UNIT – III

Virtualization, Programming Models for Cloud Computing and Software Development in Cloud

1. VIRTUALIZATION

1.1 Meaning of Virtualization

Virtualization is a fundamental technology in cloud computing that allows a single physical machine to create and run multiple virtual machines (VMs). Each virtual machine behaves like an independent computer with its own operating system and applications. In simple terms, virtualization divides a physical system into multiple virtual systems so that resources like CPU, memory, storage, and network can be efficiently shared.

Virtualization is the backbone of cloud computing because it enables resource sharing, scalability, and cost efficiency. Without virtualization, cloud providers would not be able to serve multiple users on the same physical hardware.

1.2 Types of Virtualization

1. Server Virtualization

Server virtualization divides a physical server into multiple virtual servers. Each virtual server runs independently and can host different applications.

Example:
AWS EC2 instances running multiple virtual servers on a single physical machine.

2. Storage Virtualization

Storage virtualization combines multiple physical storage devices into a single virtual storage system. Users see it as one storage unit even though data is stored in different locations.

Example:
Google Drive stores files across multiple data centers but appears as one storage space.

3. Network Virtualization

Network virtualization creates virtual networks from physical network resources. It allows multiple virtual networks to operate on the same physical infrastructure.

Example:
Virtual Private Networks (VPNs) used in organizations.

4. Desktop Virtualization

Desktop virtualization allows users to access a desktop environment remotely from any device.

Example:
Remote desktop services used in companies.

5. Application Virtualization

Application virtualization allows applications to run without being installed directly on the user's system.

Example:
Web-based applications like Microsoft Office Online.

1.3 Hypervisor (Virtual Machine Monitor)

A hypervisor is software that creates and manages virtual machines. It allocates resources to each VM and ensures isolation between them.

Types of Hypervisors

Type 1 (Bare Metal Hypervisor):
Runs directly on hardware.
Example: VMware ESXi, Microsoft Hyper-V

Type 2 (Hosted Hypervisor):
Runs on top of an operating system.
Example: Virtual Box, VMware Workstation

1.4 Advantages of Virtualization

Virtualization improves resource utilization by allowing multiple systems to run on a single physical machine. It reduces hardware cost, increases scalability, and improves system flexibility. It also provides better disaster recovery because virtual machines can be easily backed up and restored.

1.5 Disadvantages of Virtualization

Virtualization may introduce performance overhead because multiple VMs share the same hardware. It also requires powerful hardware systems and proper management tools. Security risks may arise if virtual machines are not properly isolated.

2. PROGRAMMING MODELS FOR CLOUD COMPUTING

2.1 Meaning

Programming models in cloud computing define how applications are designed and executed on distributed cloud systems. These models help developers process large-scale data efficiently by dividing tasks into smaller parts and executing them in parallel.

2.2 MAPREDUCE PROGRAMMING MODEL

2.2.1 Introduction

Map Reduce is a programming model used to process large datasets in a distributed computing environment. It was developed by Google and is widely used in cloud computing and big data processing.

Map Reduce divides a large task into smaller sub-tasks and processes them in parallel across multiple machines.

2.2.2 Working of Map Reduce

Map Reduce works in two main phases:

1. Map Phase

In the map phase, input data is divided into smaller chunks. Each chunk is processed by a mapper function that converts input data into key-value pairs.

Example:
Input text → "Cloud computing is powerful"
Output → (Cloud,1), (computing,1), (is,1), (powerful,1)

2. Reduce Phase

In the reduce phase, all intermediate key-value pairs are grouped together and aggregated to produce final output.

Example:
(Cloud,1), (Cloud,1) → (Cloud,2)

2.2.3 Map Reduce Architecture

Map Reduce consists of:

- Input data
- Splitting module
- Mapper nodes
- Shuffling and sorting
- Reducer nodes
- Final output

2.2.4 Example of Map Reduce

Word Count Problem:

- Input: Large text file
- Map: Break words and assign value 1
- Reduce: Count total occurrences of each word

2.2.5 Advantages of Map Reduce

Map Reduce is highly scalable and can process massive datasets efficiently. It provides fault tolerance because tasks can be re-executed if a node fails. It also simplifies distributed programming.

2.2.6 Disadvantages of Map Reduce

It is not suitable for real-time processing. It also has high latency because intermediate results are written to disk.

2.3 CLOUD HASKELL

2.3.1 Introduction

Cloud Haskell is a distributed programming framework based on the Haskell programming language. It is used to build concurrent and distributed applications in cloud environments.

Cloud Haskell allows communication between processes running on different machines.

2.3.2 Features of Cloud Haskell

Cloud Haskell supports message passing between distributed processes. It provides fault tolerance and process isolation. It is designed for high reliability in cloud systems.

2.3.3 Working Model

Cloud Haskell uses:

- Nodes (machines in cloud)
- Processes (lightweight threads)
- Message passing system

Processes communicate by sending messages instead of sharing memory.

2.3.4 Example

A distributed chat system where messages are sent between different servers using Cloud Haskell processes.

2.3.5 Advantages

Cloud Haskell provides strong fault tolerance and supports concurrent execution. It is suitable for building reliable distributed systems.

2.3.6 Disadvantages

It has a steep learning curve and limited industrial adoption compared to other frameworks.

3. SOFTWARE DEVELOPMENT IN CLOUD

3.1 Meaning

Software development in cloud refers to building, testing, deploying, and maintaining applications using cloud platforms instead of traditional local systems.

Cloud-based development provides flexibility, scalability, and collaboration features.

3.2 Cloud Software Development Life Cycle (SDLC)

Cloud SDLC includes the following phases:

1. Planning

Requirements are collected and analyzed.

2. Design

System architecture is designed for cloud deployment.

3. Development

Developers write code using cloud tools and platforms.

4. Testing

Applications are tested using cloud-based testing environments.

5. Deployment

Applications are deployed on cloud platforms like AWS or Azure.

6. Maintenance

Continuous updates and monitoring are done using cloud tools.

3.3 Tools Used in Cloud Development

Cloud development uses modern tools such as:

- GitHub for version control
- Docker for containerization
- Kubernetes for orchestration
- AWS/Azure/GCP for deployment
- Jenkins for CI/CD pipelines

3.4 Cloud Deployment Models in Development

Developers choose deployment models based on application needs:

- Public cloud for general applications
- Private cloud for secure applications
- Hybrid cloud for mixed requirements

3.5 Advantages of Cloud Software Development

Cloud development reduces infrastructure cost and improves collaboration among developers. It allows continuous integration and deployment (CI/CD), making software updates faster. It also provides scalability for handling large user bases.

3.6 Challenges in Cloud Development

Challenges include security risks, dependency on internet connectivity, vendor lock-in, and debugging distributed systems.

3.7 Micro services in Cloud

Micro services architecture breaks an application into small independent services.

Example:

- Login service
- Payment service
- Order service

Each service runs independently and communicates via APIs.

3.8 Advantages of Cloud Development

Cloud development reduces infrastructure cost and improves collaboration. It supports faster deployment using CI/CD pipelines. It also provides automatic scaling during high traffic.

CLOUD COMPUTING FRAMEWORKS

Meaning

Cloud computing frameworks are software environments that provide tools, libraries, and services to develop, deploy, and manage cloud applications easily. These frameworks simplify distributed computing by handling complexity such as communication, scalability, and fault tolerance.

Cloud frameworks act as a bridge between the developer and cloud infrastructure, allowing developers to focus on application logic instead of infrastructure management.

Types of Cloud Frameworks

1. Open Source Cloud Frameworks

These frameworks are freely available and allow developers to modify and customize cloud systems.

Examples:

- Apache Hadoop
- OpenStack
- Apache Spark

Example Use Case:
Hadoop is used for processing big data in distributed environments.

2. Proprietary Cloud Frameworks

These are developed and maintained by cloud providers and are not open source.

Examples:

- Amazon Web Services (AWS) SDK
- Microsoft Azure SDK
- Google Cloud SDK

Example Use Case:
Using AWS SDK to build scalable web applications.

CONTAINERIZATION IN CLOUD COMPUTING

Meaning

Containerization is a lightweight virtualization technique where applications and their dependencies are packaged into a container. These containers can run consistently across different computing environments.

Unlike traditional virtualization, containers share the same operating system kernel but run independently.

How Containerization Works

A container includes:

- Application code
- Libraries
- Dependencies
- Configuration files

Containers run on a container engine like Docker, which manages execution and isolation.

Example

A web application packaged in a Docker container can run on:

- Windows
- Linux
- Cloud platforms like AWS or Azure

without modification.

Popular Container Tools

Docker

Docker is the most widely used container platform used to create, deploy, and manage containers.

Kubernetes

Kubernetes is a container orchestration tool used to manage multiple containers across different servers.

Difference Between Virtualization and Containerization

Virtualization creates full virtual machines with separate operating systems, while containerization shares the host operating system and only isolates applications.

Virtual machines are heavier and slower, while containers are lightweight and faster.

Advantages of Containerization

Containerization provides fast deployment and portability. It uses fewer system resources compared to virtual machines. It ensures consistency across development, testing, and production environments.

Disadvantages

Containers are less secure than full virtual machines. Managing large-scale containers requires complex orchestration tools.