**NAME—KAUSHAL PHUTANE**

**ROLL-NO-19**

Q-1Explain in brief about Cloud Reference Model.

A1--The important characteristics of cloud computing is ability to deliver on demand a variety of services which can be diverse from each other. So, there is a different perception of cloud computing among the users. However, there is no uniformity in the perception among the users that we can define cloud computing into three major categories, and they are called cloud computing reference model.

**Type Of Cloud Computing Reference Model**

There are three type of cloud computing refrence model which are

**Infrastructure as a Service (IaaS)**

Platform as a Service (PaaS)

Software as a Services (SaaS)

##### **Infrastructure as a Service (IaaS)**

Iaas is the most basic category of cloud computing services. With Iaas, we can rent IT infrastructure servers, and virtual machines (VMs), storage, networks, and operating systems from a cloud provider on a pay-as-you-go basis.

It’s an instant computing infrastructure, provisioned and managed over the internet. Virtual hardware is provided on demand in the form of virtual machines instances. Pricing can be hourly basis.

Virtual storage is either raw disk space or an object store which is the higher level of abstraction entities rather than file.

##### **Platform as a Service (PaaS)**

PaaS is another category of cloud computing reference model. Paas provides an environment for building, testing, and deploying software applications.

The goal of PaaS is to help create an application as quickly as possible without having a focus on managing the underlying infrastructure.

PaaS models deliver scalable and elastic runtime environments on demand and host execution of applications. These services are backed by a core middleware platform that is responsible for creating an abstract environment where applications are deployed/executed.

Responsibility of service providers is to provide the scalability and manage fault tolerance whereas use focus on the logical part of application development which leverage the use of APIs and libraries provided by PaaS.

For Example We want to process data on spark engine For this We don’t have to install Spark environment and Scala environment if we are writing code in Scala language. Service providers already do this for users. We will be using cloud computing services as a platform.

##### **Software as a Services (SaaS)**

Saas is software that is centrally hosted and managed for the end customer. It allows users to connect to and use cloud-based apps over the internet.

Common examples are email, calendars, and office tools such as Microsoft Office 365.SaaS provides application and services on demand.

Most of the common functionalities of desktop applications (office automation, document management, photo editing, customer relationship management (CRM)) are provided via web browser which can make applications more scalable.

Applications are shared by multiple users. For example, social networking sites like Facebook, twitter which are hosted on cloud we just use them as software. Most of the social networking makes use of cloud-based infrastructures.

Q-2 Explain in brief about interprocess communication.

A-2 Interprocess communication is the mechanism provided by the operating system that allows processes to communicate with each other. This communication could involve a process letting another process know that some event has occurred or the transferring of data from one process to another.

Synchronization is a necessary part of interprocess communication. It is either provided by the interprocess control mechanism or handled by the communicating processes. Some of the methods to provide synchronization are as follows −

* **Semaphore**

A semaphore is a variable that controls the access to a common resource by multiple processes. The two types of semaphores are binary semaphores and counting semaphores.

* **Mutual Exclusion**

Mutual exclusion requires that only one process thread can enter the critical section at a time. This is useful for synchronization and also prevents race conditions.

* **Barrier**

A barrier does not allow individual processes to proceed until all the processes reach it. Many parallel languages and collective routines impose barriers.

* **Spinlock**

This is a type of lock. The processes trying to acquire this lock wait in a loop while checking if the lock is available or not. This is known as busy waiting because the process is not doing any useful operation even though it is active.

**Approaches to Interprocess Communication**

The different approaches to implement interprocess communication are given as

* **Pipe**

A pipe is a data channel that is unidirectional. Two pipes can be used to create a two-way data channel between two processes. This uses standard input and output methods. Pipes are used in all POSIX systems as well as Windows operating systems.

* **Socket**

The socket is the endpoint for sending or receiving data in a network. This is true for data sent between processes on the same computer or data sent between different computers on the same network. Most of the operating systems use sockets for interprocess communication.

* **File**

A file is a data record that may be stored on a disk or acquired on demand by a file server. Multiple processes can access a file as required. All operating systems use files for data storage.

* **Signal**

Signals are useful in interprocess communication in a limited way. They are system messages that are sent from one process to another. Normally, signals are not used to transfer data but are used for remote commands between processes.

* **Shared Memory**

Shared memory is the memory that can be simultaneously accessed by multiple processes. This is done so that the processes can communicate with each other. All POSIX systems, as well as Windows operating systems use shared memory.

* **Message Queue**

Multiple processes can read and write data to the message queue without being connected to each other. Messages are stored in the queue until their recipient retrieves them. Message queues are quite useful for interprocess communication and are used by most operating systems.

Q-3 Write short note on Hardware Architecture for Parallel Processing.

A-3 Parallel processing has been developed as an effective technology in modern computers to meet the demand for higher performance, lower cost and accurate results in real-life applications. Concurrent events are common in today’s computers due to the practice of multiprogramming, multiprocessing, or multicomputing.

Modern computers have powerful and extensive software packages. To analyze the development of the performance of computers, first we have to understand the basic development of hardware and software.

categorize parallel processing hardware implementations according to the particular resources that are shared. This chapter describes these categories:

* Shared memory systems
* Shared disk systems
* Shared nothing systems

These implementations can also be described as "tightly coupled" or "loosely coupled", according to the way the nodes communicate.

Oracle supports  these implementations of parallel processing, assuming that in a shared nothing system the software enables a node to access a disk from another node. For example, the IBM SP2 features a virtual shared disk: the disk is shared through software.

**Shared Memory Systems**

Tightly coupled shared memory systems, illustrated have the following characteristics:

* Multiple CPUs share memory
* Each CPU has full access to all shared memory through a common bus
* Communication among nodes occurs by way of shared memory
* Performance is limited by memory bus bandwidth.

Symmetric multiprocessor (SMP) machines are often comprised of nodes in a cluster. You can install multiple SMP nodes with OPS in a tightly coupled system where memory is shared among the multiple CPUs, and is accessible by all the CPUs through a memory bus. Examples of tightly coupled systems include the Pyramid, Sequent, and Sun SparcServer.

It does not make sense to run OPS on a single SMP machine, because the system would incur a great deal of unnecessary overhead from IDLM accesses.

Performance is potentially limited in a tightly coupled system by a number of factors. These include various system components such as the memory bandwidth, CPU-to-CPU communication bandwidth, the memory available on the system, the I/O bandwidth, and the common bus bandwidth.

**Shared Disk Systems**

Shared disk systems are typically loosely coupled. Such systems, illustrated have the following characteristics:

* Each node consists of one or more CPUs and associated memory
* Memory is not shared among nodes
* Communication occurs over a common high-speed bus
* Each node has access to the same disks and other resources
* A node can be an SMP if the hardware supports i

Q-4 Explain in brief about hardware Virtulization techniques.

A-4 Virtualization means creating a virtual platform of something, which will include virtual computer hardware, virtual storage devices, and virtual computer network.

In *hardware virtualization*, software called hypervisor is used. With the help of hypervisor virtual machine, software embedded into the hardware component of the server. The work of hypervisor is that it manages the physical hardware resource which is shared between the customer and the provider

 There are several extensions in the processes, which help to accelerate virtualization activities and boost the performance of hypervisors. If this virtualization is done for server platform it is known as server socialization.

hypervisor creates an abstraction layer between the software and the hardware in use. After the installation of a hypervisor, virtual representations take place such as virtual processors.

We cannot use physical processors after installation. There are several popular hypervisors such as VMware’s vSphere, based on ESXi, and Microsoft’s Hyper-V.

In this system, multiple VMs can host at a time, but every VM logically isolated from each other. This is because of the security reasons. **One of the security reasons** is a Malware attack or the crash of VM.

Because of this, the other VMs will not get affected. If multiple VMs use, the efficiency of the system will increase simultaneously and the overall performance will be better.

So, this will leads to the fact that this improved heartbeat utilization provides **various benefits** and supports system while reducing the number of servers which will save money.

## **Types of Hardware Virtualization**

This is the list of hardware virtualization in Cloud Computing:

* Full Virtualization
* Emulation Virtualization
* Para-Virtualization

**Full Virtualization**

In full virtualization, there is no need for any modification to run any application. In addition, the hardware **architecture** completely simulates, which benefits the guest software. There is an environment, quite similar to an operating system in a server.

### **Emulation Virtualization**

In emulation virtualization, hardware simulates by the virtual machine and it is independent. Here, the guest operating system does not require any other modification. In this virtualizations, computer hardware as architectural support builds and manages a fully virtualized VM.

### **Para-virtualization**

In paravirtualization, the hardware not simulates and the guest software runs its isolated system. It is not necessary to simulate the hardware, but it utilizes an API which modifies the guest operating system.

Q-5Explain distributed object frameworks with example

A- A distributed application using remote objects is executed as a set of processes located on the nodes of a network. An object's method is executed by a process or a thread (in some models, objects may also be shared between processes), and may include calls to other objects' methods. For such inter-object method calls, three situations may occur .

* The calling and called objects are in the same process this is a local invocation.
* The calling and called objects are executed by different processes on the same site this is an out-of-process invocation.
* The calling and called objects are on different nodes this is a remote invocation.

Local invocations are done like in a non-distributed object system. Non-local forms of invocation rely on an object request broker (ORB), a middleware that supports distributed objects. This term has been introduced for the CORBA architecture, but it applies to other object systems as well. An ORB has the following functions.

* Identifying and locating objects.
* Binding client to server objects.
* Performing method calls on objects.
* Managing objects' life cycle (creating, activating, deleting objects)

In the rest of this section, we give a first outline of the operation of a non-local invocation. More details on the internals of an ORB are given in Sections

An application using remote objects is typically organized using the client-server model: a process or thread executing a method of a client object sends a request to a (possibly remote, or out-of-process) server object in order to execute a method of that object.

The overall organization of a method invocation on a remote object, shown on is similar to that of an RPC, as described in It relies on a stub-skeleton pair. In contrast with RPC, the stub and the skeleton are objects in their own right. Take the example of the call from object .