VIT®
BHOPAL
www.vitbhopal.ac.in

CSE3009 - Parallel and Distributed Computing

Course Type: LTP Credits: 4

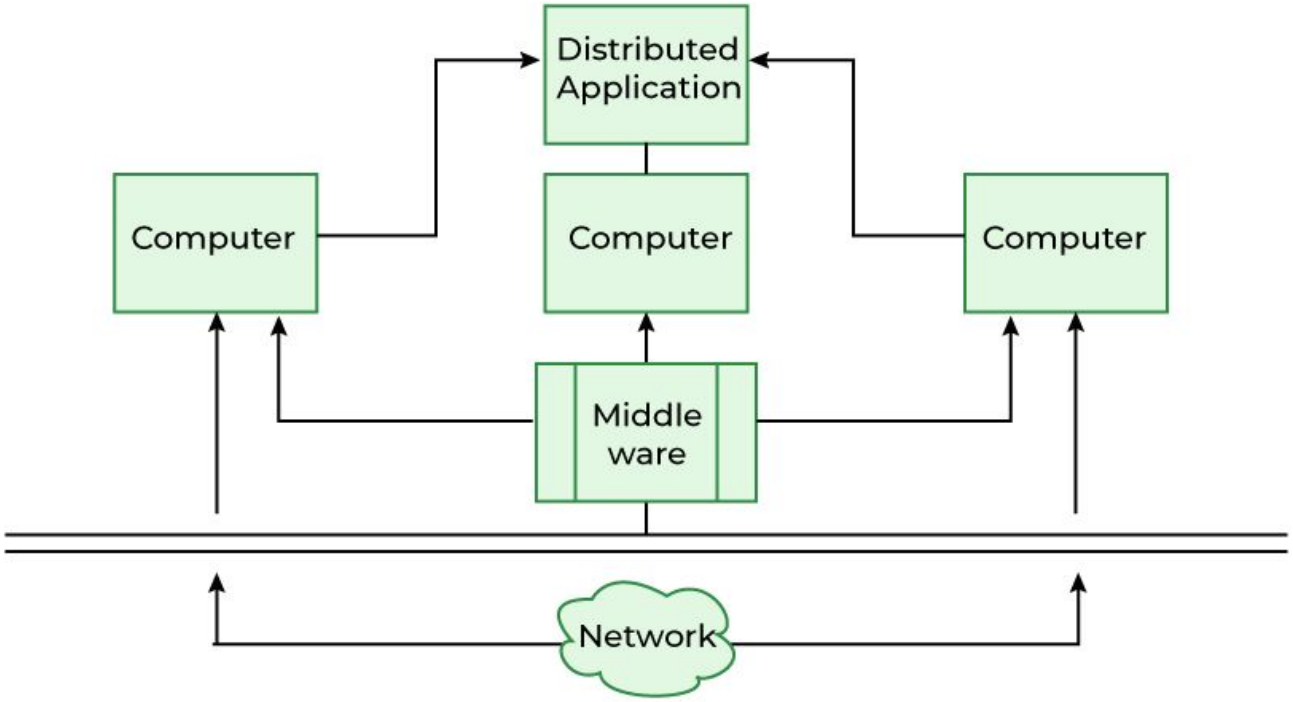
Prepared by
Dr Komarasamy G
Associate Professor (Grade-2)
School of Computing Science and Engineering
VIT Bhopal University

Unit-3

- **Introduction to Distributed Systems – Definition, Issues, Goals, Types of distributed systems, Distributed System Models, Hardware concepts, Software Concept, Design Issues.**
- Communication – Layered Protocols, Remote Procedure Call, Remote Object Invocation, Message Oriented Communication, Stream Oriented Communication – Case Study (RPC and Java RMI).
- Parallel Random Access Machine (PRAM) model, PRAM architecture.

Introduction to Distributed Systems

- Distributed System is a Network of **Machines that can exchange information with each other through Message-passing.**
- It can be very useful as it helps in **resource sharing.** It enables computers to coordinate their activities and to share the resources of the system so that users perceive the system as a single, integrated computing facility.



Types of Distributed Systems

1. **Client/Server Systems**
2. **Peer-to-Peer Systems**
3. **Middleware**
4. **Three-tier**
5. **N-tier**

1. Client/Server Systems:

- Client-Server System is the most basic communication method where the **client sends input to the server and the server replies to the client with an output.**
- The client requests the server for resources or a task to do, the server allocates the resource or performs the task and sends the result in the form of a response to the request of the client. Client Server System can be applied with multiple servers.

Types of Distributed Systems

2. **Peer-to-Peer Systems:** Peer-to-Peer System communication model works as a decentralized model in which the system works like **both Client and Server.** Nodes are an important part of a system.

- In this, each node performs its task on its **local memory and shares data through the supporting medium**, this node can work as a server or as a client for a system.
- Programs in the peer-to-peer system can communicate at the same level without any hierarchy.

3. **Middleware:** Middleware can be thought as an application that sits between two separate applications and provides service to both. It works as a base for different interoperability applications running on different operating systems. Data can be transferred to other between others by using this service.

Types of Distributed Systems

4. **Three-tier:** Three-tier system uses a separate layer and server for each function of a program.

- In this data of the **client is stored in the middle tier rather than sorted into the client system** or on their server through which development can be done easily.
- It includes an Application Layer, Data Layer, and Presentation Layer. This is mostly used in **web or online applications.**

5. **N-tier:** N-tier is also called a **multitier distributed system.** The N-tier system can contain any number of functions in the network.

- N-tier systems contain similar structures to three-tier architecture. When interoperability sends the request to another application to perform a task or to provide a service. **N-tier is commonly used in web applications and data systems.**

Types of Distributed Systems

Characteristics of Distributed System

- **Resource Sharing:** It is the ability to use any Hardware, Software, or Data anywhere in the System.
- **Concurrency:** It is naturally present in Distributed Systems, that deal with the same activity or functionality that can be performed by separate users who are in remote locations. Every local system has its independent Operating Systems and Resources.
- **Scalability:** It increases the scale of the system as several processors communicate with more users by accommodating to improve the responsiveness of the system.
- **Transparency:** It hides the complexity of the Distributed Systems from the Users and Application programs as there should be privacy in every system.

Types of Distributed Systems

Challenges of Distributed Systems

- **Network latency:** The communication network in a distributed system can introduce latency, which can affect the performance of the system.
- **Distributed coordination:** Distributed systems require coordination among the nodes, which can be challenging because of the distributed nature of the system.
- **Data consistency:** Maintaining data consistency across multiple nodes in a distributed system can be challenging.

Distributed System Models

- Distributed computing refers to a system where processing and data storage is distributed across multiple devices or systems, rather than being handled by a single central device.

Types of Distributed Computing System Models

- **Physical Model**
- A physical model is basically a **representation of the underlying hardware elements of a distributed system**. It encompasses the **hardware composition** of a distributed system in terms of computers and other devices and their interconnections. It is primarily used to design, manage, implement and determine the performance of a distributed system. A physical model majorly consists of the following components:

Distributed System Models

- **Nodes** – Nodes are the **end devices that have the ability of processing data**, executing tasks and communicating with the other nodes.
- These end devices are generally the computers at the user end or can be servers, workstations etc.
- Nodes provision the distributed system with an interface in the presentation layer that enables the user to interact with other back-end devices, or nodes, that can be used for storage and database services, or processing, web browsing etc.
- **Each node has an Operating System, execution environment and different middleware requirements that facilitate communication and other vital tasks.**

Distributed System Models

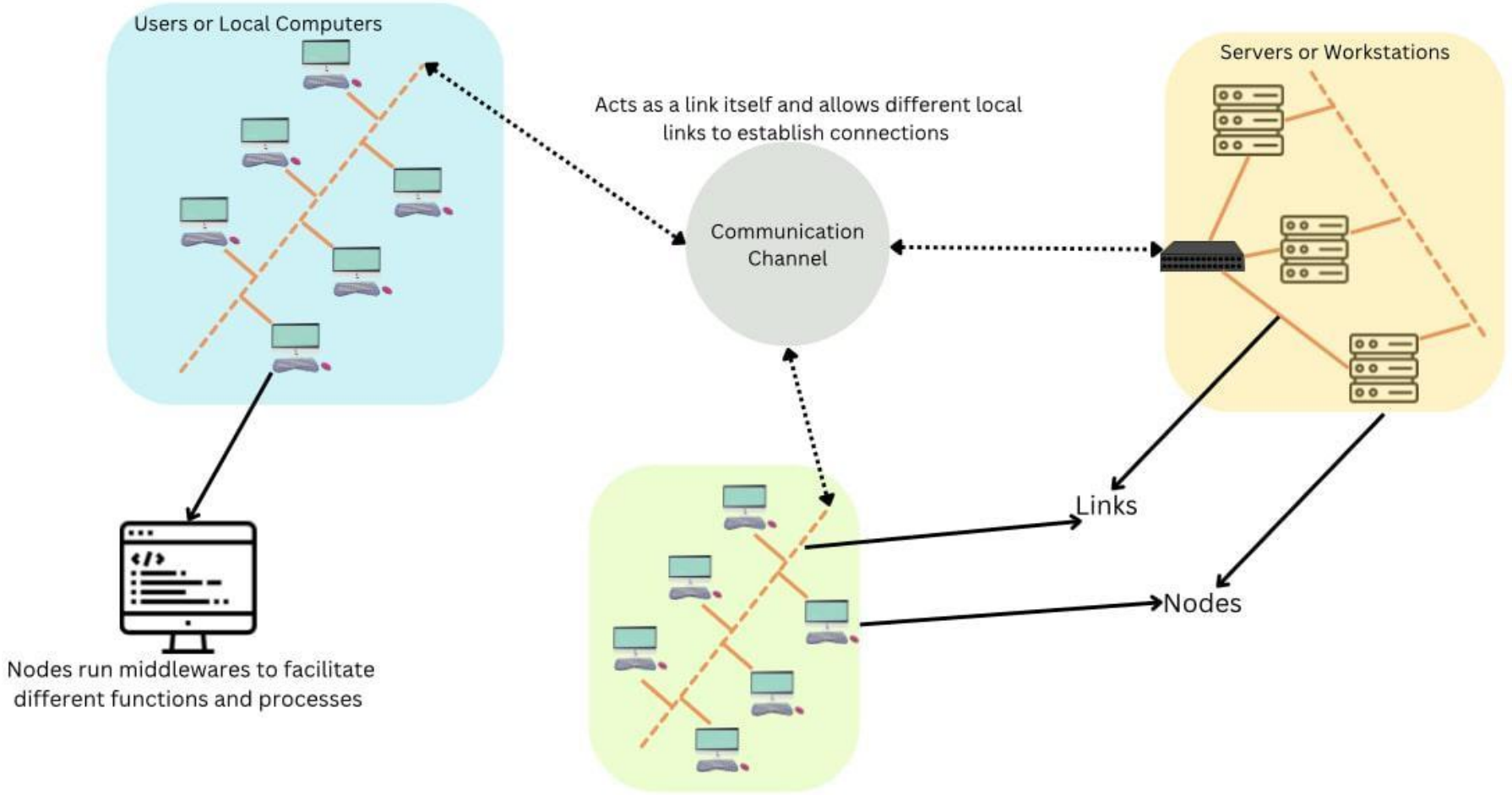
- **Links** – Links are the **communication channels between different nodes and intermediate devices.** **These may be wired or wireless.**
- Wired links or physical media are implemented using copper wires, fibre optic cables etc. The choice of the medium depends on the environmental conditions and the requirements.
- Generally, physical links are required for high performance and real-time computing. **Different connection types that can be implemented are as follows:**
  - **Point-to-point links –** It establishes a connection and allows data transfer between only two nodes.
  - **Broadcast links –** It enables a single node to transmit data to multiple nodes simultaneously.
  - **Multi-Access links –** Multiple nodes share the same communication channel to transfer data. Requires protocols to avoid interference while transmission.

Distributed System Models

- **Middleware –** These are the software's installed and executed on the nodes. By running middleware on each node, the **distributed computing system achieves a decentralized control and decision-making.**
- It handles various tasks like communication with other nodes, resource management, fault tolerance, synchronization of different nodes and security to prevent malicious and unauthorized access.
- **Network Topology –** This defines the arrangement of nodes and links in the distributed computing system.
- The most common network topologies that are implemented are **bus, star, mesh, ring or hybrid.**
- Choice of topology is done by determining the exact use cases and the requirements.

Distributed System Models

- **Communication Protocols** – Communication protocols are the set rules and procedures for transmitting data from in the links. Examples of these protocols include **TCP, UDP, HTTPS, MQTT** etc. These allow the nodes to communicate and interpret the data.

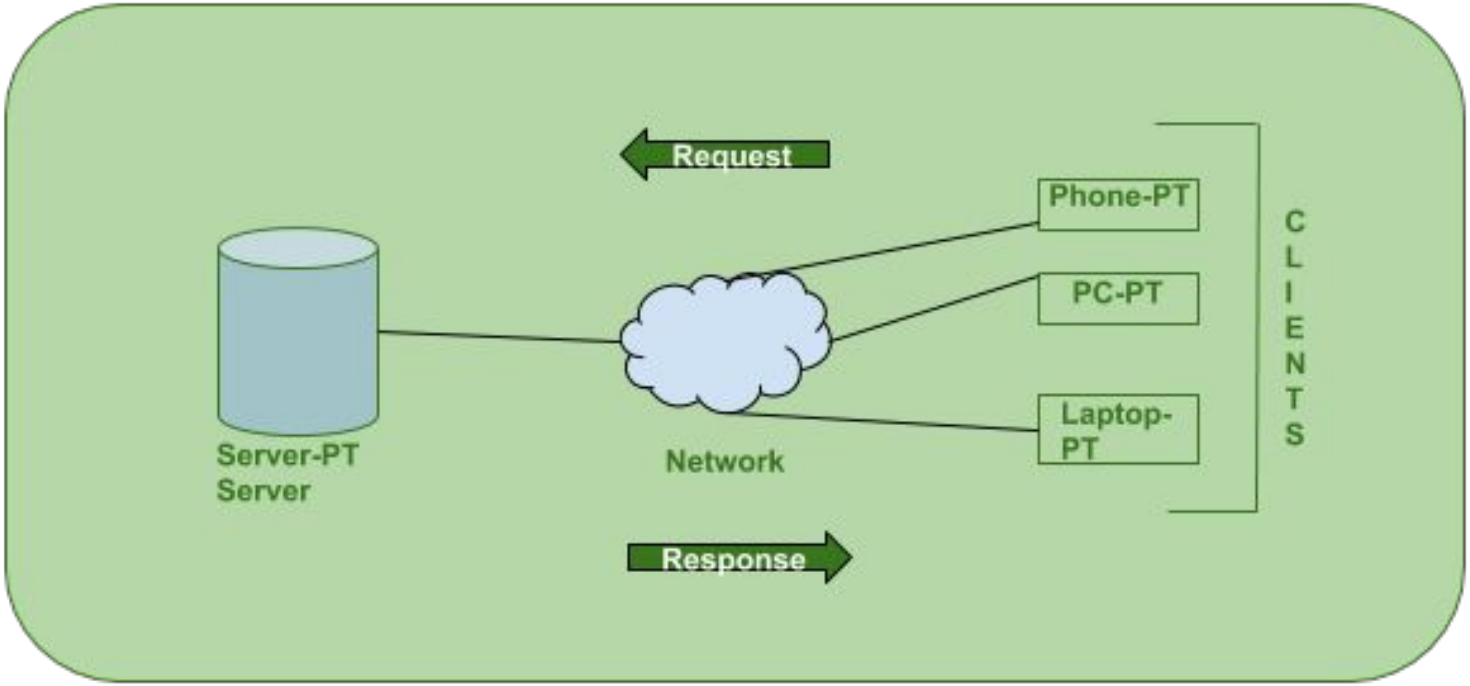


Distributed System Models

- **Architectural Model -** Architectural model in distributed computing system is the **overall design and structure of the system**, and how its different components are organized to interact with each other and provide the desired functionalities.
- It is an overview of the system, on how will the development, deployment and operations take place. Construction of a good architectural model is required for efficient cost usage, and highly improved scalability of the applications. **The key aspects of architectural model are –**
- **Client-Server model –** It is a centralized approach in which the **clients initiate requests for services and severs respond by providing those services.** It mainly works on the request-response model where the client sends a request to the server and the server processes it, and responds to the client accordingly. It can be achieved by using TCP/IP, HTTP protocols on the transport layer. This is mainly used in web services, cloud computing, database management systems etc.

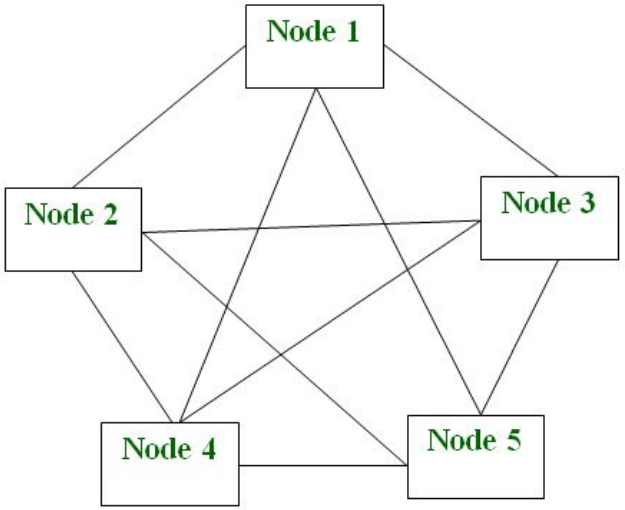
Distributed System Models

- **Client-Server model**



Distributed System Models

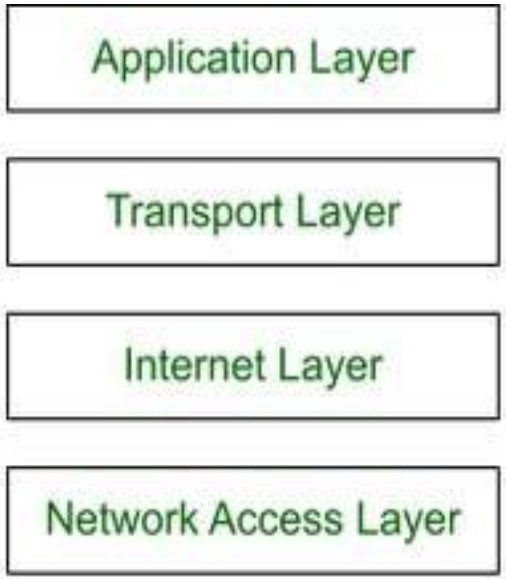
- **Peer-to-peer model** – It is a **decentralized approach in which all the distributed computing nodes**, known as peers, are all the same in terms of computing capabilities and can both request as well as provide services to other peers. It is a highly scalable model because the peers can join and leave the system dynamically, which makes it an ad-hoc form of network. The resources are distributed and the peers need to look out for the required resources as and when required. The communication is directly done amongst the peers without any intermediaries according to some set rules and procedures defined in the P2P networks. **The best example of this type of computing is BitTorrent.**



P2P Architecture

Distributed System Models

- **Layered model** – It involves **organizing the system into multiple layers**, where each layer will provision a specific service. Each layer communicated with the adjacent layers using certain well-defined protocols without affecting the integrity of the system. A hierarchical structure is obtained where each layer abstracts the underlying complexity of lower layers.

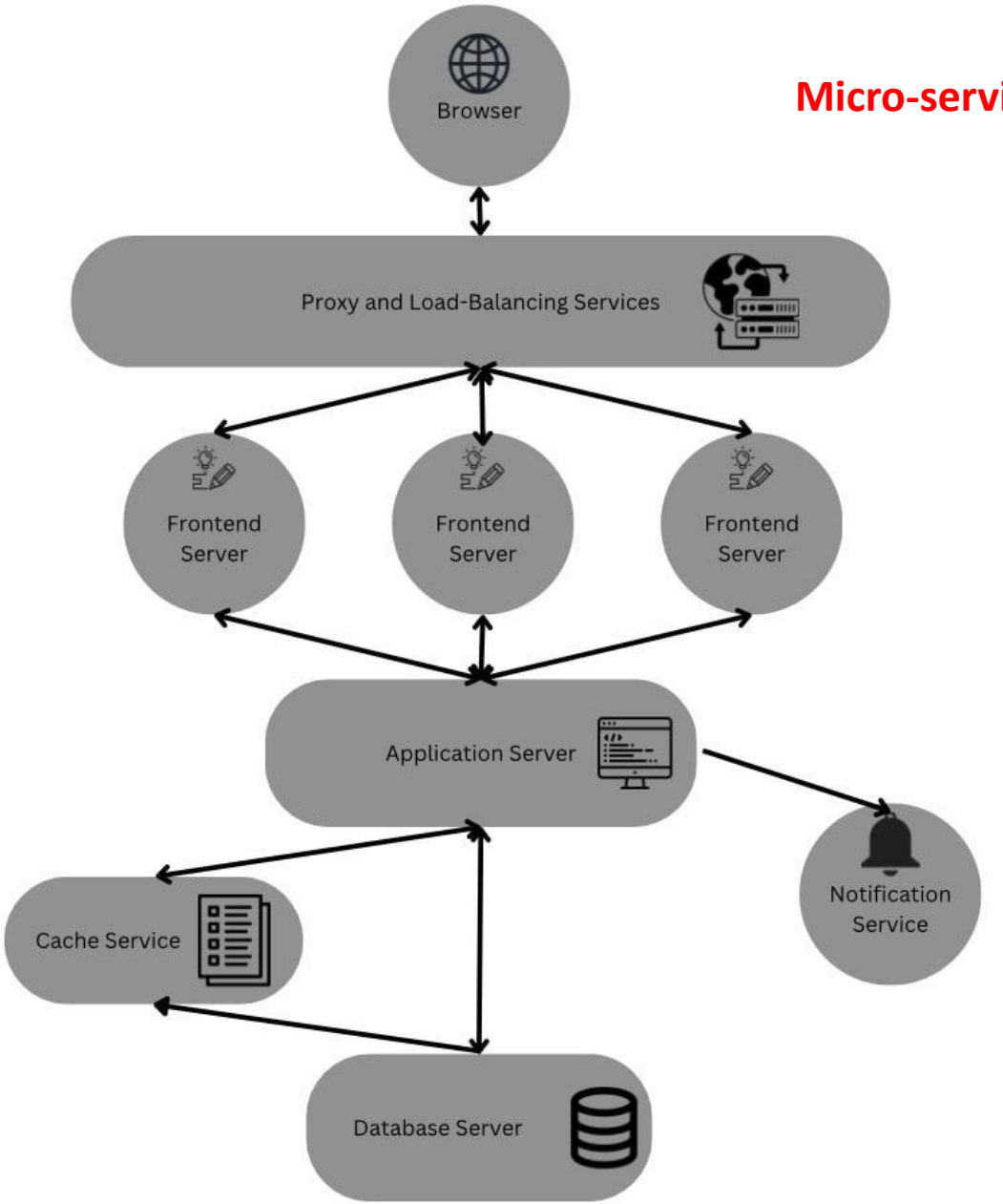


Various Layers of the TCP/ IP Model

Distributed System Models

- **Micro-services model** – In this system, a **complex application or task, is decomposed into multiple independent tasks** and these services running on different servers.
- **Each service performs only a single function** and is focused on a specific business-capability.
- This makes the overall system more maintainable, scalable and easier to understand.
- Services can be independently developed, deployed and scaled **without affecting the ongoing services.**

Distributed System Models



Micro-services model

Distributed System Models

- **Fundamental Model**
- The fundamental model in a distributed computing system is a **broad conceptual framework that helps in understanding the key aspects of the distributed systems.**
- These are concerned with more formal description of properties that are generally common in all architectural models.
- It represents the essential components that are required to understand a distributed system's behavior.

Three fundamental models are as follows:

- **Interaction Model – Distributed computing systems are full of many processes interacting with each other in highly complex ways.**
- Interaction model provides a framework to understand the mechanisms and patterns that are used for communication and coordination among various processes.

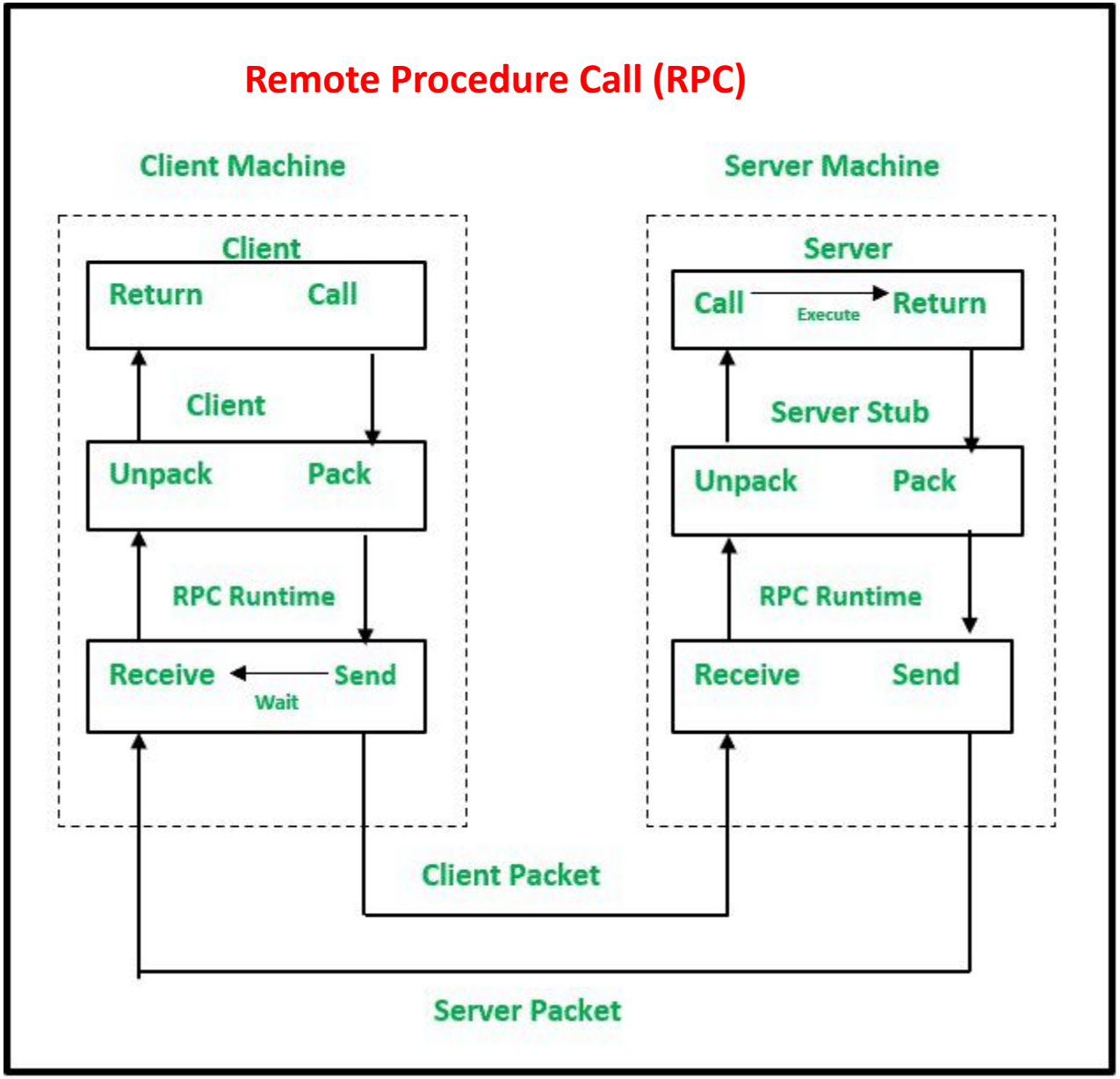
Distributed System Models

- **Different components that are important in this model are –**
  - **Message Passing –** It **deals with passing messages that may contain, data, instructions, a service request, or process synchronization between different computing nodes**. It may be synchronous or asynchronous depending on the types of tasks and processes.
  - **Publish/Subscribe Systems –** Also known as pub/sub system. In this the publishing process can publish a message **over a topic and the processes that are subscribed to that topic can take it up and execute the process for themselves.** It is more important in an event-driven architecture.

Distributed System Models

- **Remote Procedure Call (RPC)** – It is a communication paradigm that has an ability to invoke a **new process or a method on a remote process as if it were a local procedure call.**
- The client process makes a procedure call using RPC and **then the message is passed to the required server process using communication protocols.**
- These message passing protocols are abstracted and the result once obtained from the server process, is sent back to the client process to continue execution.

Distributed System Models



Distributed System Models

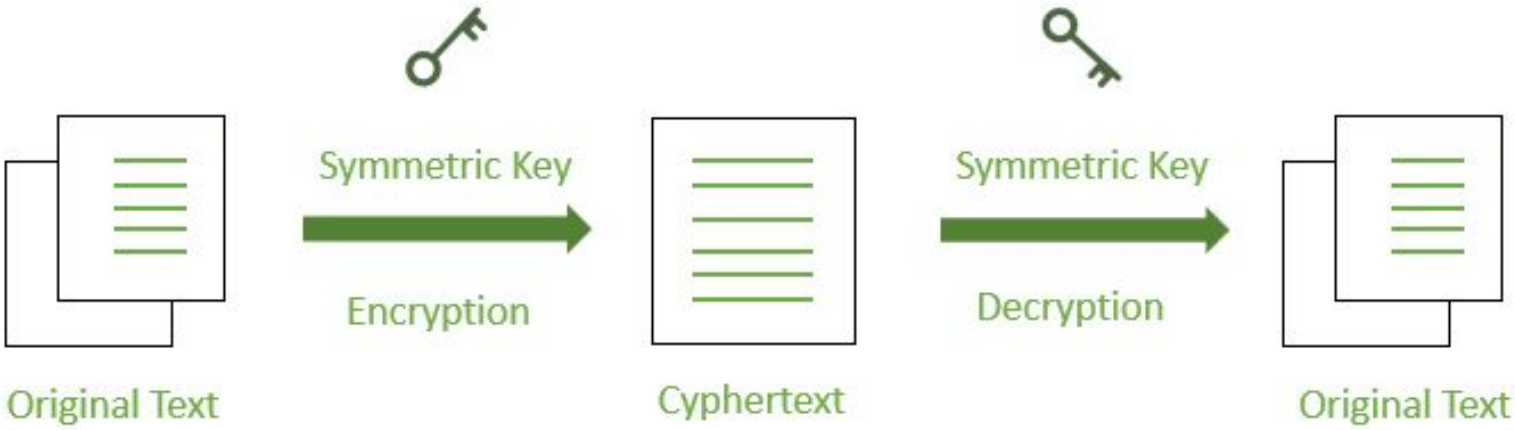
- **Failure Model** – This model addresses the **faults and failures that occur in the distributed computing system.** It provides a framework to identify and rectify the faults that occur or may occur in the system.
- Fault tolerance mechanisms are implemented so as to handle failures by replication and error detection and recovery methods.
- **Different failures that may occur are:**
  - **Crash failures –** A process or node unexpectedly stops functioning.
  - **Omission failures –** It involves a loss of message, resulting in absence of required communication.
  - **Timing failures –** The process deviates from its expected time quantum and may lead to delays or unsynchronized response times.
  - **Byzantine failures –** The process may send malicious or unexpected messages that conflict with the set protocols.

Distributed System Models

- **Security Model –** Distributed computing systems may suffer **malicious attacks, unauthorized access and data breaches.** Security model provides a framework for understanding the security requirements, threats, vulnerabilities, and mechanisms to safeguard the system and its resources. Various aspects that are vital in the security model are – **Authentication –** It verifies the identity of the users accessing the system. It ensures that only the authorized and trusted entities get access. It involves –
  - **Password-based authentication –** Users provide a unique password to prove their identity.
  - **Public-key cryptography –** Entities possess a private key and a corresponding public key, allowing verification of their authenticity.
  - **Multi-factor authentication –** Multiple factors, such as passwords, biometrics, or security tokens, are used to validate identity.

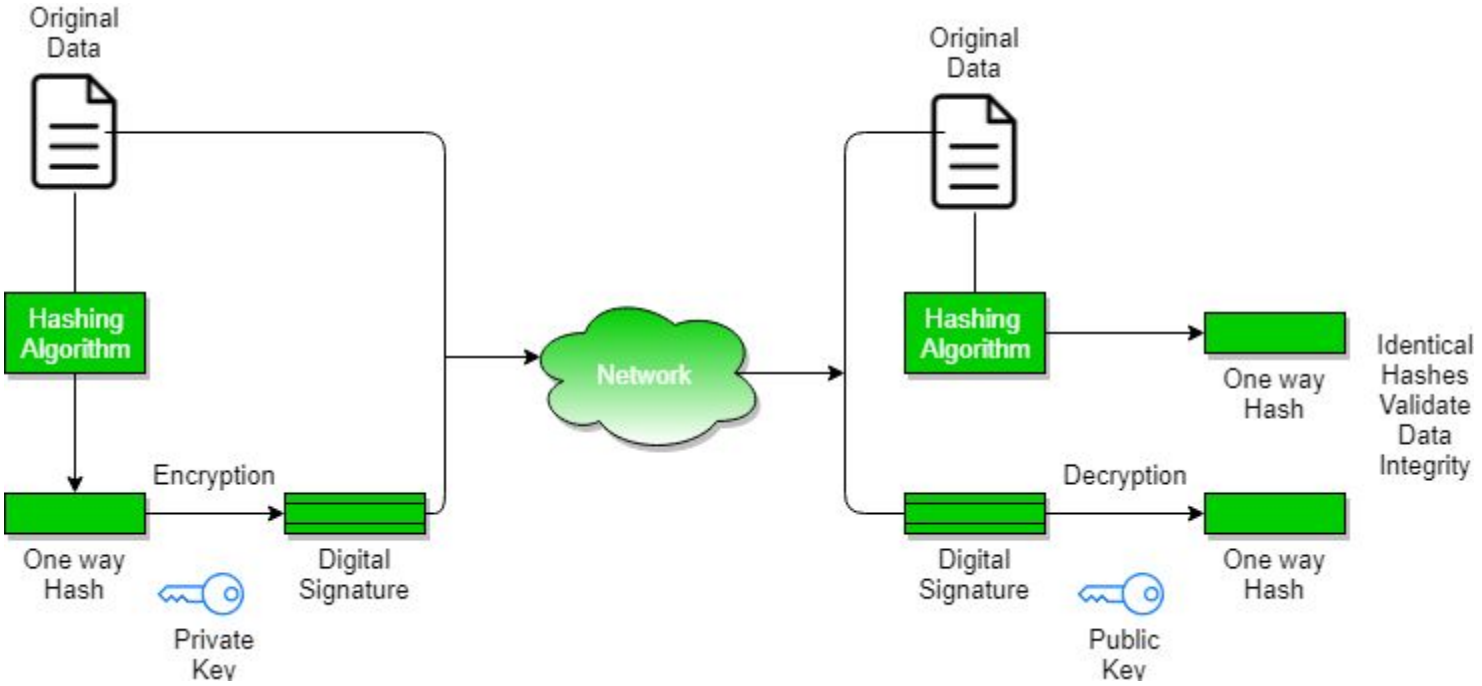
Distributed System Models

- **Encryption –** It is the process of transforming data into a format that is unreadable without a decryption key.
- **It protects sensitive information from unauthorized access or disclosure.**



Distributed System Models

- **Data Integrity** – Data integrity mechanisms protect against **unauthorized modifications or tampering of data.** They ensure that data remains unchanged during storage, transmission, or processing.
- Data integrity mechanisms include:
  - **Hash functions –** Generating a hash value or checksum from data to verify its integrity.
  - **Digital signatures –** Using cryptographic techniques to sign data and verify its authenticity and integrity.

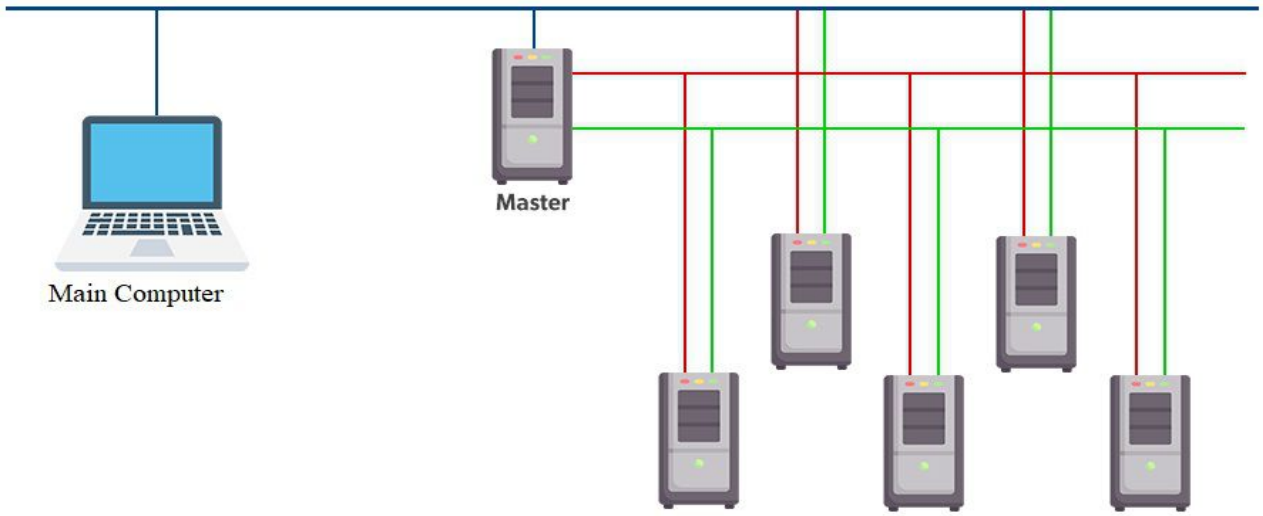


Ways of Distributed Systems

- A distributed system is also known as **distributed computer science and distributed databases**; independent components that interact with other different machines that exchange messages to achieve common goals.
- As such, the distributed system appears to the end-user like an interface or a computer.
- **Together the system can maximize resources and information while preventing system failure and did not affect service availability.**

Ways of Distributed Systems

- **1. Distributed Computing System**
- This distributed system is used in performance computation which requires high computing.
- **Cluster Computing:** Cluster Computing is a collection of connected computers that work together as a unit to perform operations together, functioning in a single system.
- Clusters are generally connected quickly via local area networks & each node is running the same operating system.



Ways of Distributed Systems

- When input comes from a client to the main computer, the **master CPU divides the task into simple jobs and sends it to the slave note to do it when the jobs are done by the slave nodes**, they send it back to the master node, and then it shows the result to the main computer.

Advantages of Cluster Computing

- High Performance
- Easy to manage
- Scalable
- Expandability
- Availability
- Flexibility
- Cost-effectiveness
- Distributed applications

Ways of Distributed Systems

Disadvantages of Cluster Computing

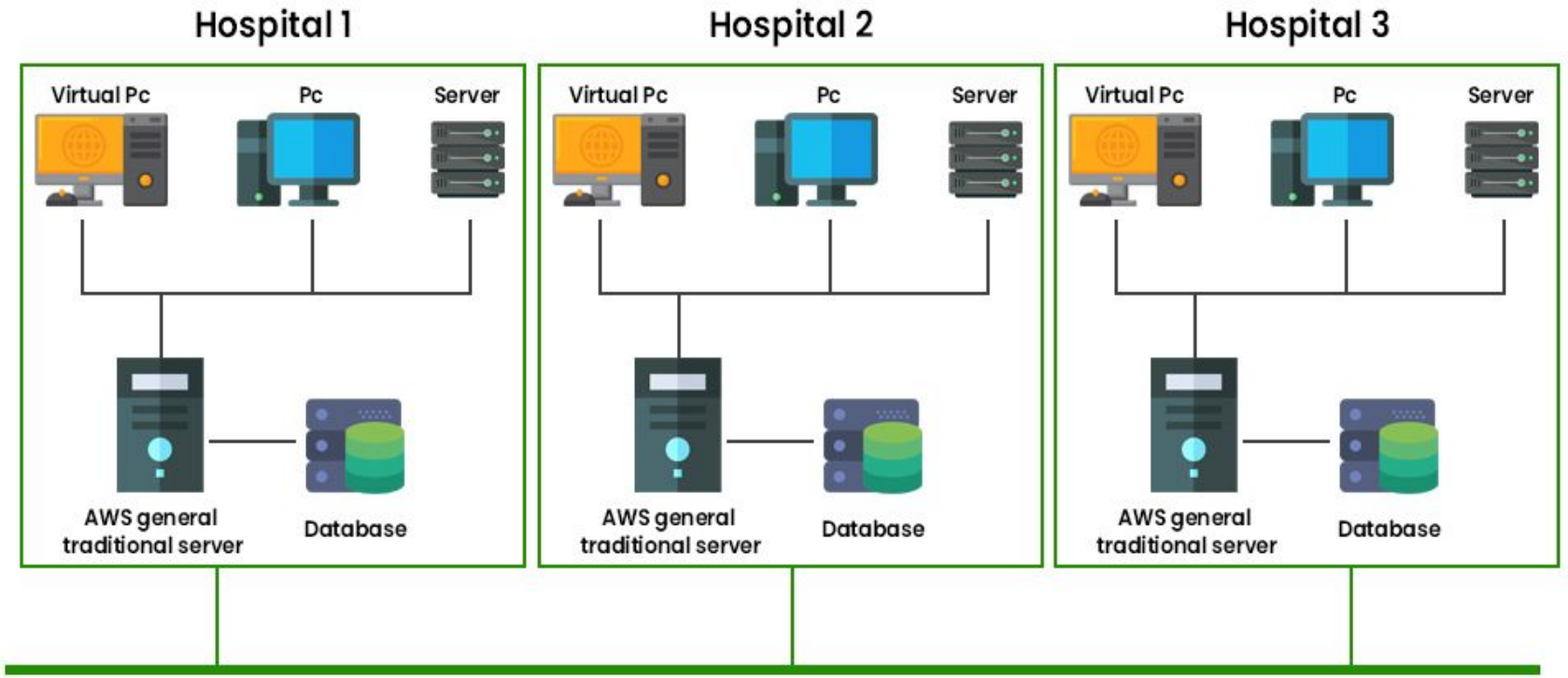
- High cost.
- The problem is finding the fault.
- More space is needed.
- The increased infrastructure is needed.
- In distributed systems, it is challenging to provide adequate security because both the nodes and the connections must be protected.

Applications of Cluster Computing

- In many web applications functionalities such as Security, Search Engines, Database servers, web servers, proxy, and email.
- It is flexible to allocate work as small data tasks for processing.
- Assist and help to solve complex computational problems.
- Cluster computing can be used in weather modeling.
- Earthquake, Nuclear, Simulation, and tornado forecast.

Ways of Distributed Systems

- **Grid Computing:** In grid computing, the subgroup consists of distributed systems, which are often set up as a network of computer systems, each system can belong to a different administrative domain and can **differ greatly in terms of hardware, software, and implementation network technology.**



Ways of Distributed Systems

- The different department has different computer with different OS to make the control node present which helps different computer with different OS to communicate with each other and transfer messages to work.
- **Advantages of Grid Computing**
  - Can solve bigger and more complex problems in a shorter time frame. Easier collaboration with other organizations and better use of existing equipment.
  - Existing hardware is used to the fullest.
  - Collaboration with organizations made easier
- **Disadvantages of Grid Computing**
  - Grid software and standards continue to evolve.
  - Getting started learning curve.
  - Non-interactive job submission.
  - You may need a fast connection between computer resources.
  - Licensing on many servers can be prohibitive for some applications.

Ways of Distributed Systems

- **Applications of Grid Computing**
- Organizations that develop grid standards and practices for the guild line.
- Works as a middleware solution for connecting different businesses.
- It is a solution-based solution that can meet computing, data, and network needs.

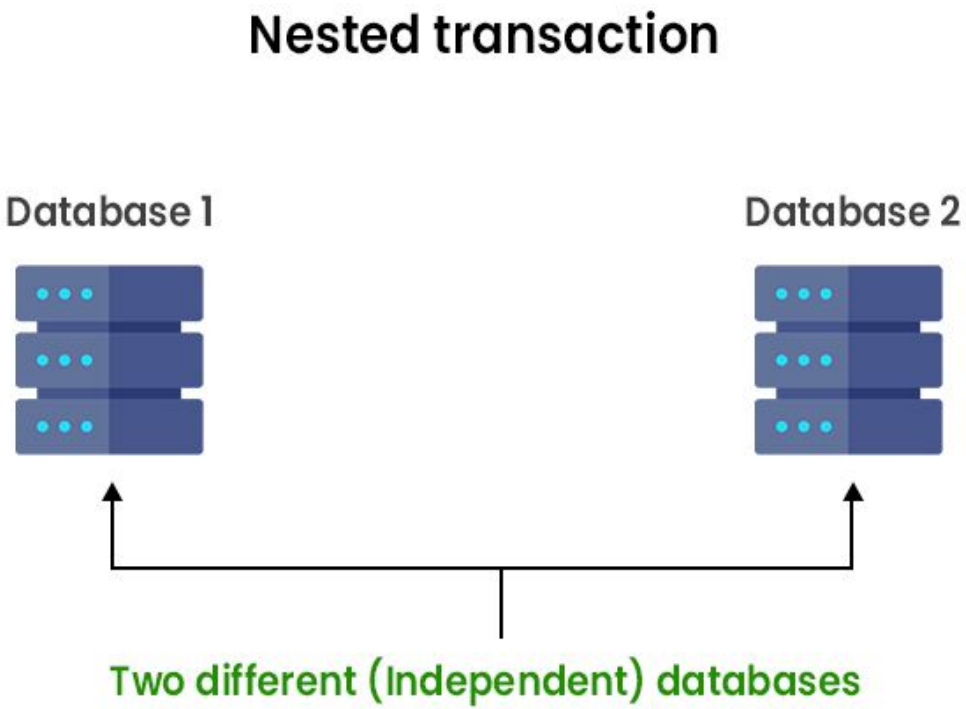
Ways of Distributed Systems

2. Distributed Information System

- **Distributed transaction processing:** It works across different servers using multiple communication models. The four characteristics (**ACID properties**) that transactions have:
  - **Atomic:** the transaction taking place must be indivisible to the others.
  - **Consistent:** The transaction should be consistent after the transaction has been done.
  - **Isolated:** A transaction must not interfere with another transaction.
  - **Durable:** Once an engaged transaction, the changes are permanent. Transactions are often constructed as several sub-transactions, jointly forming a nested transaction.

Ways of Distributed Systems

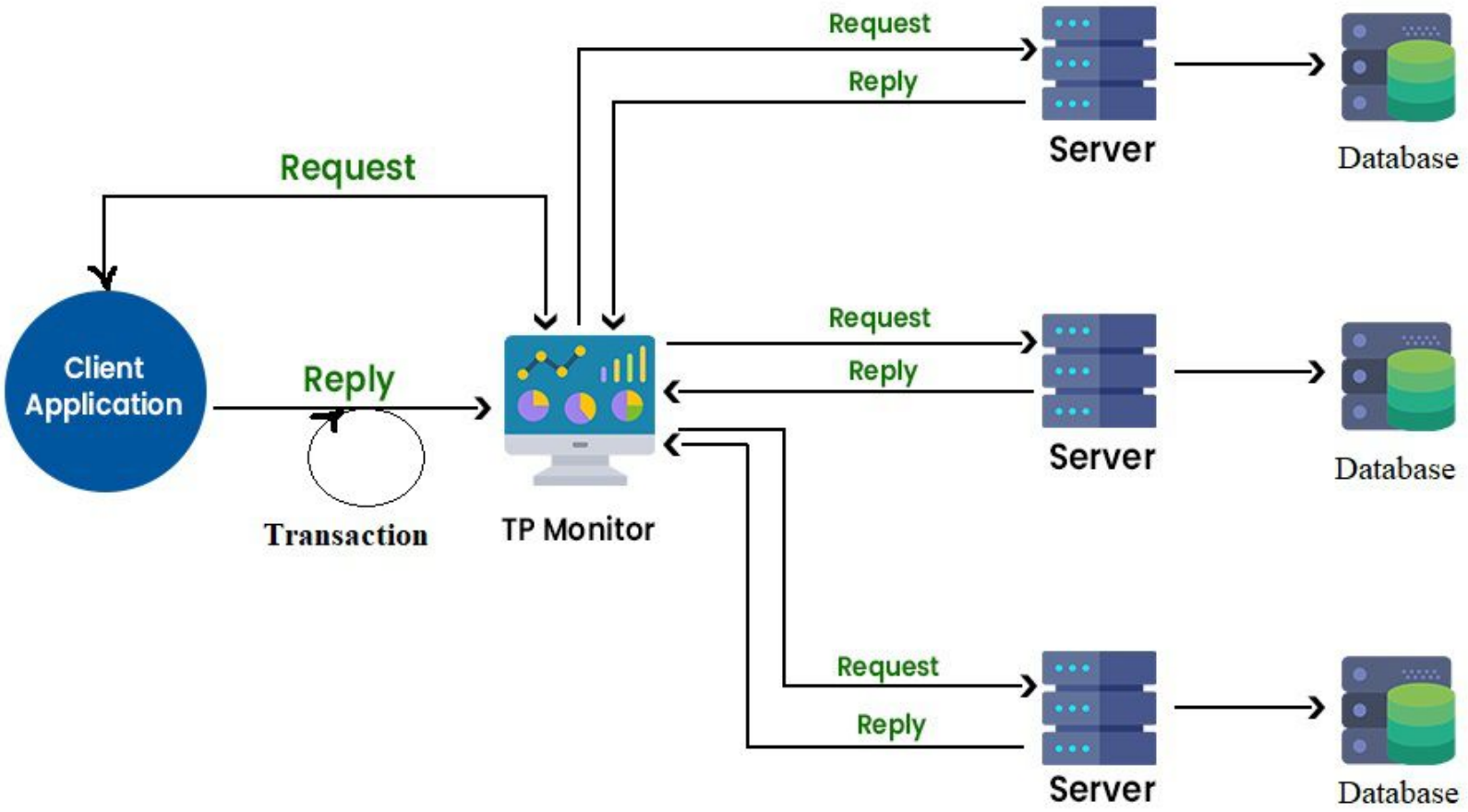
2. Distributed Information System



Ways of Distributed Systems

- Each database can perform its query containing data retrieval from two different databases to give one single result
- In the company's middleware systems, the component that manages distributed (or nested) transactions has formed the application integration core at the server or database.
- This was referred to as the Transaction Processing Monitor(TP Monitor).
- Its main task was to allow an application to access multiple servers/databases by providing a transactional programming model.
- Many requests are sent to the database to get the result, to ensure each request gets successfully executed and deliver result to each request, this work is handled by the TP Monitor.

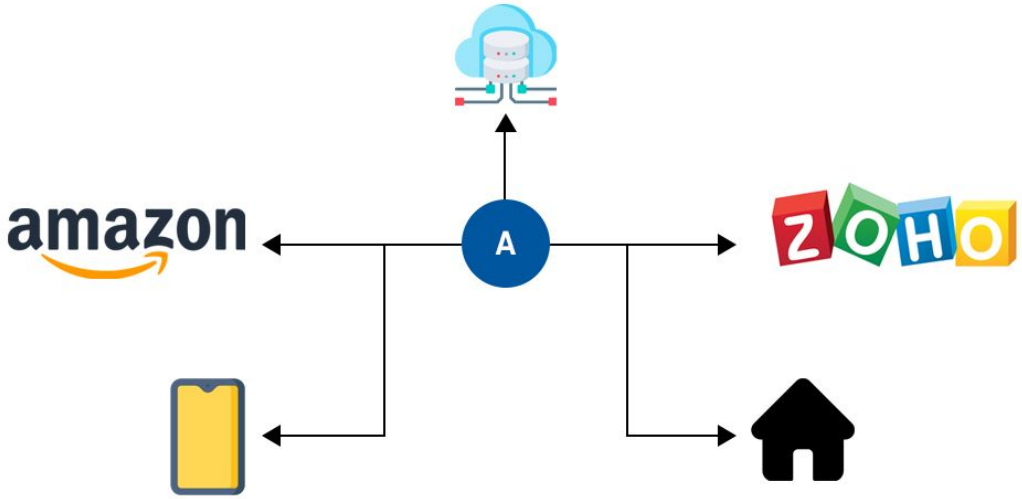
Ways of Distributed Systems



Distributed Transaction Processing

Ways of Distributed Systems

- **Enterprise Application Integration:** Enterprise Application Integration (EAI) is the process of bringing different businesses together. The databases and workflows associated with business applications ensure that the business uses information consistently and that changes in data done by one business application are reflected correctly in another's. Many organizations collect different data from different plate forms in the internal systems and then they use those data are used in the Trading system /physical medium.

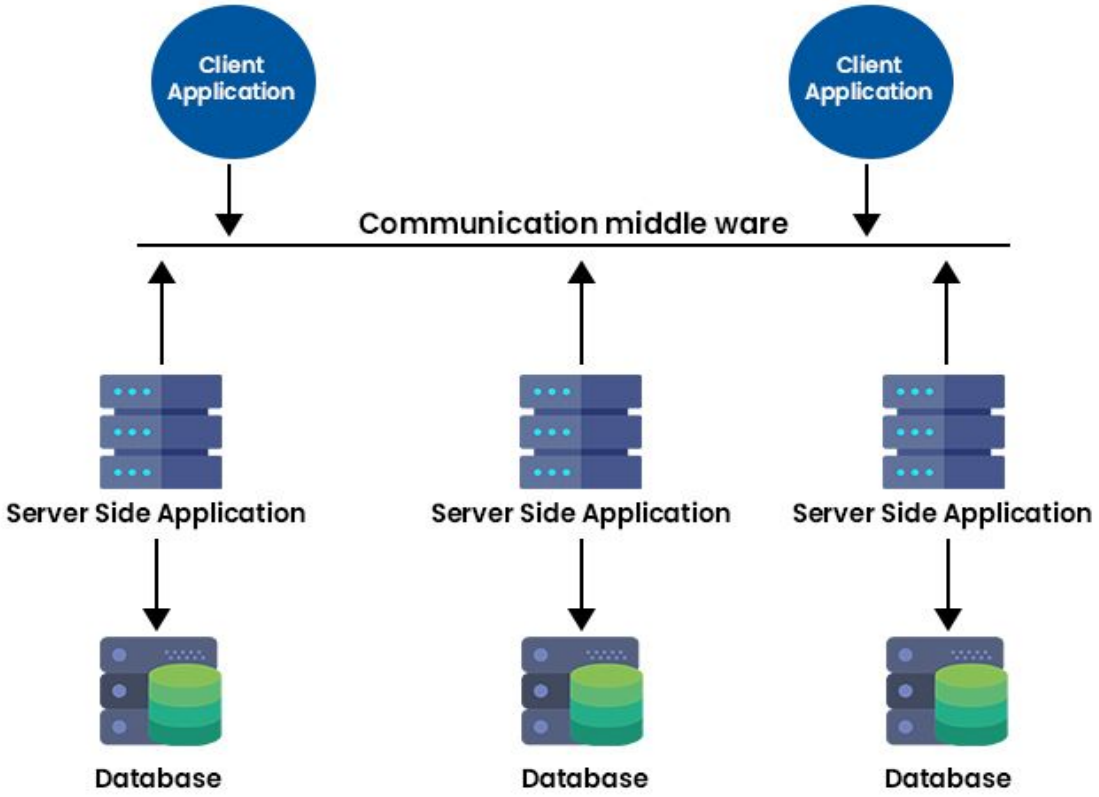


Ways of Distributed Systems

- **RPC:** Remote Procedure Calls (RPC), a software element that sends a request to every other software element with the aid of using creating a nearby method name and retrieving the data Which is now known as remote method invocation (RMI).
- An app can have a different database for managing different data and then they can communicate with each other on different platforms.
- Suppose, if you login into your android device and watch your video on YouTube then you go to your laptop and open YouTube you can see the same video is in your watch list.
- RPC and RMI have the disadvantage that the sender and receiver must be running at the time of communication.

Ways of Distributed Systems

- **RPC:**



Purposes

Targets the application rules and implements them in the EAI system so that even if one of the lines of business applications is replaced by the application of another vendor.

An EAI system can use a group of applications as a front end, provide only one, consistent access interface to those applications, and protect users from learning how to use different software packages.

Ways of Distributed Systems

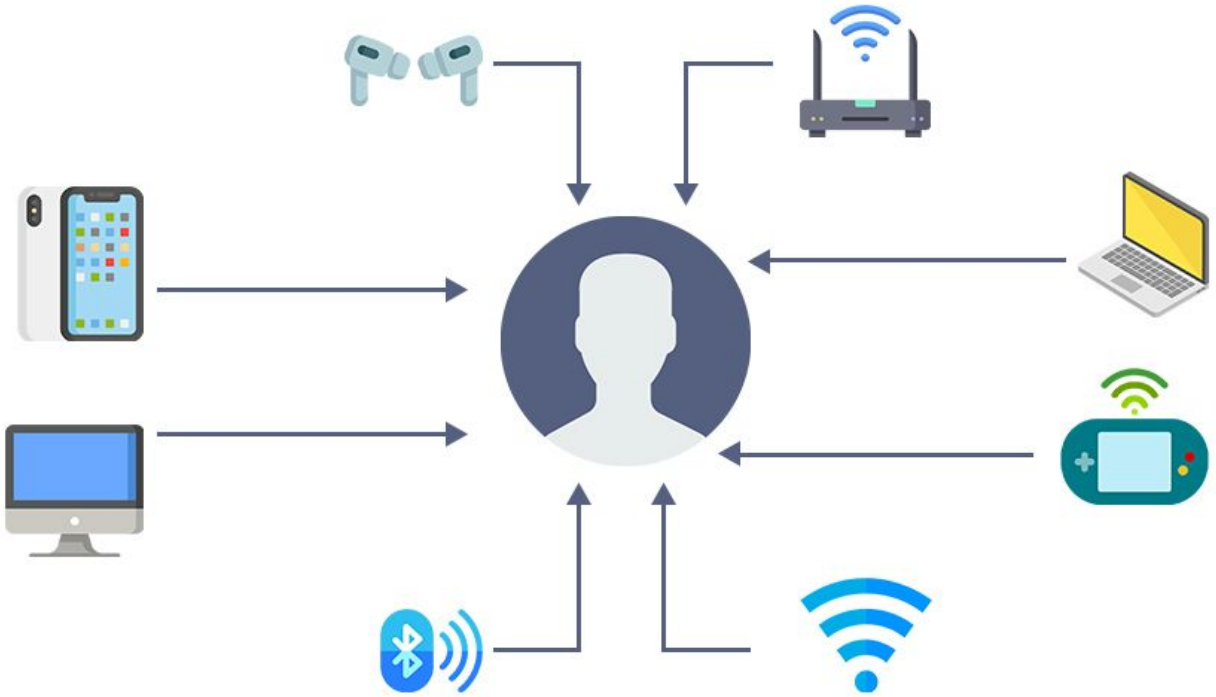
3. Distributed Pervasive System

- Pervasive Computing is also abbreviated as ubiquitous (Changed and removed) computing and it is the new step towards integrating everyday objects with microprocessors so that this information can communicate.
- A computer system available anywhere in the company or as a generally available consumer system that looks like that same everywhere with the same functionality but that operates from computing power, storage, and locations across the globe.
- **Home system:** Nowadays many devices used in the home are digital so we can control them from anywhere and effectively.

Ways of Distributed Systems

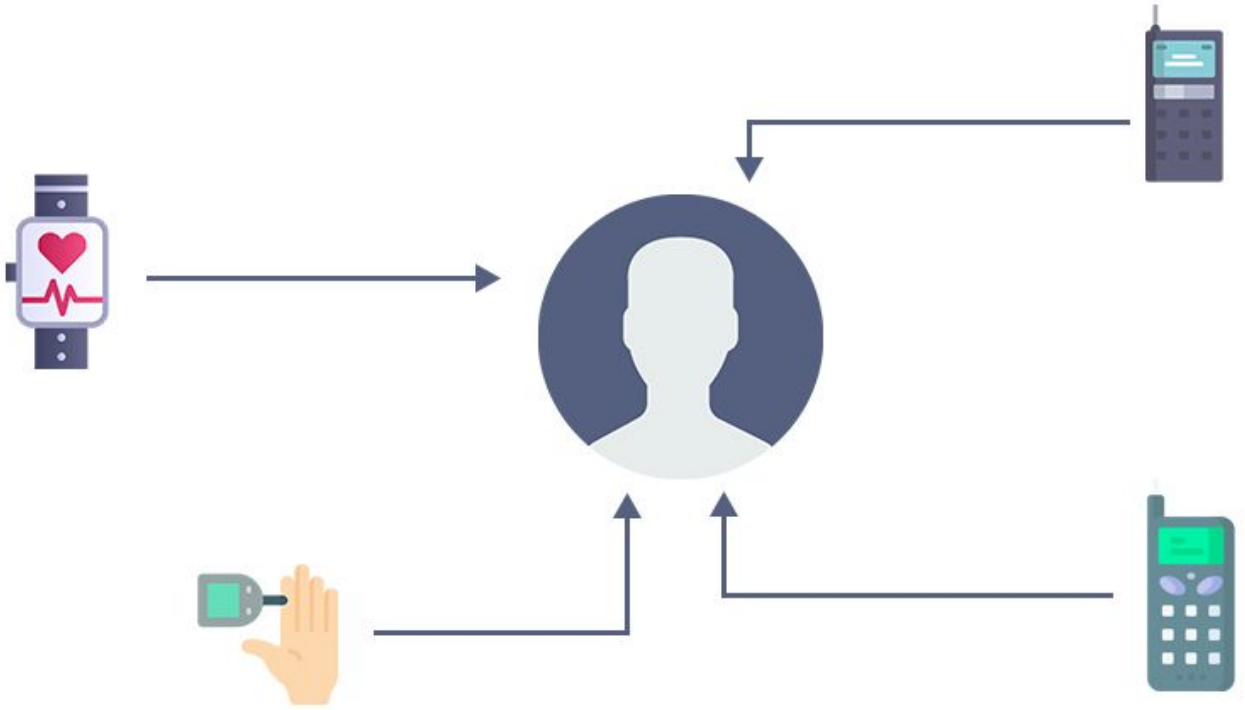
3. Distributed Pervasive System

Home system: Nowadays many devices used in the home are digital so we can control them from anywhere and effectively.



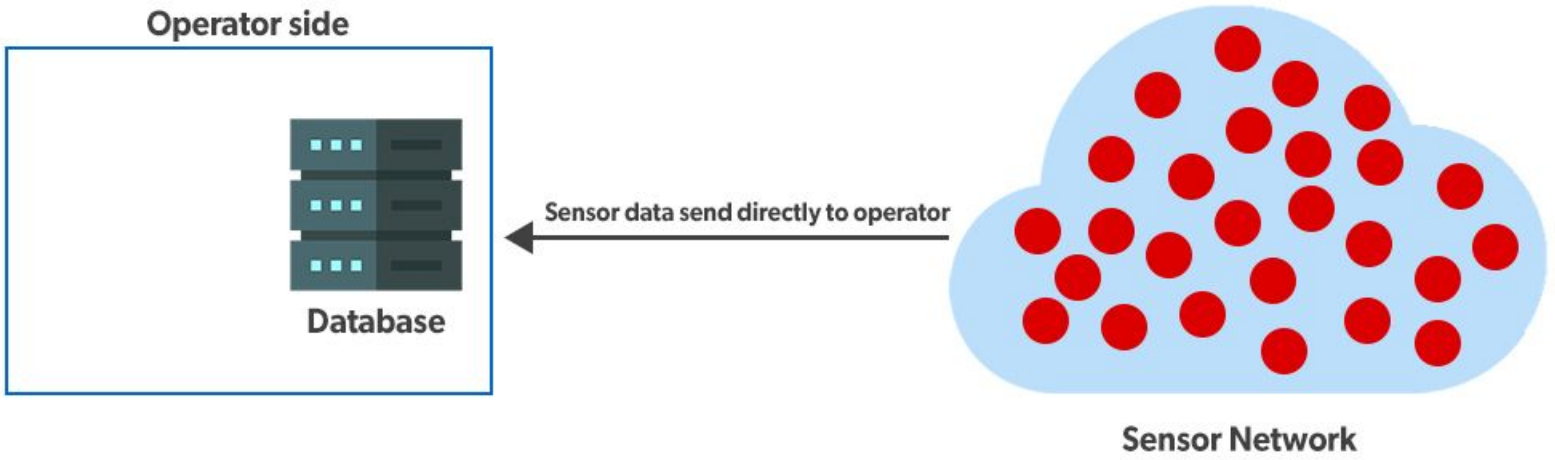
Ways of Distributed Systems

- **Electronic Health System:** Nowadays smart medical wearable devices are also present through which we can monitor our health regularly.



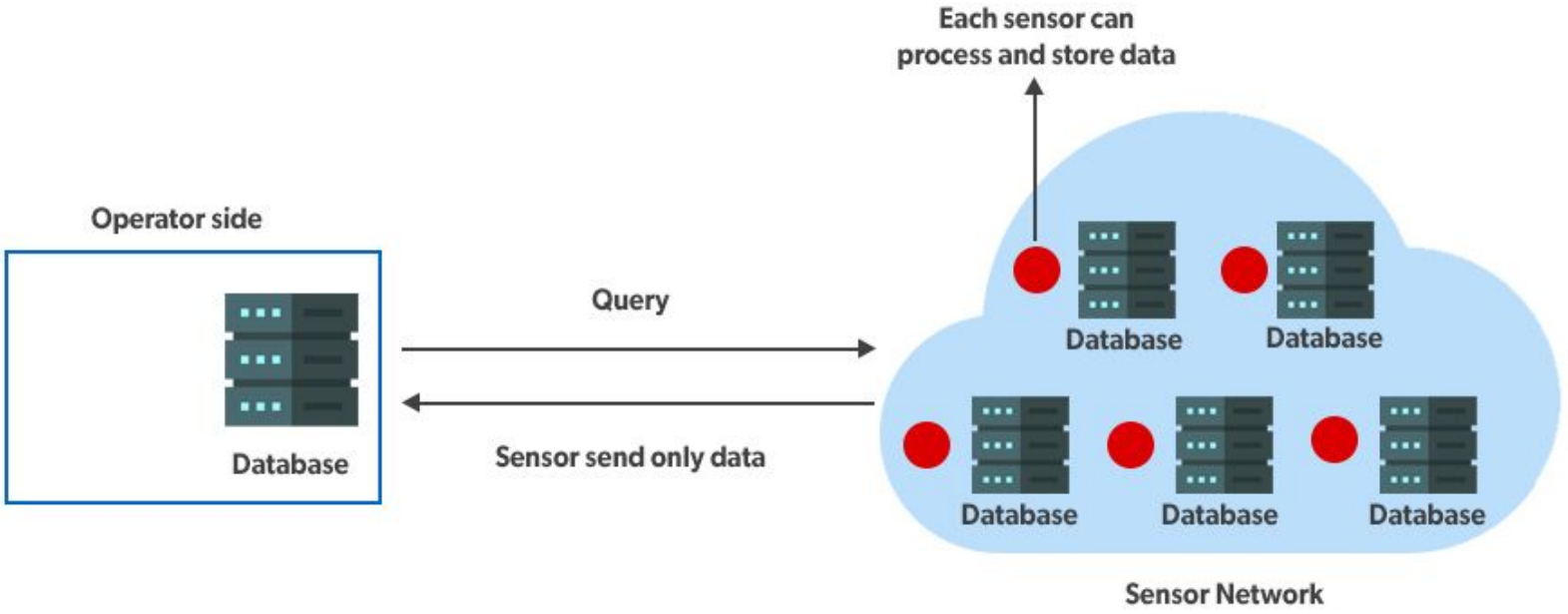
Ways of Distributed Systems

- **Sensor Network (IoT devices):** Internet devices only send data to the client to act according to the data send to the device.



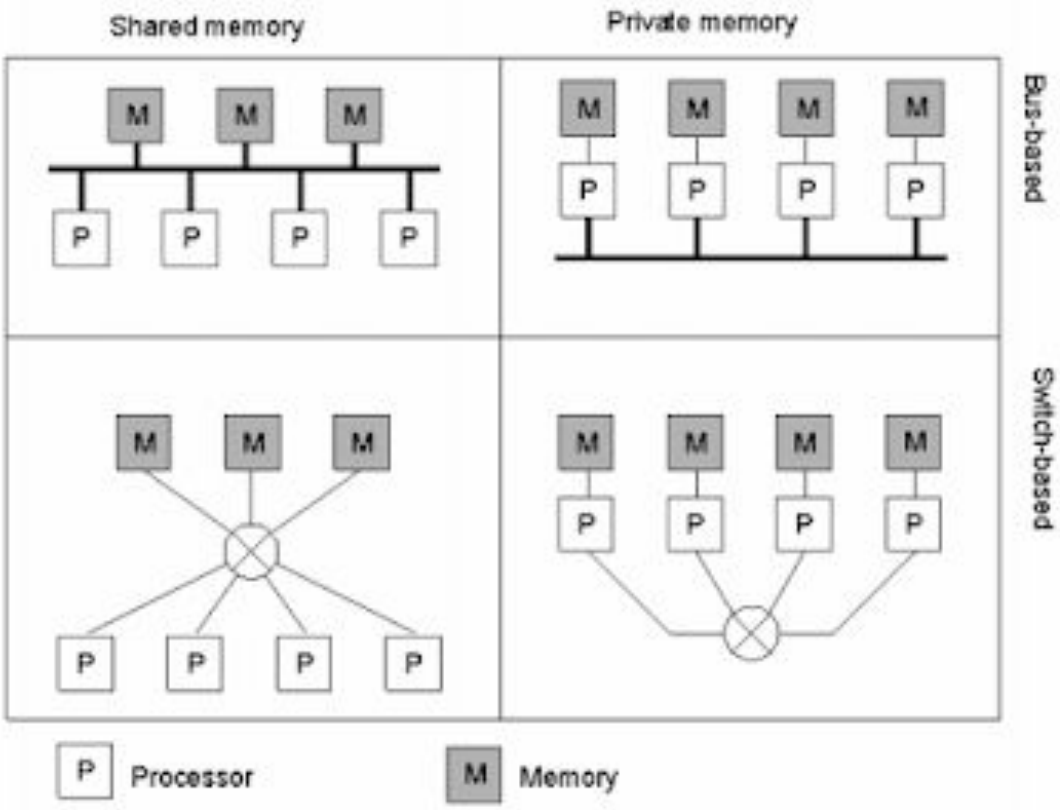
Ways of Distributed Systems

- Before sensory devices only send and send data to the client but now, they can store and process the data to manage it efficiently.

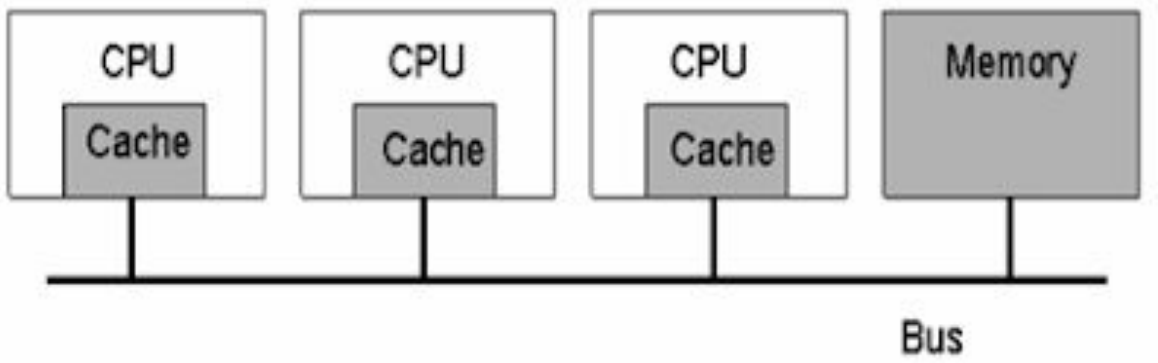


Distributed Systems Hardware & Software Concept

- **Hardware in Distributed Systems** can be organized in several different ways:
  - Shared Memory (Multiprocessors , which have a single address space).
  - Private Memory (Multicomputer, each CPU has a direct connection to its local memory).



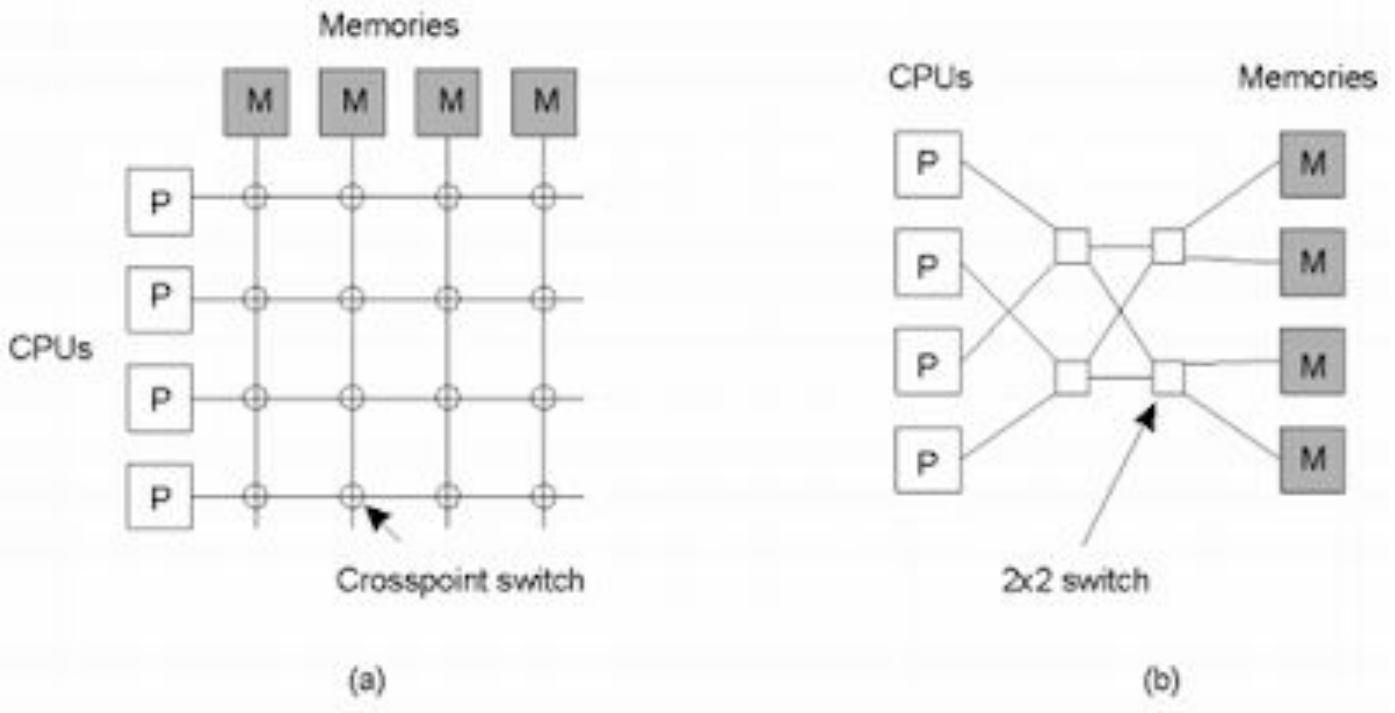
- **Multiprocessors – Bus Based**
- Have limited scalability
- Cache Memory help avoid bus overloading.



- **Multiprocessors – Switch Based**
- Different CPUs can access different memories simultaneously
- The number of switches limits the number of CPUs that can access memory simultaneously

  a) A crossbar switch

  b) An omega switching network



Distributed Systems Hardware & Software Concept

- **Multicomputer :-**
- **Homogeneous:**
  - All CPUs and memory are identical;
  - Connected through a broadcast shared multi access network (like Ethernet) in bus based systems;
  - Messages routed through an interconnection network in switch-based multicomputer (e.g., grids, hypercube…).
- **Heterogeneous:**
  - The most usual topology;
  - Computers may vary widely with respect to processor type, memory size, I/O bandwidth;
  - Connections are also diverse (a single multicomputer can simultaneously use LANs, Wide Area ATM, and frame relay networks);
  - Sophisticated software is needed to build applications due to the inherent heterogeneity;
  - Examples: SETI@home, WWW…

Distributed Systems Hardware & Software Concept

- **Software Concepts :-**
- **Uniprocessor Operating Systems**
- An OS acts as a resource manager or an arbitrator : Manages CPU, I/O devices, memory
- OS provides a virtual interface that is easier to use than hardware
- Structure of uniprocessor operating systems: Monolithic (e.g., MS-DOS, early UNIX)
- One large kernel that handles everything: Layered design
- Functionality is decomposed into N layers
- Each layer uses services of layer N-1 and implements new service(s) for layer N+1

Design Issues of Distributed System

- Distributed System is a collection of autonomous computer systems that are physically separated but are connected by a **centralized computer network that is equipped with distributed system software.**
- These are used in numerous applications, such as online gaming, web applications, and cloud computing.
- However, creating a distributed system is not simple, and there are a number of design considerations to take into account.

Design Issues of Distributed System

- **Heterogeneity**: Heterogeneity is applied to the network, computer hardware, operating system, and implementation of different developers.
- A key component of the heterogeneous distributed system client-server environment is middleware. Middleware is a set of services that enables applications and end-user to interact with each other across a heterogeneous distributed system.
- **Openness**: The openness of the distributed system is determined primarily by the degree to which new resource-sharing services can be made available to the users.
- Open systems are characterized by the fact that their key interfaces are published. It is based on a uniform communication mechanism and published interface for access to shared resources. It can be constructed from heterogeneous hardware and software.

Design Issues of Distributed System

- **Scalability**: The scalability of the system should remain efficient even with a significant increase in the number of users and resources connected.
- It shouldn't matter if a program has 10 or 100 nodes; performance shouldn't vary.
- A distributed system's scaling requires consideration of a number of elements, including size, geography, and management.
- **Security**: The security of an information system has three components Confidentially, integrity, and availability.
- Encryption protects shared resources and keeps sensitive information secrets when transmitted.

Design Issues of Distributed System

- **Failure Handling**: When some faults occur in hardware and the software program, it may produce incorrect results or they may stop before they have completed the intended computation so corrective measures should to implemented to handle this case.
- Failure handling is difficult in distributed systems because the failure is partial i, e, some components fail while others continue to function.
- **Concurrency**: There is a possibility that several clients will attempt to access a shared resource at the same time. Multiple users make requests on the same resources, i.e. read, write, and update.
- Each resource must be safe in a concurrent environment. Any object that represents a shared resource in a distributed system must ensure that it operates correctly in a concurrent environment.

Design Issues of Distributed System

- **Transparency**: Transparency ensures that the distributed system should be perceived as a single entity by the users or the application programmers rather than a collection of autonomous systems, which is cooperating.
- The user should be unaware of where the services are located and the transfer from a local machine to a remote one should be transparent.