Types of distributed systems:

High performance distributed computing:

- Started with the introduction of **multiprocessor machines**. It was not scalable.
- To overcome the limitations of shared-memory systems, high-performance computing moved to **distributed shared memory multi computers.** Allows a processor to address a memory location at another computer as if it were local memory. It had performance issue too.

Cluster Computing:

Started with **Beowulf clusters:**

- collection of reticulated standalone computers working together as a single integrated computing resource.
- It has no custom hardware or software but consists of standard, off-the-shelf computers
- One server node controlling a set of "client nodes" connected by Ethernet and functioning as a single machine on the Linux operating system.
- For optimal performance, all nodes except the master node should have same hardware specifications
- To provide cost-effective computing power for scientific applications

Followed by **MOSIX systems**:

- MOSIX attempts to provide a **single-system image** of a cluster, meaning that to a process a cluster computer offers the ultimate distribution transparency by appearing to be a single computer.
- MOSIX does load balancing automatically just 'fork and forget'. It allows processes to dynamically and pre-emptively migrate between the nodes that make up the cluster.
- Supports different cluster types, including clusters with different CPU speeds.
- Consider a heavily-used University server that has many simultaneous login sessions and an extremely high load. MOSIX is ideal for these types of situations, since the high load and large number of processes means that MOSIX will have many opportunities to distribute the load among nodes in the cluster.

Followed by **Hybrid Solutions**:

- nodes are specifically configured for certain tasks.
- Having compute nodes with dedicated, lightweight operating systems will most likely provide optimal performance for compute-intensive applications
- storage functionality can most likely be optimally handled by other specially configured nodes such as file and directory servers.

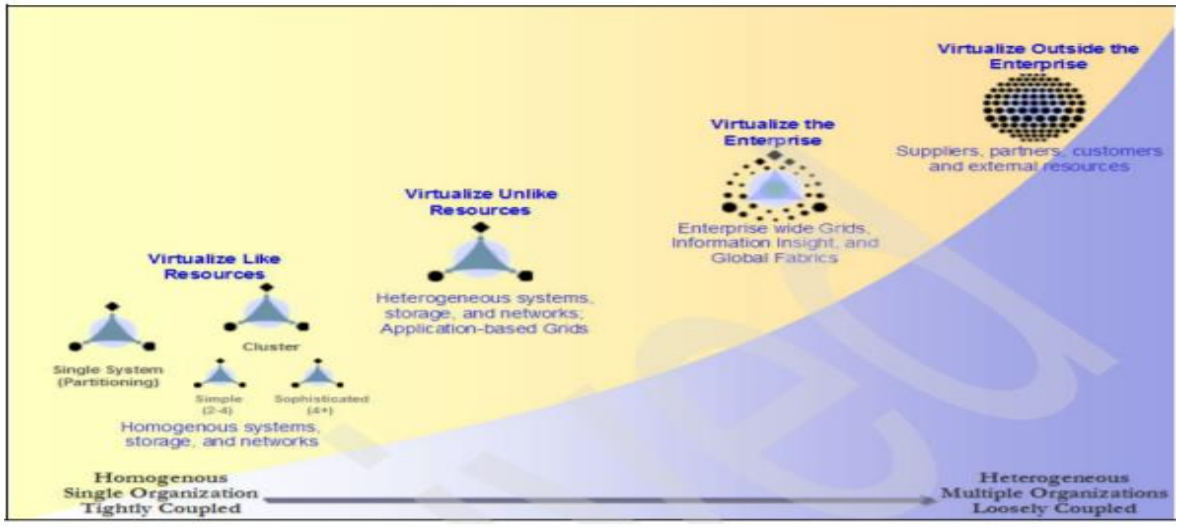
Grid Computing:

The grand vision is often presented as an analogy to power grids where users (or electrical appliances) get access to electricity through wall sockets with no care or consideration for where or how the electricity is actually generated.

Resources from different organizations are brought together to allow the collaboration of a group of people from different institutions, indeed forming a federation of systems

Such a collaboration is realized in the form of a **virtual organization**.

could be defined as any of a variety of levels of virtualization along a continuum.



Architecture of GRID:

Fabric Layer:

- All shareable resources are placed in this layer; such as processors, memories, sensors and actuators.
- It will provide functions for querying the state and capabilities of a resource.

Connectivity layer:

- protocols are placed which are related to core communication and authentication.
- The GSI (Grid Security Infrastructure) [6] protocol underlies every Grid transaction

Resource Layer:

- All common actions related to network parts are guided in this layer; like negotiation, initiation, monitoring, control, accounting and payment of sharing operations on individual resources.
- The GRAM (Grid Resource Access and Management) protocol is used for allocation of computational resources and for monitoring and control of computation on those resources, and GridFTP for data access and high-speed data transfer.

Collective Layer:

- Any Collaborative operations in the shareable resources are placed in this layer.
- Condor-G and Nimrod-G are examples of co-allocating, scheduling and brokering services,

Application Layer:

- It comprises whatever user applications built on top of the above protocols and APIs and operate in VO environments.

Open Grid Services Architecture(**OGSA**):

- Alternative architecture in GRID computing
- addresses the requirements of grid computing in an open and standard way,
- It extends Web Service and SOA to the grid computing environment.

Cloud Computing:

- A new concept developed called utility computing by which a customer could upload tasks to a data centre and be charged on a per-resource basis. It forms the basis of cloud computing.
- pay-per-use model in which guarantees are offered by means of customized service-level agreements.
- It has more business aspect rather than technological aspect to it.
- A serious alternative to maintaining huge local infrastructures.
- One can expect to have monetary savings, because fewer machines or network connections need to be maintained.

B_c : benefits of migrating a compute-intensive component
M_c : total number of migrated compute-intensive components
B_s : benefits of migrating a storage-intensive component
M_s : total number of migrated storage-intensive components

$$B_c \cdot M_c + B_s \cdot M_s$$

Layers of Cloud Computing:

I. **Hardware**: processors, routers, but also power and cooling systems

II. **Infrastructure:** virtual storage and computing resources,Hypervisors.

III. **Platform**: provides computing platform which primarily includes resources like operating system, programming language, database, web server

IV. **Application:** Web Services, Multimedia and Business apps

Services provided to Customers:

Infrastructure-as-a-Service (**IaaS**):

Minimal requirements to build IaaS cloud employs: Hypervisor – VMM (Virtual Machine Monitor), and Networking topology – public or private. IaaS mitigates the need for a data center, and maintaining hardware at the local level. Sometimes IaaS is otherwise regarded as Hardware as a Service (HaaS). Examples: Datacentres, Amazon S3

Platform-as-a-Service (**PaaS**):

PaaS provides computing platform which primarily includes resources like operating system, programming language, database, web server that automatically scales to meet the application demands. MS Azure, Google App engine

Software-as-a-Service (**SaaS**) :

Applications services are provided by using multitenant architecture.

The advantage is that it requires zero installation of the software and hardware structures and is capable of being accessible from anywhere with an internet connection

Google Docs, Gmail

Distributed information systems:

Distributed transaction processing:

I. operations on a database are carried out in the form of **transactions**.
II. Transactions follow ACID properties:

Atomicity:

- Either the entire transaction takes place at once or doesn't happen at all.
- All or nothing rule.

Consistency:

- Data is in a consistent state when a transaction starts and when it ends.
- For example, in an application that transfers funds from one account to another, the consistency property ensures that the total value of funds in both the accounts is the same at the start and end of each transaction.
- For example, if we try to add a record in a sale table with the code of a product that does not exist in the product table, the transaction will fail.

Isolation:

- Transaction will not be changed by any other concurrent transaction

Durability:

- Once a transaction completes successfully, the changes it has made into the database should be permanent even if there is a system failure.

III. transactions are often constructed as a number of sub transactions, jointly forming a **nested transaction.**
IV. TP monitor is used to handle nested transactions**.**

Enterprise application integration:

I. Merges many applications running in an enterprise through an intermediary interface(middleware) which can combine all applications, their databases, and processes running internally or externally
II. Focusses on Interoperability and Coordination.

III. Many different communication models exist as middleware for inter application communication:

- RPC to send a request to another application component by doing a local procedure call.(tightly coupled)
- RMI calls to invoke objects running on different JVM.(tightly coupled)
- **MOM** or **publish-subscribe** systems send messages to logical contact points.

Four ways to integrate applications:

File transfer: text binary, XML

Shared database: through SQL triggers cursers

Remote procedure call : for series of actions to be performed

Messaging: to curb the drawback of RPC

Pervasive systems:

- ✓ Small devices invisibly embedded in the everyday human surroundings and therefore provide an easy and omnipresent access to a computing environment.
- ✓ no single dedicated interface, such as a screen/keyboard combination.
- ✓ equipped with many **sensors** that pick up various aspects of a user's behavior
- ✓ myriad of **actuators** to provide information
- ✓ The fundamental properties of a system comprised of tabs, pads and boards include wireless communications, embedded and mobile devices, distributed computing, and context awareness.

Ubiquitous computing systems:

the system is pervasive and continuously present. The latter means that a user will be continuously interacting with the system, often not even being aware that interaction is taking place.

Properties:

- ✓ Devices are networked, distributed, and accessible in transparent manner forming a single coherent system.
- ✓ Interaction between users and devices are happening without any attention in a seamless way integrated into the user's tasks.
- ✓ Systems can adjust their properties and behavior according to the information about the current state of the user.(Context Awareness)
- ✓ Devices operate autonomously without human intervention, and are thus highly self-managed.(Autonomy)
- ✓ The system as a whole can handle a wide range of dynamic actions and interactions(Intelligence)

Mobile computing systems:

- ✓ devices include remote controls, pagers, active badges, car equipment, various GPS-enabled devices, and so on.
- ✓ use wireless communication.
- ✓ the location of a device is assumed to change over time.

Sensor networks:

- ✓ tens to hundreds or thousands of relatively small nodes, each equipped with one or more sensing devices.
- ✓ nodes can often act as actuator
- ✓ Perfect Example: automatic activation of sprinklers when a fire has been detected.