTING	DISTRIBUTED COMPUTING
1.	Many operations are performed simultaneously	System components are located at different locations

2.	Single computer is required	Uses multiple computers
3.	Multiple processors perform multiple operations	Multiple computers perform multiple operations
4.	It may have shared or distributed memory	It have only distributed memory
5.	Processors communicate with each other through bus	Computer communicate with each other through message passing.
6.	Improves the system performance	Improves system scalability, fault tolerance and resource sharing capabilities

Parallel Computing – It is the use of multiple processing elements simultaneously for solving any problem. Problems are broken down into instructions and are solved concurrently as each resource which has been applied to work is working at the same time.

Advantages of Parallel Computing over Serial Computing are as follows:

1. It saves time and money as many resources working together will reduce the time and cut potential costs.
2. It can be impractical to solve larger problems on Serial Computing.
3. It can take advantage of non-local resources when the local resources are finite.
4. Serial Computing 'wastes' the potential computing power, thus Parallel Computing makes better work of hardware.

Types of Parallelism:

1. **Bit-level parallelism:** It is the form of parallel computing which is based on the increasing processor's size. It reduces the number of instructions that the system must execute in order to perform a task on large-sized data.
   Example: Consider a scenario where an 8-bit processor must compute the sum of two 16-bit integers. It must first sum up the 8 lower-order bits, then add the 8 higher-order bits, thus requiring two instructions to perform the operation. A 16-bit processor can perform the operation with just one instruction.

2. **Instruction-level parallelism:** A processor can only address less than one instruction for each clock cycle phase. These instructions can be re-ordered and grouped which are later on executed concurrently without affecting the result of the program. This is called instruction-level parallelism.
3. **Task Parallelism:** Task parallelism employs the decomposition of a task into subtasks and then allocating each of the subtasks for execution. The processors perform execution of sub tasks concurrently.

Why parallel computing?

- The whole real world runs in dynamic nature i.e. many things happen at a certain time but at different places concurrently. This data is extensively huge to manage.
- Real world data needs more dynamic simulation and modeling, and for achieving the same, parallel computing is the key.
- Parallel computing provides concurrency and saves time and money.
- Complex, large datasets, and their management can be organized only and only using parallel computing's approach.
- Ensures the effective utilization of the resources. The hardware is guaranteed to be used effectively whereas in serial computation only some part of hardware was used and the rest rendered idle.
- Also, it is impractical to implement real-time systems using serial computing.

Applications of Parallel Computing:

- Data bases and Data mining.
- Real time simulation of systems.
- Science and Engineering.
- Advanced graphics, augmented reality and virtual reality.

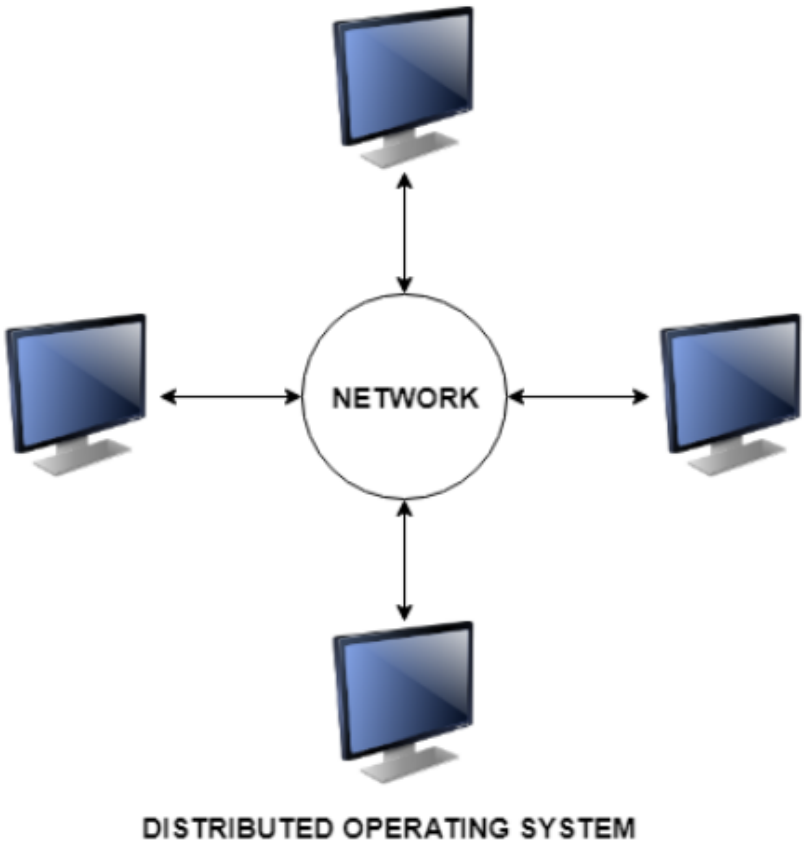
Limitations of Parallel Computing:

- It addresses such as communication and synchronization between multiple sub-tasks and processes which is difficult to achieve.
- The algorithms must be managed in such a way that they can be handled in the parallel mechanism.
- The algorithms or program must have low coupling and high cohesion. But it's difficult to create such programs.
- More technically skilled and expert programmers can code a parallelism based program well.

Future of Parallel Computing: The computational graph has undergone a great transition from serial computing to parallel computing. Tech giant such as Intel has already taken a step towards parallel computing by employing multicore processors. Parallel computation will revolutionize the way computers work in the future, for the better good. With all the world connecting to each other even more than before, Parallel Computing does a better role in helping us stay that way. With faster networks, distributed systems, and multi-processor computers, it becomes even more necessary.

A distributed system contains multiple nodes that are physically separate but linked together using the network. All the nodes in this system communicate with each other and handle processes in tandem. Each of these nodes contains a small part of the distributed operating system software.

A diagram to better explain the distributed system is −



Types of Distributed Systems

The nodes in the distributed systems can be arranged in the form of client/server systems or peer to peer systems. Details about these are as follows −

Client/Server Systems

In client server systems, the client requests a resource and the server provides that resource. A server may serve multiple clients at the same time while a client is in contact with only one server. Both the client and server usually communicate via a computer network and so they are a part of distributed systems.

Peer to Peer Systems

The peer to peer systems contains nodes that are equal participants in data sharing. All the tasks are equally divided between all the nodes. The nodes interact with each other as required as share resources. This is done with the help of a network.

Advantages of Distributed Systems

Some advantages of Distributed Systems are as follows −

- All the nodes in the distributed system are connected to each other. So nodes can easily share data with other nodes.
- More nodes can easily be added to the distributed system i.e. it can be scaled as required.
- Failure of one node does not lead to the failure of the entire distributed system. Other nodes can still communicate with each other.
- Resources like printers can be shared with multiple nodes rather than being restricted to just one.

Disadvantages of Distributed Systems

Some disadvantages of Distributed Systems are as follows −

- It is difficult to provide adequate security in distributed systems because the nodes as well as the connections need to be secured.
- Some messages and data can be lost in the network while moving from one node to another.
- The database connected to the distributed systems is quite complicated and difficult to handle as compared to a single user system.
- Overloading may occur in the network if all the nodes of the distributed system try to send data at once.

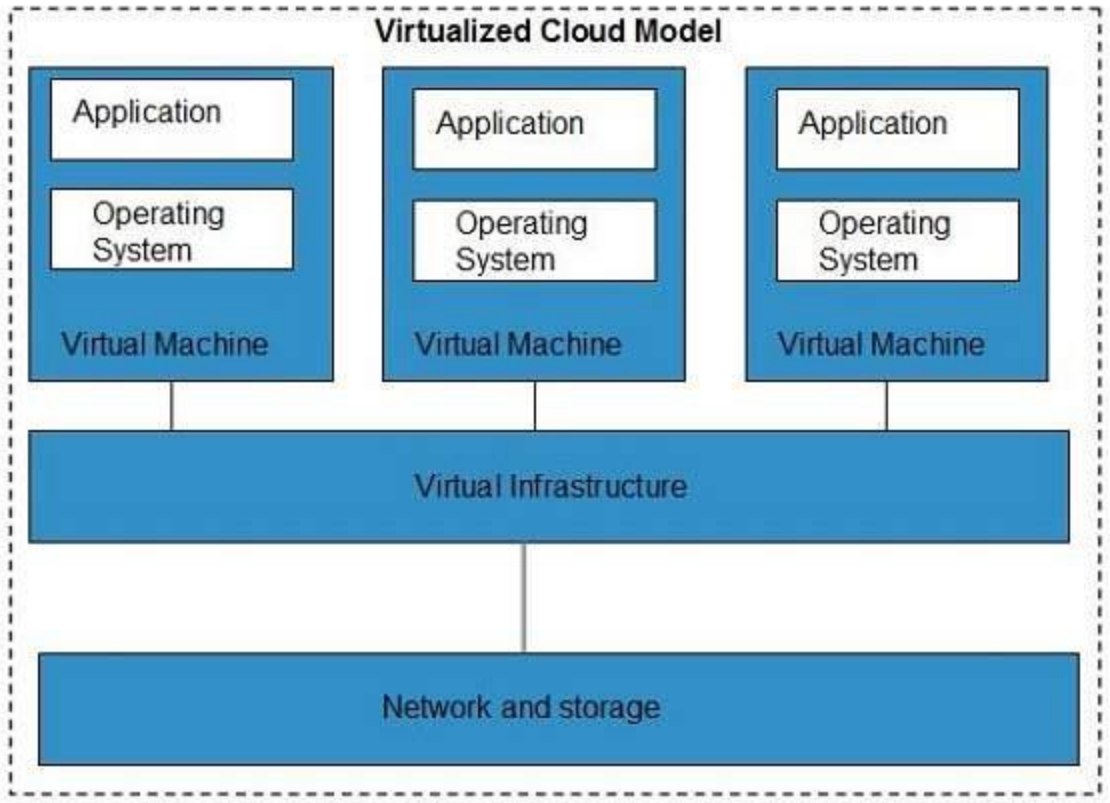
DISTRIBUTED COMPUTING MODELS

There are certain technologies working behind the cloud computing platforms making cloud computing flexible, reliable, and usable. These technologies are listed below:

- Virtualization
- Service-Oriented Architecture (SOA)
- Grid Computing
- Utility Computing

Virtualization

Virtualization is a technique, which allows to share single physical instance of an application or resource among multiple organizations or tenants (customers). It does this by assigning a logical name to a physical resource and providing a pointer to that physical resource when demanded.

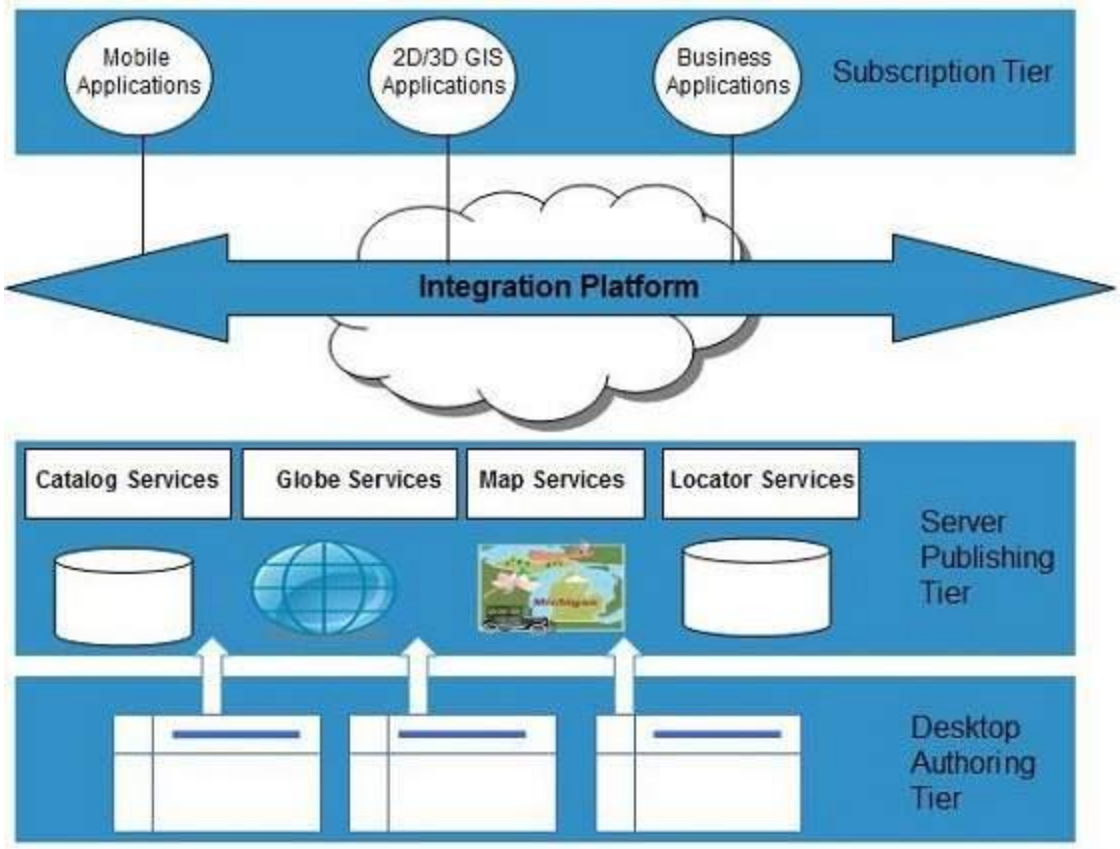


The **Multitenant** architecture offers **virtual isolation** among the multiple tenants. Hence, the organizations can use and customize their application as though they each have their instances running.

Service-Oriented Architecture (SOA)

Service-Oriented Architecture helps to use applications as a service for other applications regardless the type of vendor, product or technology. Therefore, it is possible to exchange the data between applications of different vendors without additional programming or making changes to services.

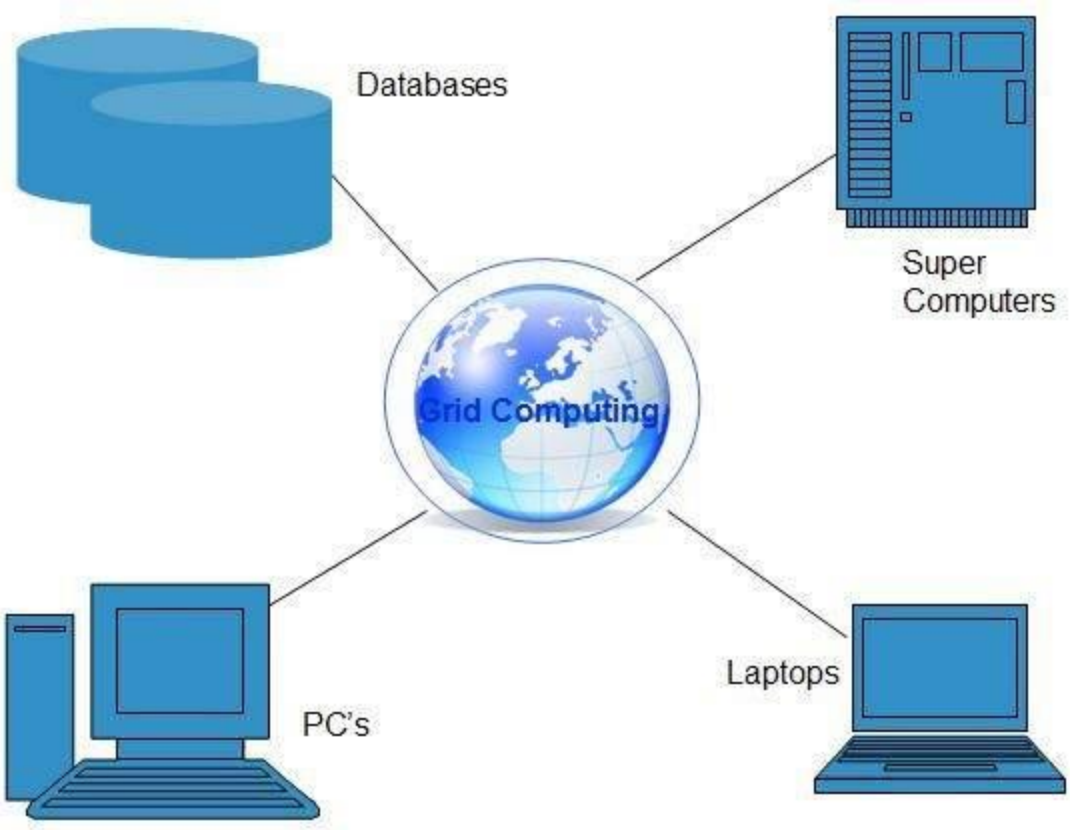
The cloud computing service oriented architecture is shown in the diagram below.



Grid Computing

Grid Computing refers to distributed computing, in which a group of computers from multiple locations are connected with each other to achieve a common objective. These computer resources are heterogeneous and geographically dispersed.

Grid Computing breaks complex task into smaller pieces, which are distributed to CPUs that reside within the grid.



Utility Computing

Utility computing is based on **Pay-per-Use model.** It offers computational resources on demand as a metered service. Cloud computing, grid computing, and managed IT services are based on the concept of utility computing.

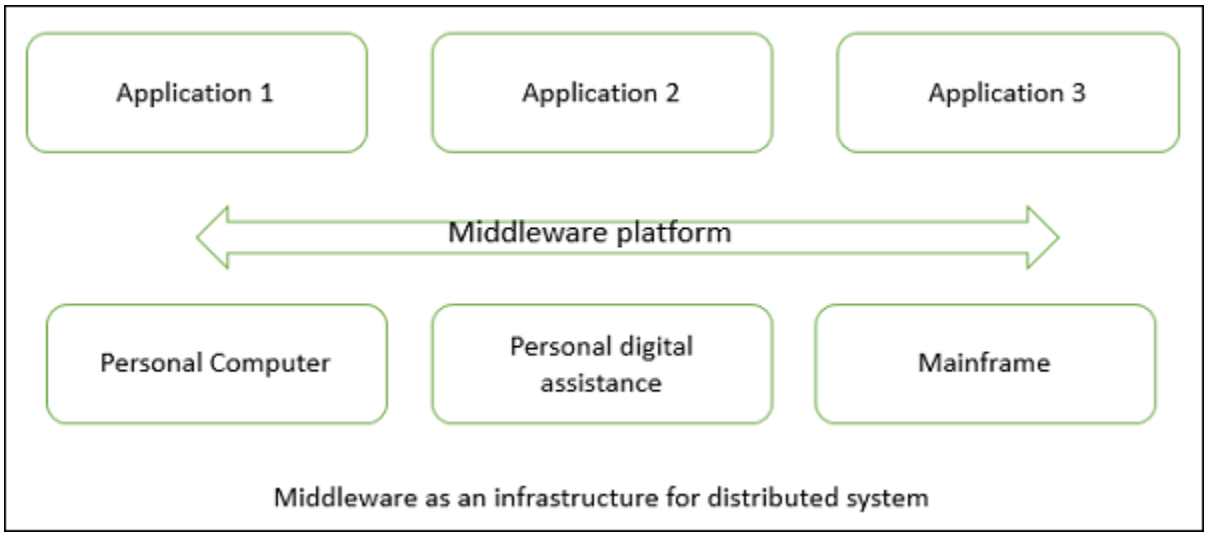
DISTRIBUTED COMPUTING MODELS

In distributed architecture, components are presented on different platforms and several components can cooperate with one another over a communication network in order to achieve a specific objective or goal.

- In this architecture, information processing is not confined to a single machine rather it is distributed over several independent computers.
- A distributed system can be demonstrated by the client-server architecture which forms the base for multi-tier architectures; alternatives are the broker architecture such as CORBA, and the Service-Oriented Architecture (SOA).

- There are several technology frameworks to support distributed architectures, including .NET, J2EE, CORBA, .NET Web services, AXIS Java Web services, and Globus Grid services.
- Middleware is an infrastructure that appropriately supports the development and execution of distributed applications. It provides a buffer between the applications and the network.
- It sits in the middle of system and manages or supports the different components of a distributed system. Examples are transaction processing monitors, data convertors and communication controllers etc.

Middleware as an infrastructure for distributed system



Middleware as an infrastructure for distributed system

The basis of a distributed architecture is its transparency, reliability, and availability.

The following table lists the different forms of transparency in a distributed system −

Sr.No.	Transparency & Description
1	**Access** Hides the way in which resources are accessed and the differences in data platform.
2	**Location** Hides where resources are located.
3	**Technology**

	Hides different technologies such as programming language and OS from user.
4	**Migration / Relocation** Hide resources that may be moved to another location which are in use.
5	**Replication** Hide resources that may be copied at several location.
6	**Concurrency** Hide resources that may be shared with other users.
7	**Failure** Hides failure and recovery of resources from user.
8	**Persistence** Hides whether a resource (software) is in memory or disk.

Advantages

- **Resource sharing** − Sharing of hardware and software resources.
- **Openness** − Flexibility of using hardware and software of different vendors.
- **Concurrency** − Concurrent processing to enhance performance.
- **Scalability** − Increased throughput by adding new resources.
- **Fault tolerance** − The ability to continue in operation after a fault has occurred.

Disadvantages

- **Complexity** − They are more complex than centralized systems.
- **Security** − More susceptible to external attack.
- **Manageability** − More effort required for system management.
- **Unpredictability** − Unpredictable responses depending on the system organization and network load.

Centralized System vs. Distributed System

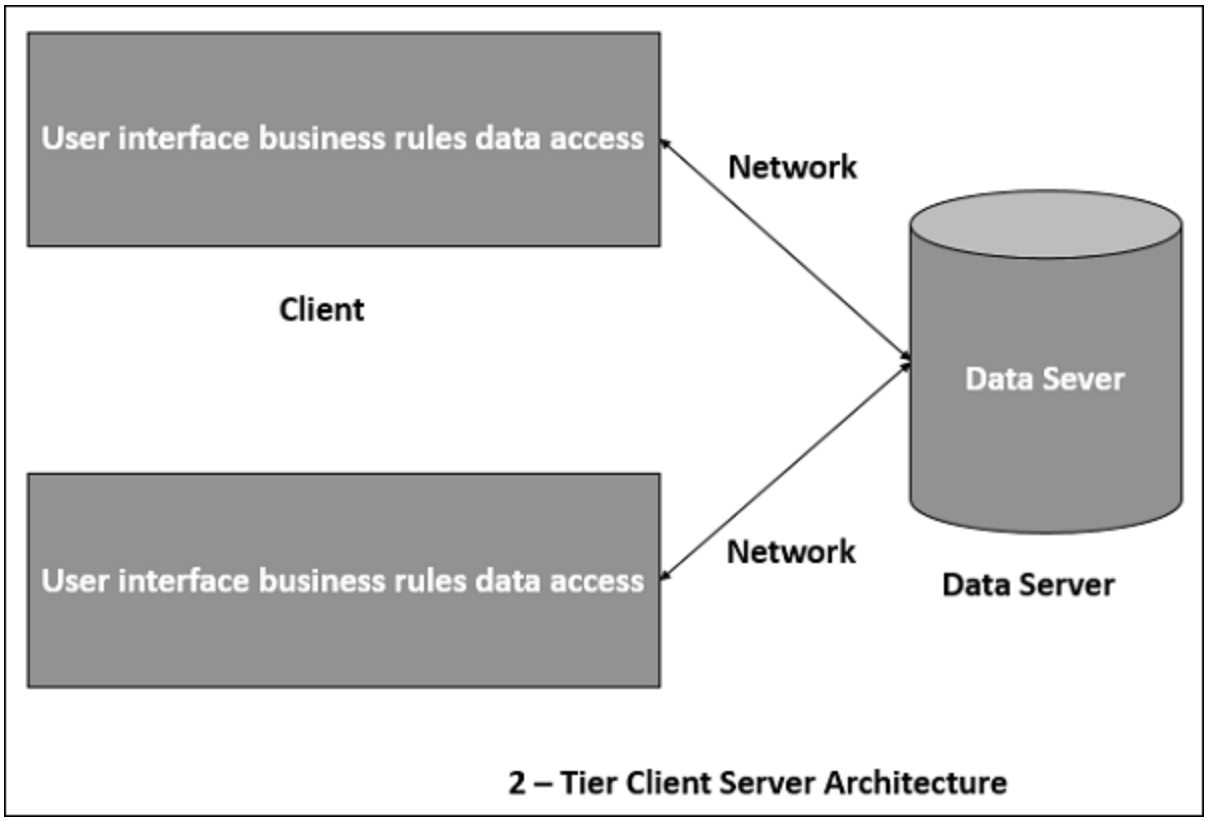
Criteria	Centralized system	Distributed System
Economics	Low	High
Availability	Low	High
Complexity	Low	High
Consistency	Simple	High
Scalability	Poor	Good
Technology	Homogeneous	Heterogeneous
Security	High	Low

Client-Server Architecture

The client-server architecture is the most common distributed system architecture which decomposes the system into two major subsystems or logical processes −

- **Client** − This is the first process that issues a request to the second process i.e. the server.
- **Server** − This is the second process that receives the request, carries it out, and sends a reply to the client.

In this architecture, the application is modelled as a set of services that are provided by servers and a set of clients that use these services. The servers need not know about clients, but the clients must know the identity of servers, and the mapping of processors to processes is not necessarily 1 : 1



Client-server Architecture can be classified into two models based on the functionality of the client −

Thin-client model

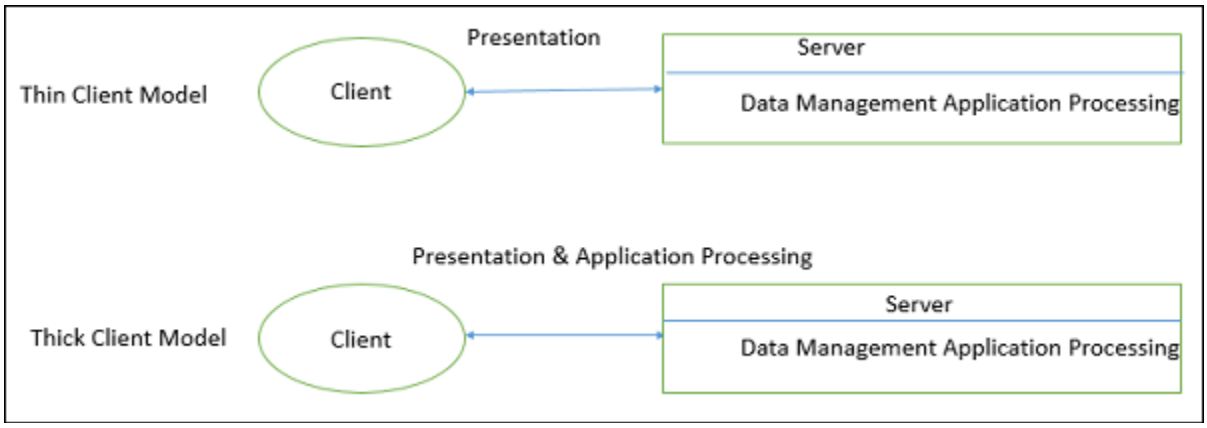
In thin-client model, all the application processing and data management is carried by the server. The client is simply responsible for running the presentation software.

- Used when legacy systems are migrated to client server architectures in which legacy system acts as a server in its own right with a graphical interface implemented on a client
- A major disadvantage is that it places a heavy processing load on both the server and the network.

Thick/Fat-client model

In thick-client model, the server is only in charge for data management. The software on the client implements the application logic and the interactions with the system user.

- Most appropriate for new C/S systems where the capabilities of the client system are known in advance
- More complex than a thin client model especially for management. New versions of the application have to be installed on all clients.



Advantages

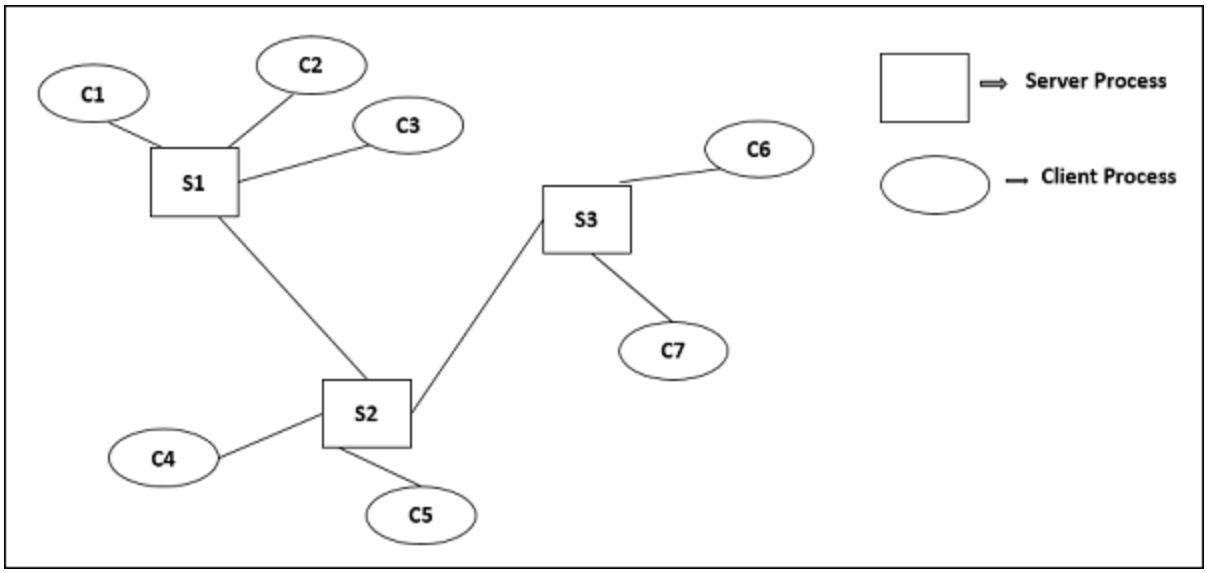
- Separation of responsibilities such as user interface presentation and business logic processing.
- Reusability of server components and potential for concurrency
- Simplifies the design and the development of distributed applications
- It makes it easy to migrate or integrate existing applications into a distributed environment.
- It also makes effective use of resources when a large number of clients are accessing a high-performance server.

Disadvantages

- Lack of heterogeneous infrastructure to deal with the requirement changes.
- Security complications.
- Limited server availability and reliability.
- Limited testability and scalability.
- Fat clients with presentation and business logic together.

Multi-Tier Architecture (n-tier Architecture)

Multi-tier architecture is a client–server architecture in which the functions such as presentation, application processing, and data management are physically separated. By separating an application into tiers, developers obtain the option of changing or adding a specific layer, instead of reworking the entire application. It provides a model by which developers can create flexible and reusable applications.



The most general use of multi-tier architecture is the three-tier architecture. A three-tier architecture is typically composed of a presentation tier, an application tier, and a data storage tier and may execute on a separate processor.

Presentation Tier

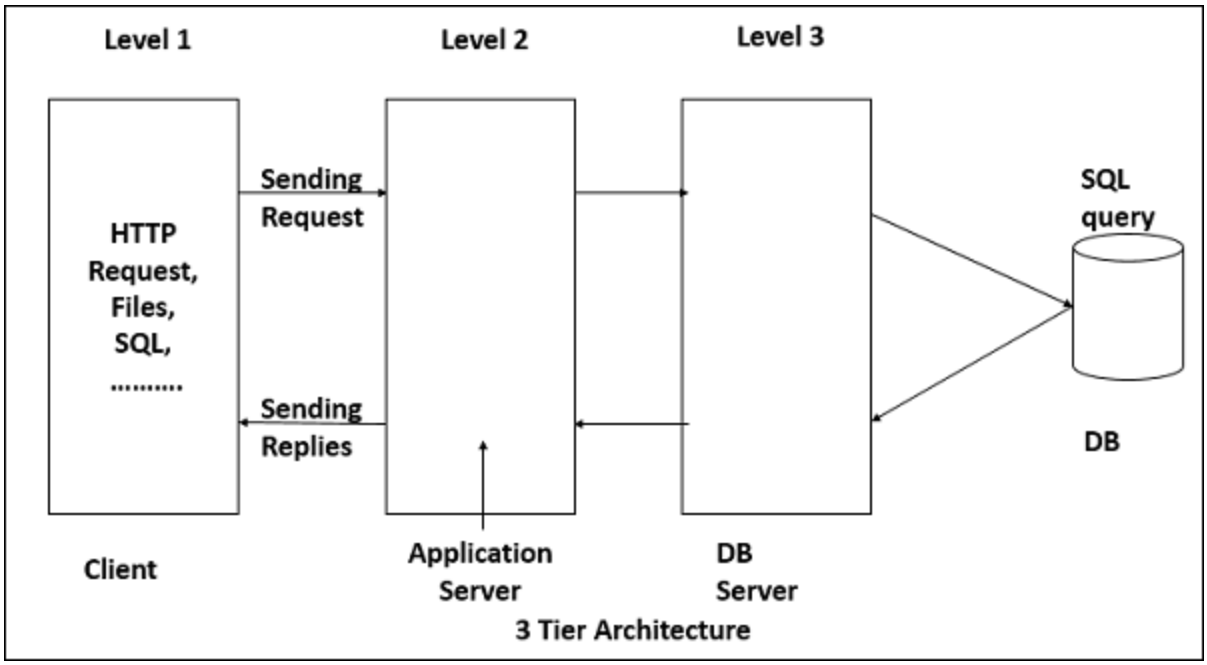
Presentation layer is the topmost level of the application by which users can access directly such as webpage or Operating System GUI (Graphical User interface). The primary function of this layer is to translate the tasks and results to something that user can understand. It communicates with other tiers so that it places the results to the browser/client tier and all other tiers in the network.

Application Tier (Business Logic, Logic Tier, or Middle Tier)

Application tier coordinates the application, processes the commands, makes logical decisions, evaluation, and performs calculations. It controls an application's functionality by performing detailed processing. It also moves and processes data between the two surrounding layers.

Data Tier

In this layer, information is stored and retrieved from the database or file system. The information is then passed back for processing and then back to the user. It includes the data persistence mechanisms (database servers, file shares, etc.) and provides API (Application Programming Interface) to the application tier which provides methods of managing the stored data.



Advantages

- Better performance than a thin-client approach and is simpler to manage than a thick-client approach.
- Enhances the reusability and scalability − as demands increase, extra servers can be added.
- Provides multi-threading support and also reduces network traffic.
- Provides maintainability and flexibility

Disadvantages

- Unsatisfactory Testability due to lack of testing tools.
- More critical server reliability and availability.

Broker Architectural Style

Broker Architectural Style is a middleware architecture used in distributed computing to coordinate and enable the communication between registered servers and clients. Here, object communication takes place through a middleware system called an object request broker (software bus).

- Client and the server do not interact with each other directly. Client and server have a direct connection to its proxy which communicates with the mediator-broker.
- A server provides services by registering and publishing their interfaces with the broker and clients can request the services from the broker statically or dynamically by look-up.

- CORBA (Common Object Request Broker Architecture) is a good implementation example of the broker architecture.

Components of Broker Architectural Style

The components of broker architectural style are discussed through following heads −

Broker

Broker is responsible for coordinating communication, such as forwarding and dispatching the results and exceptions. It can be either an invocation-oriented service, a document or message - oriented broker to which clients send a message.

- It is responsible for brokering the service requests, locating a proper server, transmitting requests, and sending responses back to clients.
- It retains the servers' registration information including their functionality and services as well as location information.
- It provides APIs for clients to request, servers to respond, registering or unregistering server components, transferring messages, and locating servers.

Stub

Stubs are generated at the static compilation time and then deployed to the client side which is used as a proxy for the client. Client-side proxy acts as a mediator between the client and the broker and provides additional transparency between them and the client; a remote object appears like a local one.

The proxy hides the IPC (inter-process communication) at protocol level and performs marshaling of parameter values and un-marshaling of results from the server.

Skeleton

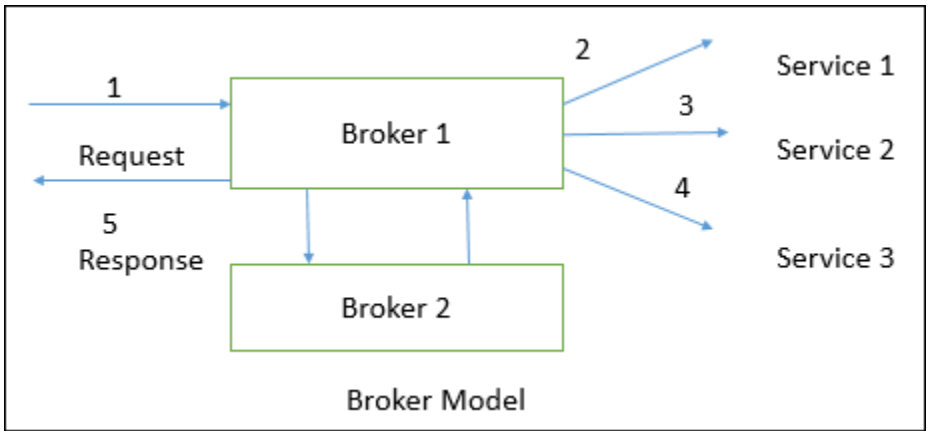
Skeleton is generated by the service interface compilation and then deployed to the server side, which is used as a proxy for the server. Server-side proxy encapsulates low-level system-specific networking functions and provides high-level APIs to mediate between the server and the broker.

It receives the requests, unpacks the requests, unmarshals the method arguments, calls the suitable service, and also marshals the result before sending it back to the client.

Bridge

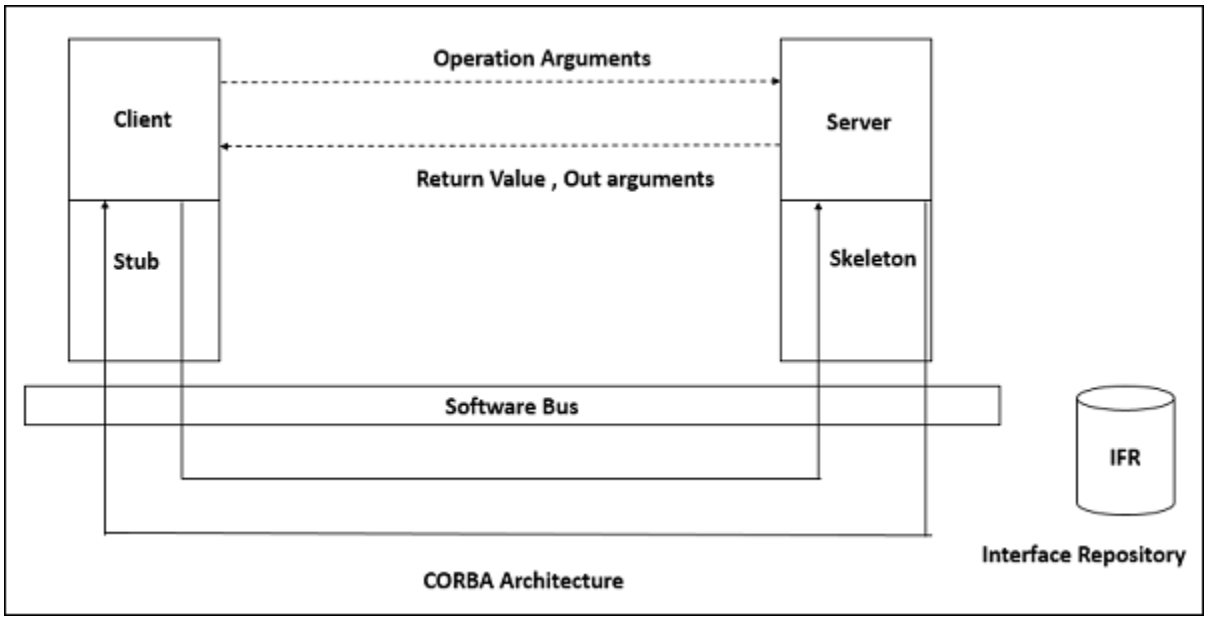
A bridge can connect two different networks based on different communication protocols. It mediates different brokers including DCOM, .NET remote, and Java CORBA brokers.

Bridges are optional component, which hides the implementation details when two brokers interoperate and take requests and parameters in one format and translate them to another format.



Broker implementation in CORBA

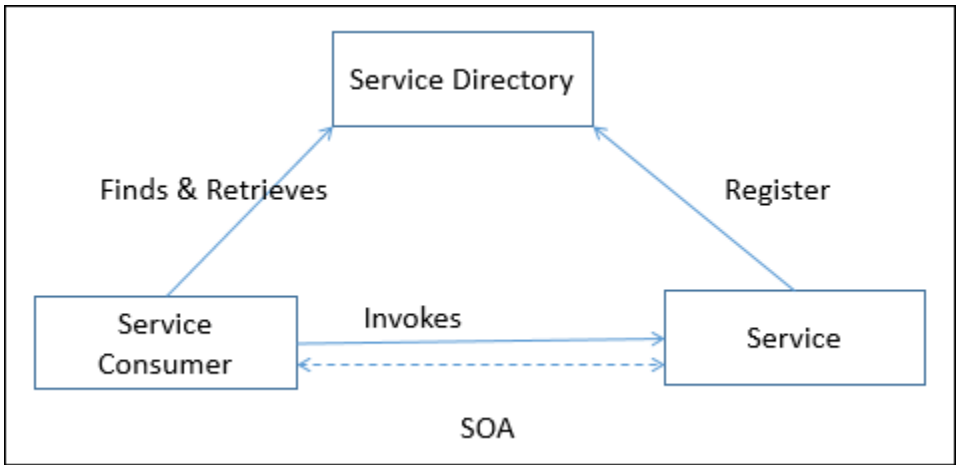
CORBA is an international standard for an Object Request Broker – a middleware to manage communications among distributed objects defined by OMG (object management group).



Service-Oriented Architecture (SOA)

A service is a component of business functionality that is well-defined, self-contained, independent, published, and available to be used via a standard programming interface. The connections between services are conducted by common and universal message-oriented protocols such as the SOAP Web service protocol, which can deliver requests and responses between services loosely.

Service-oriented architecture is a client/server design which support business-driven IT approach in which an application consists of software services and software service consumers (also known as clients or service requesters).



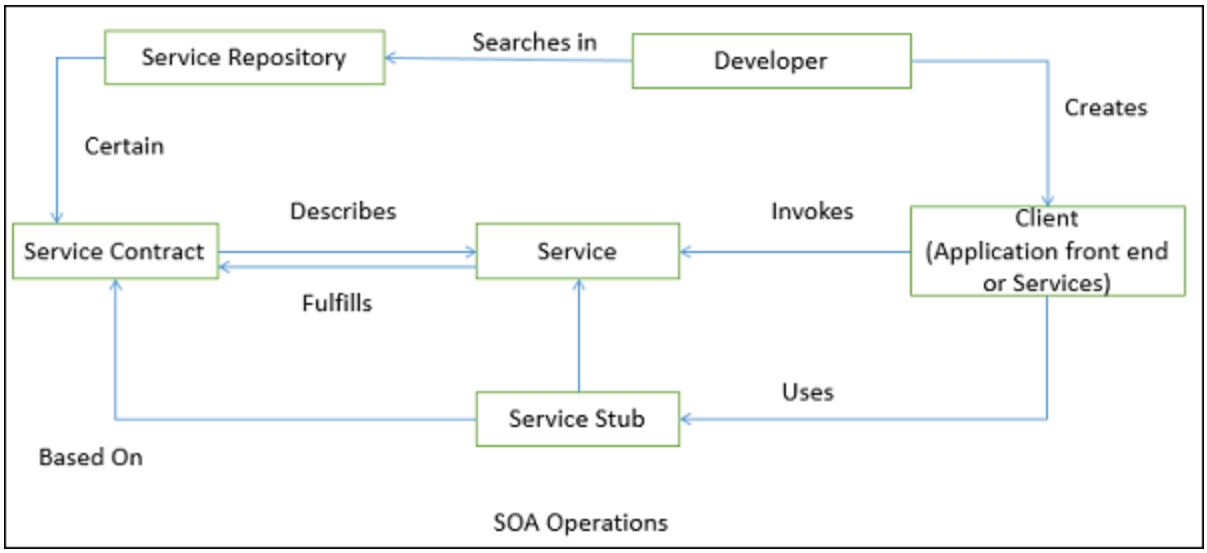
Features of SOA

A service-oriented architecture provides the following features −

- **Distributed Deployment** − Expose enterprise data and business logic as loosely, coupled, discoverable, structured, standard-based, coarse-grained, stateless units of functionality called services.
- **Composability** − Assemble new processes from existing services that are exposed at a desired granularity through well defined, published, and standard complaint interfaces.
- **Interoperability** − Share capabilities and reuse shared services across a network irrespective of underlying protocols or implementation technology.
- **Reusability** − Choose a service provider and access to existing resources exposed as services.

SOA Operation

The following figure illustrates how does SOA operate −



Advantages

- Loose coupling of service–orientation provides great flexibility for enterprises to make use of all available service recourses irrespective of platform and technology restrictions.
- Each service component is independent from other services due to the stateless service feature.
- The implementation of a service will not affect the application of the service as long as the exposed interface is not changed.
- A client or any service can access other services regardless of their platform, technology, vendors, or language implementations.
- Reusability of assets and services since clients of a service only need to know its public interfaces, service composition.
- SOA based business application development are much more efficient in terms of time and cost.
- Enhances the scalability and provide standard connection between systems.
- Efficient and effective usage of ‘Business Services’.
- Integration becomes much easier and improved intrinsic interoperability.
- Abstract complexity for developers and energize business processes closer to end users.

Service-Oriented Architecture(sos)

Service-Oriented Architecture (SOA) is an architectural approach in which applications make use of services available in the network. In this architecture, services are provided to form applications, through a communication call over the internet.

- SOA allows users to combine a large number of facilities from existing services to form applications.
- SOA encompasses a set of design principles that structure system development and provide means for integrating components into a coherent and decentralized system.
- SOA based computing packages functionalities into a set of interoperable services, which can be integrated into different software systems belonging to separate business domains.

There are two major roles within Service-oriented Architecture:

1. **Service provider:** The service provider is the maintainer of the service and the organization that makes available one or more services for others to use. To advertise services, the provider can publish them in a registry, together with a service contract that specifies the nature of the service, how to use it, the requirements for the service, and the fees charged.
2. **Service consumer:** The service consumer can locate the service metadata in the registry and develop the required client components to bind and use the service.

Services might aggregate information and data retrieved from other services or create workflows of services to satisfy the request of a given service consumer. This practice is known as service orchestration Another important interaction pattern is service choreography, which is the coordinated interaction of services without a single point of control.

Components of SOA:

Guiding Principles of SOA:

1. **Standardized service contract:** Specified through one or more service description documents.
2. **Loose coupling:** Services are designed as self-contained components, maintain relationships that minimize dependencies on other services.
3. **Abstraction:** A service is completely defined by service contracts and description documents. They hide their logic, which is encapsulated within their implementation.

4. **Reusability:** Designed as components, services can be reused more effectively, thus reducing development time and the associated costs.
5. **Autonomy:** Services have control over the logic they encapsulate and, from a service consumer point of view, there is no need to know about their implementation.
6. **Discoverability:** Services are defined by description documents that constitute supplemental metadata through which they can be effectively discovered. Service discovery provides an effective means for utilizing third-party resources.
7. **Composability:** Using services as building blocks, sophisticated and complex operations can be implemented. Service orchestration and choreography provide a solid support for composing services and achieving business goals.

Advantages of SOA:

- **Service reusability:** In SOA, applications are made from existing services.Thus, services can be reused to make many applications.
- **Easy maintenance:** As services are independent of each other they can be updated and modified easily without affecting other services.
- **Platform independant:** SOA allows making a complex application by combining services picked from different sources, independent of the platform.
- **Availability:** SOA facilities are easily available to anyone on request.
- **Reliability:** SOA applications are more reliable because it is easy to debug small services rather than huge codes
- **Scalability:** Services can run on different servers within an environment, this increases scalability

Disadvantages of SOA:

- **High overhead:** A validation of input parameters of services is done whenever services interact this decreases performance as it increases load and response time.
- **High investment:** A huge initial investment is required for SOA.
- **Complex service management:** When services interact they exchange messages to tasks. the number of messages may go in millions. It becomes a cumbersome task to handle a large number of messages.

Practical applications of SOA: SOA is used in many ways around us whether it is mentioned or not.

1. SOA infrastructure is used by many armies and air force to deploy situational awareness systems.
2. SOA is used to improve the healthcare delivery.
3. Nowadays many apps are games and they use inbuilt functions to run. For example, an app might need GPS so it uses inbuilt GPS functions of the device. This is SOA in mobile solutions.

4. SOA helps maintain museums a virtualized storage pool for their information and content.

WEB SERVICES

Different books and different organizations provide different definitions to Web Services. Some of them are listed here.

- A web service is any piece of software that makes itself available over the internet and uses a standardized XML messaging system. XML is used to encode all communications to a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. As all communication is in XML, web services are not tied to any one operating system or programming language—Java can talk with Perl; Windows applications can talk with Unix applications.
- Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML.
- Web services are XML-based information exchange systems that use the Internet for direct application-to-application interaction. These systems can include programs, objects, messages, or documents.
- A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

To summarize, a complete web service is, therefore, any service that −

- Is available over the Internet or private (intranet) networks
- Uses a standardized XML messaging system
- Is not tied to any one operating system or programming language
- Is self-describing via a common XML grammar
- Is discoverable via a simple find mechanism

Components of Web Services

The basic web services platform is XML + HTTP. All the standard web services work using the following components −

- SOAP (Simple Object Access Protocol)
- UDDI (Universal Description, Discovery and Integration)
- WSDL (Web Services Description Language)

All these components have been discussed in the Web Services Architecturechapter.

Web Service Work

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of −

- XML to tag the data
- SOAP to transfer a message
- WSDL to describe the availability of service.

You can build a Java-based web service on Solaris that is accessible from your Visual Basic program that runs on Windows.

You can also use C# to build new web services on Windows that can be invoked from your web application that is based on JavaServer Pages (JSP) and runs on Linux.

Example

Consider a simple account-management and order processing system. The accounting personnel use a client application built with Visual Basic or JSP to create new accounts and enter new customer orders.

The processing logic for this system is written in Java and resides on a Solaris machine, which also interacts with a database to store information.

The steps to perform this operation are as follows −

- The client program bundles the account registration information into a SOAP message.
- This SOAP message is sent to the web service as the body of an HTTP POST request.
- The web service unpacks the SOAP request and converts it into a command that the application can understand.
- The application processes the information as required and responds with a new unique account number for that customer.
- Next, the web service packages the response into another SOAP message, which it sends back to the client program in response to its HTTP request.
- The client program unpacks the SOAP message to obtain the results of the account registration process.

GRID COMPUTING

Grid computing is a distributed structure of a large number of computers connected to solve a complicated problem. In grid computing, servers and computers run independently and are loosely connected by the Internet. Computers may connect directly or through scheduling systems.
In other words, Grid Computing involves a large number of computer which are connected parallel and makes a computer cluster.
Grid computing is used in various types of applications such as mathematical, scientific, and educational tasks via various computing resources.
Grid computing is a processor architecture that integrates computer resources from various domains to achieve a primary goal. The computers on the network will work together in grid computing on a project, thus acting as a supercomputer.
Grid systems are mainly designed for resource sharing by Distributed
and cluster computing on a large scale. It divides the complex tasks into smaller pieces that are distributed to the CPUs

Cloud Computing

loud Computing is defined as the on-demand facility of computer power, database storage, applications, and other IT resources through the internet. ***It provides a solution for IT infrastructure at a low price.***
In simple words, cloud computing means storing and accessing the data via the internet instead of the computer's hard drive.
Cloud computing is a pay-per-use model.

Grid Computing Vs Cloud Computing

1. **Main objective: –** The main objective of cloud computing is to offer the Service at a lower rate. It also offers scalability and flexibility so that the customer efficiently uses cloud computing with increased security and availability.
However, the grid computing objective is to focus on the network to solve complicated problems; it also provides a computer as a utility.
2. Types and Division: – The types of cloud computing are public clouds, private clouds, community clouds, and hybrid clouds.
However, Grid computing is a distributed computing system so its types are distributed information system and distributed pervasive systems.

- **Use and Security: –** A Large amount of data is stored in the cloud. So it offers security according to it. The data which is stored on the cloud is secured and only be access with the help of credentials.
Grid computing relates to idol energy in computers and is mainly used for something sensible.
- **Basis of Dependency: –** Cloud Computing is totally dependent on the internal. The cloud offers high security along with high performance.
Grid computing can do its work continuously, even if a computer stops or failure. The other computer will pick the working and make the system more efficient and reliable.

- **Difference and Similarity: –** cloud computing and grid computing are different from each other in some terms like architecture, business model, and interoperability.
The similarity between cloud computing and grid computing is both are network-based technologies
.**6. Space and Storage: –** In cloud computing, backup and restores the data is easy due to of its fast data processors. The updation in cloud computing are automatic and efficient. However, in grid computing, space is saved, and access to additional resources can be done.
7.Remote Usage: – In cloud computing, management of computing resources are within a single location, which is located at a different place.
However, in grid computing, there is a distributed system where the resources are allocated at various locations, and can be located from various sites.
8. Resource Requirements: – Grid computing requires more resources, and cloud computing doesn't access the resource directly; it gets resources via the internet.
9. Problem-Solving Techniques: – For job scheduling, grid computing uses all kinds of computing resources. In grid computing, the big task is split into multiple tasks, which is solved by various computers as the work assigns to a particular computer.
Cloud computing has resources that are pooling via grouping resources and needs a base from a cluster of servers.
10. Services and Capabilities: – Academic researchers mainly used Grid computing and have the ability to handle a large set of job that are complex and includes a large collection of data.
Cloud computing is totally internet-based computing. The cloud offers various types of services like management of data, job queries, the security of data, etc. It removes the cost of purchasing new hardware and software which are required to build applications.
11. Interoperability: – In grid computing, interoperability can be handled quickly, but cloud computing does not support interoperability.

Grid Computing Vs. Cloud Computing: Comparison Chart

GRID COMPUTING	CLOUD COMPUTING
Grid computing is for Application-oriented.	Cloud computing is for Service-oriented.
In Grid computing, resources are shared among multiple computing units for processing a single task.	In cloud computing, all the resources are managed centrally and are place over different servers in clusters.
Grid computing is a collection of Interconnected computers and networks that can be called for large scale processing tools.	In cloud computing, more than one computer coordinates to resolve the problem together.
Grid computing is operated within a corporate network	Cloud computing can be accessed via the Internet.

In this, Grids are mainly owned and managed by an organization within its premises.	The cloud servers are owned by infrastructure providers and are placed in physically various locations.
It offers a shared pool of computing resources based on needs.	Cloud computing includes dealing with a common problem using a varying number of computing resources.

PROS &CONS

Advantages and Disadvantages of Cloud Computing

Advantages of Cloud Computing

As we all know that Cloud computing is trending technology. Almost every company switched their servi
cloud to rise the company growth.

Here, we are going to discuss some important advantages of Cloud Computing-

1) Back-up and restore data

Once the data is stored in the cloud, it is easier to get back-up and restore that data using the cloud.

2) Improved collaboration

Cloud applications improve collaboration by allowing groups of people to quickly and easily share inform
cloud via shared storage.

3) Excellent accessibility

Cloud allows us to quickly and easily access store information anywhere, anytime in the whole world, us
internet connection. An internet cloud infrastructure increases organization productivity and efficiency b
that our data is always accessible.

4) Low maintenance cost

Cloud computing reduces both hardware and software maintenance costs for organizations.

5) Mobility

Cloud computing allows us to easily access all cloud data via mobile.

6) IServices in the pay-per-use model

Cloud computing offers Application Programming Interfaces (APIs) to the users for access services on th
pays the charges as per the usage of service.

7) Unlimited storage capacity

Cloud offers us a huge amount of storing capacity for storing our important data such as documents, im
video, etc. in one place.

8) Data security

Data security is one of the biggest advantages of cloud computing. Cloud offers many advanced feature
security and ensures that data is securely stored and handled.

Disadvantages of Cloud Computing

A list of the disadvantage of cloud computing is given below -

1) Internet Connectivity

As you know, in cloud computing, every data (image, audio, video, etc.) is stored on the cloud, and we
data through the cloud by using the internet connection. If you do not have good internet connectivity,
access these data. However, we have no any other way to access data from the cloud.

2) Vendor lock-in

Vendor lock-in is the biggest disadvantage of cloud computing. Organizations may face problems when t
their services from one vendor to another. As different vendors provide different platforms, that can cau
moving from one cloud to another.

3) Limited Control

As we know, cloud infrastructure is completely owned, managed, and monitored by the service provider
users have less control over the function and execution of services within a cloud infrastructure.

4) Security

Although cloud service providers implement the best security standards to store important information.
adopting cloud technology, you should be aware that you will be sending all your organization's sensitive
to a third party, i.e., a cloud computing service provider. While sending the data on the cloud, there ma
that your organization's information is hacked by Hackers.

REAL TIME APPLICATIONS

Cloud Computing has its applications in almost all the fields such as business, entertainment, data st networking, management, entertainment, education, art and **global positioning system,** etc. Some famous cloud computing applications are discussed here in this tutorial:

Business Applications

Cloud computing has made businesses more collaborative and easy by incorporating various as **MailChimp, Chatter, Google Apps for business,** and **Quickbooks.**

SN	Application Description
1	**MailChimp** It offers an **e-mail publishing platform.** It is widely employed by the businesses to design and send their e-mail campaigns.
2	**Chatter** **Chatter app** helps the employee to share important information about organization in real time. One can get the instant feed regarding any issue.
3	**Google Apps for Business** **Google** offers **creating text documents, spreadsheets, presentations,**etc., on **Google Docs** which allows the business users to share them in collaborating manner.
4	**Quickbooks** It offers **online accounting solutions** for a business. It helps in **monitoring cash flow, creating VAT returns** and **creating business reports.**

Data Storage and Backup

Box.com, Mozy, Joukuu are the applications offering data storage and backup services in cloud.

SN	Application Description

1	**Box.com** **Box.com** offers drag and drop service for files. The users need to drop the files into Box and access from anywhere.
2	**Mozy** **Mozy** offers online backup service for files to prevent data loss.
3	**Joukuu** **Joukuu** is a web-based interface. It allows to display a single list of contents for files stored in **Google Docs, Box.net and Dropbox.**

Management Applications

There are apps available for management task such as **time tracking, organizing notes.** Application such tasks are discussed below:

SN	Application Description
1	**Toggl** It helps in tracking time period assigned to a particular project.
2	**Evernote** It organizes the sticky notes and even can read the text from images which helps the user to locate the notes easily.
3	**Outright** It is an accounting app. It helps to track income, expenses, profits and losses in real time.

Social Applications

There are several social networking services providing websites such as Facebook, Twitter, etc.

SN	Application Description
1	**Facebook** It **offers** social networking service. One can share photos, videos, files, status and much more.
2	**Twitter** It **helps** to interact with the public directly. One can follow any celebrity, organization and any person, who is on twitter and can have latest updates regarding the same.

Entertainment Applications

SN	Application Description
1	**Audio box.fm** It offers streaming service. The music files are stored online and can be played from cloud using the own media player of the service.

Art Applications

SN	Application Description
1	**Moo** It offers art services such as designing and printing **business cards, postcards** and **mini cards.**