EXPERIMENT NO. 1

AIM – Study of Distributed Computing System Architecture

Tools – MS Word

Introduction to Distributed Computing –

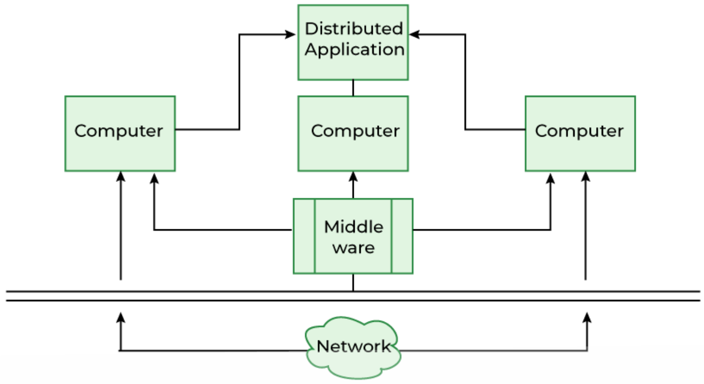
Distributed computing is a model in which multiple independent computers, often located in different physical locations, collaborate to solve a common problem or perform a task. These systems work together to complete processes by sharing computational resources, such as CPU, memory, and storage, across a network.

Key concepts in distributed computing include:

- **Decentralization:** In a distributed system, no single machine or server has complete control over the entire system. Instead, tasks are divided across multiple systems that communicate and synchronize to achieve common goals.
- **Concurrency:** Multiple tasks or processes are executed simultaneously across different machines, allowing for faster completion of tasks and better resource utilization.
- **Fault Tolerance:** Distributed systems are designed to continue functioning even if some components fail. Redundancy, replication, and error handling mechanisms ensure reliability and minimize downtime.
- **Scalability:** Distributed systems can scale horizontally, meaning more machines can be added to handle increased workload without significantly affecting performance.
- **Transparency:** The system hides the complexity of distributed components from the user. The user interacts with the system as if it were a single entity, without needing to understand the underlying distribution.

Distributed systems have several architectural components that work together:

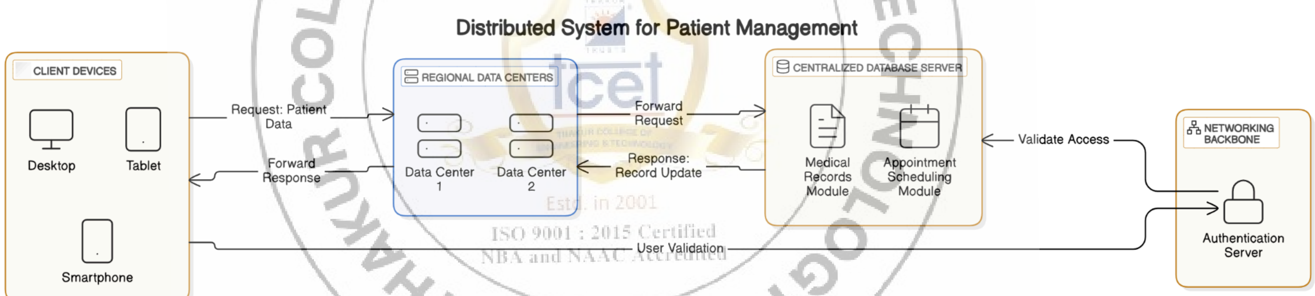
- **Nodes:** The individual computers or devices in the network that perform computations, store data, or handle communication.
- **Communication Network:** The infrastructure (e.g., the internet, LAN) that allows nodes to exchange data. Communication protocols such as TCP/IP are used to ensure reliable and secure data transfer.
- **Middleware:** Software that provides common services and capabilities for distributed applications, helping to manage tasks like load balancing, synchronization, and communication between nodes.
- **Databases and Data Storage:** Distributed databases are used to store and retrieve data across multiple nodes, ensuring consistency, availability, and partition tolerance (as per the CAP theorem).
- **Consistency Models:** Mechanisms to ensure that all nodes in the distributed system have a consistent view of data, even in the presence of network partitions or system failures.



Components of Distributed Computing

Distributed Computing in Healthcare

Distributed computing has revolutionized healthcare by enabling scalable solutions for complex medical challenges. From telemedicine to genomic analysis, distributed systems provide the computational power, accessibility, and reliability required to improve patient outcomes and streamline medical operations.



Distributed System for Patient Management (Client-Server Architecture)

Key Components –

- **Electronic Health Record (EHR) Servers**:
  Maintain centralized, secure databases for patient records, accessible to authorized personnel across different locations.
- **Diagnostic Servers**:
  Use distributed computing to run complex algorithms for imaging and lab test analysis, such as identifying anomalies in X-rays or predicting disease risks.
- **Telemedicine Servers**:
  Facilitate real-time video consultations and remote patient monitoring using IoT devices and sensors.

- **Regional Data Centers**:
  Handle localized data to reduce latency and comply with regional healthcare regulations.
- **Networking Backbone**:
  Ensures secure, encrypted communication between healthcare providers, patients, and diagnostic systems.

Working of the Architecture –

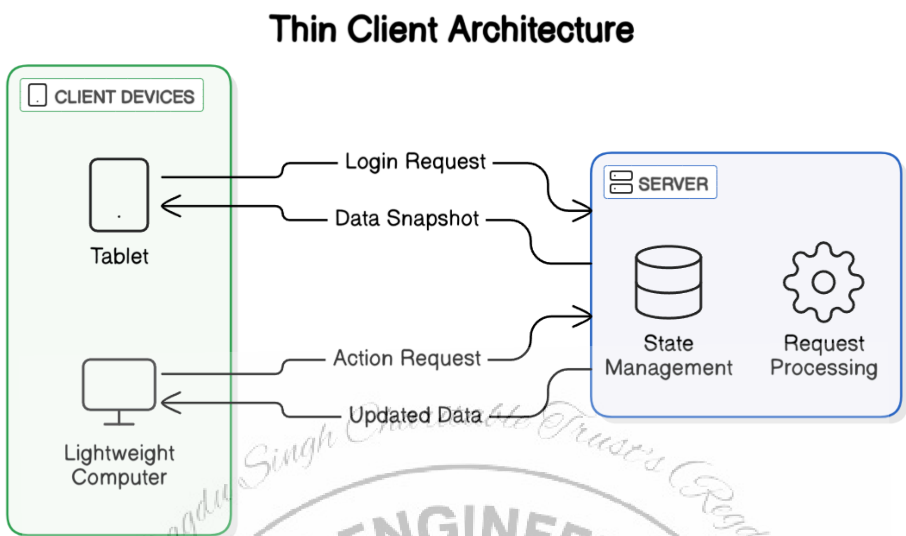
1. **Data Collection and Storage**:
   - Patient data is collected from devices, hospitals, or clinics and sent to EHR servers.
   - Local data centers process and store records while synchronizing with global servers for redundancy.
2. **Real-Time Diagnosis and Monitoring**:
   - Diagnostic servers process imaging data using AI models.
   - Telemedicine servers receive real-time vital signs and send alerts to healthcare providers when anomalies are detected.
3. **Remote Consultations**:
   - Patients connect to the nearest regional data center for low-latency video calls.
   - Doctors access patient data from EHR servers to provide informed recommendations.
4. **Global Synchronization**:
   - Updates made in one facility are synchronized across servers, ensuring consistency in patient records and treatment plans.

Applications in Healthcare

1. **Telemedicine Platforms**:
   Platforms like Amwell and Teladoc Health leverage distributed systems for secure video consultations and prescription management.
2. **Genomic Analysis**:
   Tools like GenomeCloud use distributed computing to analyze large genomic datasets, enabling personalized medicine.
3. **Remote Patient Monitoring**:
   IoT-enabled devices, such as heart rate monitors and glucose sensors, transmit real-time data to diagnostic servers for continuous monitoring.

Thin Client Architecture Solution

In a Thin Client approach, patient devices (e.g., smartphones, tablets) act as interfaces, while most computations and data processing occur on healthcare servers.

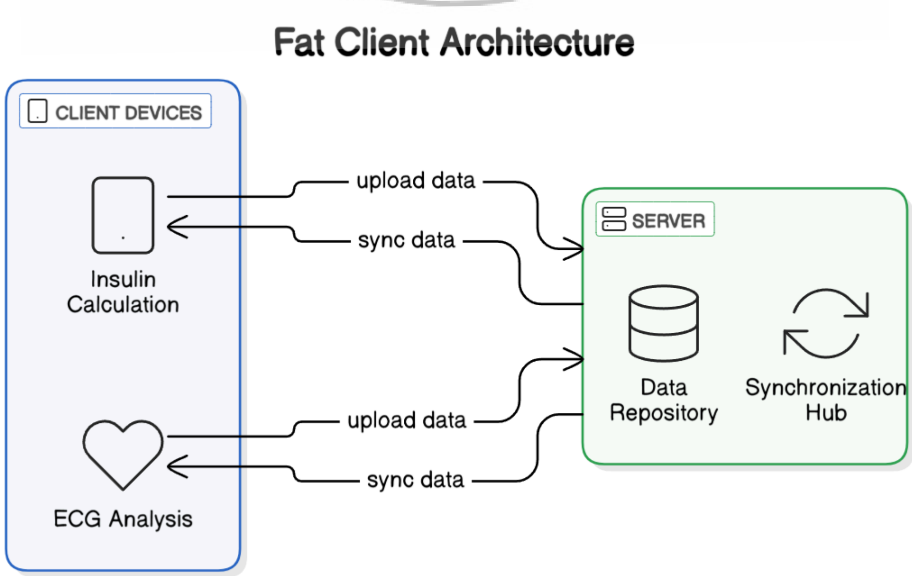


Working –

- **Server-Centric Data Management:** Servers hold all patient data and perform tasks like medical imaging analysis, drug interaction checks, and predictive modeling.
- **Device Interaction:** Patients use apps to access real-time updates, view health records, or communicate with doctors.
- **Benefits:** Lower hardware requirements for patients and enhanced security since sensitive data remains on the server.

Fat Client Architecture Solution

In a Fat Client approach, client devices handle significant processing, such as running local health applications or offline diagnostic tools.



Working –

- **Client-Managed Processing:** Devices process real-time data locally, such as calculating insulin doses or analyzing ECGs.
- **Server Role:** Servers synchronize patient data across platforms and ensure consistency.
- **Challenges:** Risk of data breaches if client devices are compromised.

Advantages of Distributed Computing

- **Scalability**: Easily accommodates growing patient data and processing demands.
- **Improved Diagnostics**: AI-powered servers analyze medical data faster and more accurately.
- **Global Accessibility**: Enables remote consultations and services in underserved regions.
- **Resilience**: Redundancy ensures no data loss during server failures.
- **Cost Efficiency**: Cloud-based solutions optimize resource allocation.

Disadvantages of Distributed Computing

- **Complexity**: Requires sophisticated design and robust security protocols.
- **Latency Issues**: Slow networks can delay critical diagnostics or alerts.
- **High Initial Costs**: Infrastructure setup can be expensive for smaller organizations.
- **Regulatory Compliance**: Systems must comply with regional laws like HIPAA or GDPR.

Learning Objectives –

LO1: Understand the basics of Architecture of Distributed Computing

LO2: Study the Architecture of Distributed Computing in Healthcare

Conclusion

For Faculty Use:

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				