CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | |

Internal Assessment Test 4 – Feb. 2022

Sub:	**CLOUD COMPUTING**							**Sub Code:**	**20MCA342**
Date:	**02-02-2022**	**Duration:**	**90 min's**	**Max Marks:**	**50**	**Sem:**	**III**	**Branch:**	**MCA**

Note : Answer TWO full questions from part I and III each and ONE full question from part II

	Marks	OBE CO	OBE RBT
PART-I (Answer any TWO)			
1. Compare the characteristics of parallel and distributed system. Draw and explain the layered view of distributed system.	[10]	CO2	L2
2. What are the various hardware architectures for parallel processing? Discuss each type and draw appropriate diagrams	[10]	CO2	L2
3. Explain the software architectural styles.	[10]	CO2	L2

PART-II

4. Explain in detail about Remote Procedure Call (RPC) with necessary diagram.	[10]	CO2	L2

(OR)

5. Explain the process of live migration of VM from one host to another.	[10]	CO3	L2

PART-III (Answer any TWO)

6. Explain the following with necessary diagrams: Full virtualisation, para virtualisation and partial virtualisation.	[10]	CO3	L2
7. Explain in detail about the virtualization support at OS level with example.	[10]	CO3	L2
8. Explain Xen architecture with suitable diagram.	[10]	CO3	L2

Answers:

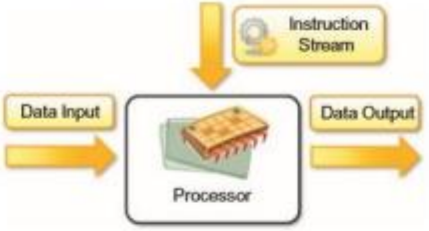
1.

S.NO	Parallel Computing	Distributed Computing
1.	Many operations are performed simultaneously	System components are located at different locations
2.	Single computer is required	Uses multiple computers
3.	Multiple processors perform multiple operations	Multiple computers perform multiple operations
4.	It may have shared or distributed memory	It have only distributed memory
5.	Processors communicate with each other through bus	Computer communicate with each other through message passing.
6.	Improves the system performance	Improves system scalability, fault tolerance and resource sharing capabilities

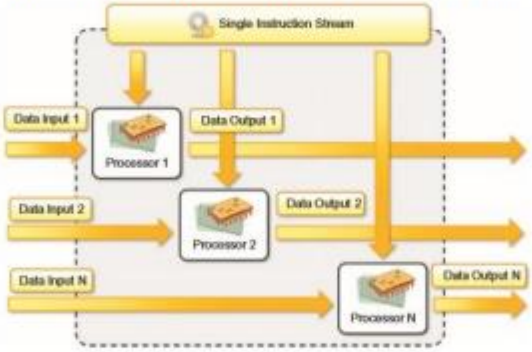
2. The hardware architecture of parallel computing is disturbed along the following categories as given below :

1. Single-instruction, single-data (SISD) systems
2. Single-instruction, multiple-data (SIMD) systems
3. Multiple-instruction, single-data (MISD) systems
4. Multiple-instruction, multiple-data (MIMD) systems

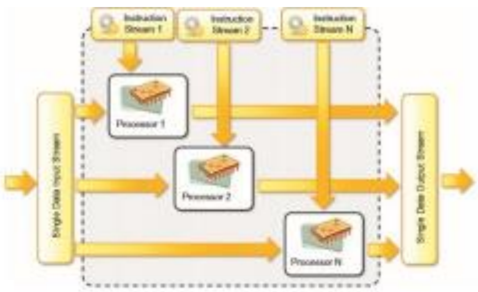
• Single-instruction, single-data (SISD) systems - An SISD computing system is a uniprocessor machine capable of executing a single instruction,
which operates on a single data stream (see Figure 2.2). In SISD, machine instructions are processed sequentially; hence computers adopting this model are popularly called sequential computers. Most conventional computers are built using the SISD model. All the instructions and data to be processed have to be stored in primary memory. The speed of the processing element in the SISD model is limited by the rate at which the computer can transfer information internally. Dominant representative SISD systems are IBM PC, Macintosh, and workstations



• Single-instruction, multiple-data (SIMD) systems - An SIMD computing system is a multiprocessor machine capable of executing the same instruction on all the CPUs but operating on different data streams (see Figure 2.3). Machines based on an SIMD model are well suited to scientific computing since they involve lots of vector and matrix operations. For instance, statements such as
Ci = Ai * Bi
can be passed to all the processing elements (PEs); organized data elements of vectors A and B can be divided into multiple sets (N-sets for N PE systems); and each PE can process one data set. Dominant representative SIMD systems are Cray's vector processing machine and Thinking Machines' cm.

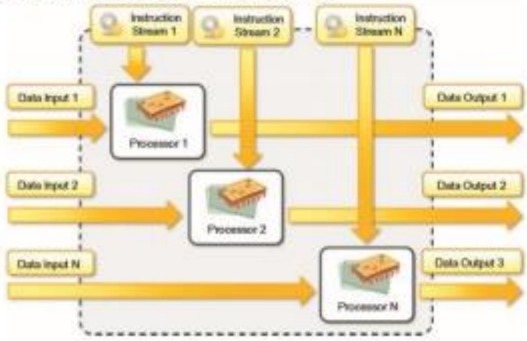


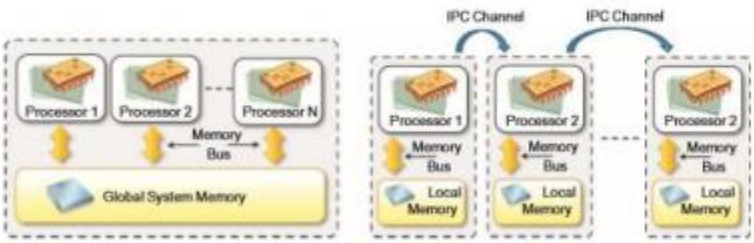
• Multiple-instruction, single-data (MISD) systems - An MISD computing system is a multiprocessor machine capable of executing different instructions on different PEs but all of them operating on the same data set (see Figure 2.4). For instance, statements such as perform different operations on the same data set. Machines built using the MISD model are not useful in most of the applications; a few machines are built, but none of them are available commercially. They became more of an intellectual exercise than a practical configuration



• Multiple-instruction, multiple-data (MIMD) systems - An MIMD computing system is a multiprocessor machine capable of executing multiple instructions on multiple data sets (see Figure 2.5). Each PE in the MIMD model has separate instruction and data streams; hence machines built using this model are well suited to any kind of application. Unlike SIMD and MISD machines, PEs in MIMD machines work asynchronously. MIMD machines are broadly categorized into shared-memory MIMD and distributed-memory MIMD based on the way PEs are coupled to the main memory.

Shared memory MIMD machines





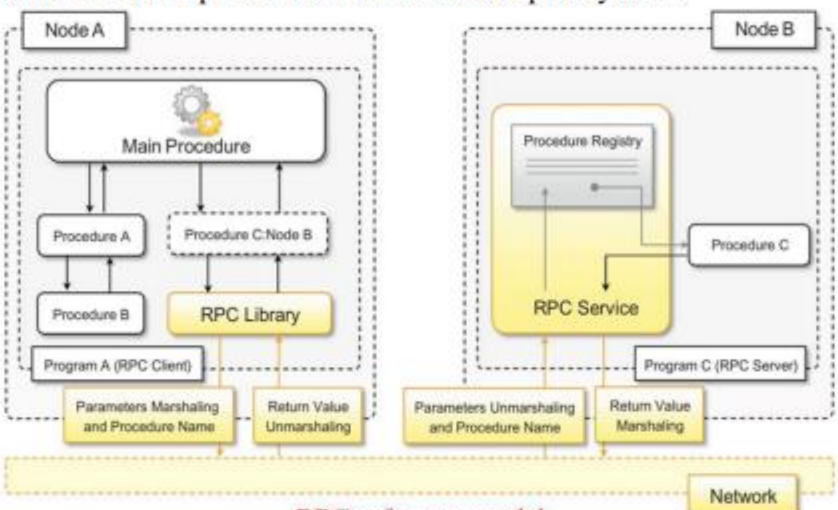
In the shared memory MIMD model, all the PEs are connected to a single global memory and they all have access to it (see Figure 2.6). Systems based on this model are also called tightly coupled multiprocessor systems. The communication between PEs in this model takes place through the shared memory; modification of the data stored in the global memory by one PE is visible to all other PEs. Dominant representative shared memory MIMD systems are Silicon Graphics machines and Sun/IBM's SMP (Symmetric Multi-Processing). Distributed memory MIMD machines In the distributed memory MIMD model, all PEs have a local memory. Systems based on this model are also called loosely coupled multiprocessor systems. The communication between PEs in this model takes place through the interconnection network (the interprocess communication channel, or IPC). The network connecting PEs can be configured to tree, mesh, cube, and so on. Each PE operates asynchronously, and if communication/synchronization among tasks is necessary, they can do so by exchanging messages between them. The shared-memory MIMD architecture is easier to program but is less tolerant to failures and harder to extend with respect to the distributed memory MIMD model. Failures in a shared-memory MIMD affect the entire system, whereas this is not the case of the distributed model, in which each of the PEs can be easily isolated. Moreover, shared memory MIMD architectures are less likely to scale because the addition of more PEs leads to memory contention. This is a situation that does not happen in the case of distributed memory, in which each PE has its own memory. As a result, distributed memory MIMD architectures are most popular today

3. Explain the software architectural styles.

Category	Most Common Architectural Styles
Data-centered	Repository Blackboard
Data flow	Pipe and filter Batch sequential
Virtual machine	Rule-based system Interpreter
Call and return	Main program and subroutine call/top-down systems Object-oriented systems Layered systems
Independent components	Communicating processes Event systems

4. Explain in detail about Remote Procedure Call (RPC) with necessary diagram.

- Remote procedure call
- fundamental abstraction enabling the execution of procedures on client's request.
- allows extending the concept of a procedure call beyond the boundaries of a process and a single memory address space
- The called procedure and calling procedure may be on the same system or they may be on different systems in a network
- Even though it is a quite old technology, RPC is still used today as a fundamental component for IPC in more complex systems.



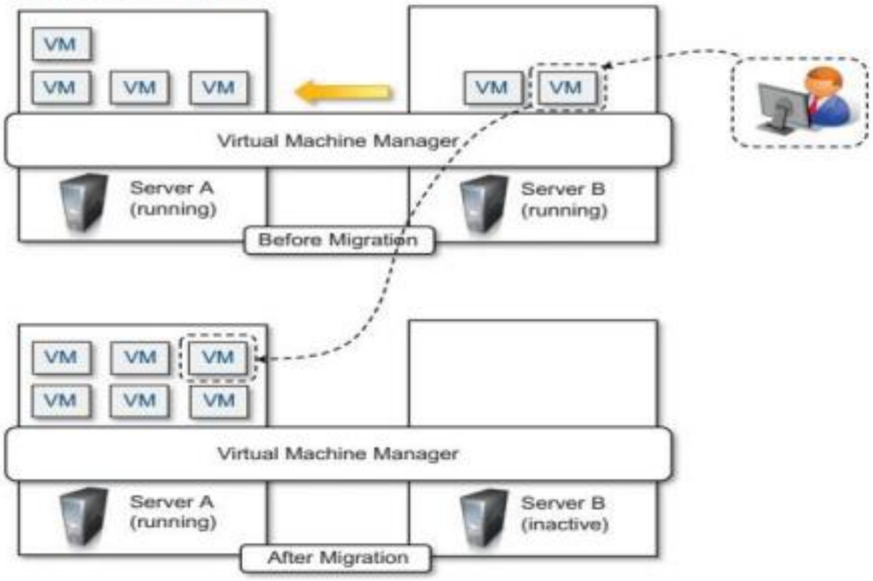
RPC reference model

- Marshaling: identifies the process of converting parameter and return values into a form that is more suitable to be transported over a network through a sequence of bytes. [packing of arguments(or parameters) into a message packet]

- o Unmarshaling: the opposite procedure. [unpacking of arguments(or parameters) received from the call packet].
- The RPC runtime is also for handling the request-reply interaction that happens between the client and the server process in a completely transparent manner.
- RPC for IPC consists of the following steps:
  - o Design and implementation of the server procedures that will be exposed for remote invocation.
  - o Registration of remote procedures with the RPC server on the node where they will be made available.
  - o Design and implementation of the client code that invokes the remote procedure(s).

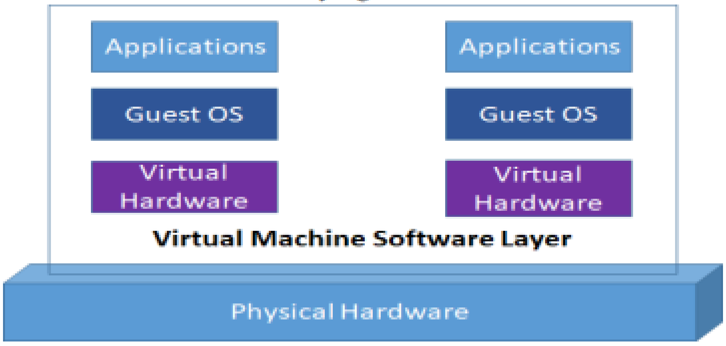
5. Explain the process of live migration of VM from one host to another.

- o Server Consolidation: allows reducing the number of active resources by aggregating virtual machines over a smaller number of resources that become fully utilized
- o Virtual Machine Migration: the movement of virtual machine instances
- o Live Migration: the process of moving a running virtual machine or application between different physical machines without disconnecting the client or application

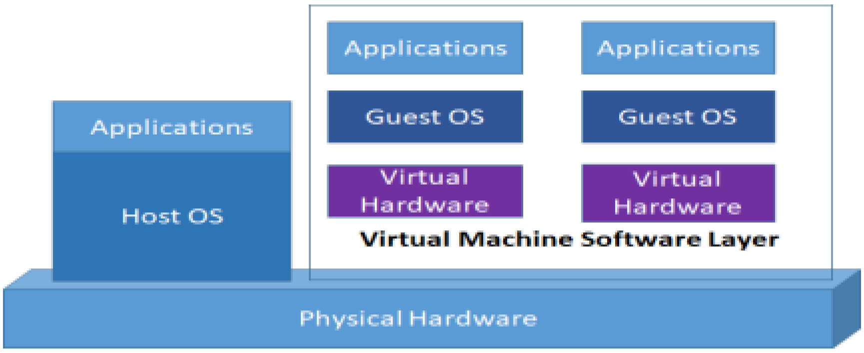


6. Explain the following with necessary diagrams: Full virtualisation, para virtualisation and partial virtualisation.

- Full virtualization
- the ability to run a program, most likely an operating system, directly on top of a virtual machine and without any modification, as though it were run on the raw hardware.
- Virtual Machine Managers (VMM) are required to provide a complete emulation of the entire underlying hardware



- Para-Virtualisation
- expose a software interface to the virtual machine that is slightly modified from the host and, as a consequence, guest OS needs to be modified.
- Aim: to provide the capability to demand the execution of performance-critical operations directly on the host, thus preventing performance losses that would otherwise be experienced in managed execution.
- Example: VMWare, Parallels
- Embedded and real-time environments: TRANGO, WindRiver, and XtratuM



- Partial virtualization
- Many different operating systems can run simultaneously
- provides a partial emulation of the underlying hardware, thus not allowing the complete execution of the guest operating system in complete isolation.
- allows many applications to run transparently, but not all the features of the operating system can be supported, as happens with full virtualization.
- Example: IBM M44/44X: Partial virtualisation as realized for full virtualisation
  - Address space virtualization is a common feature of contemporary operating systems.

7. Explain in detail about the virtualization support at OS level with example.

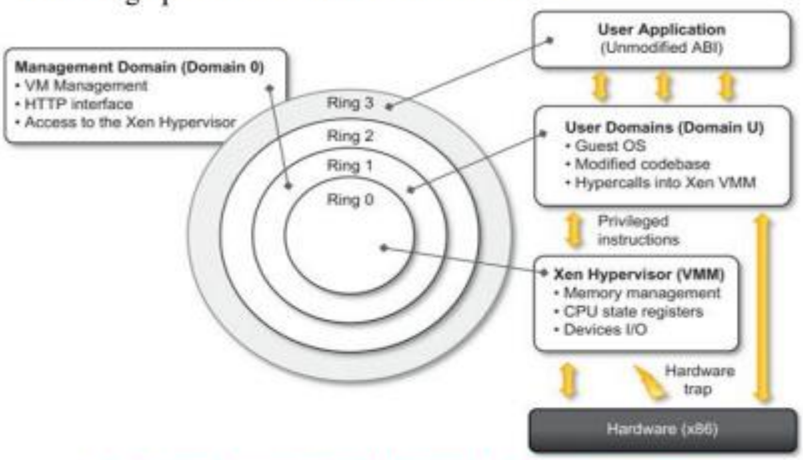
- offers the opportunity to create different and separated execution environments for applications that are managed concurrently.
- Differently from hardware virtualization, there is no virtual machine manager or hypervisor
- the virtualization is done within a single operating system, where the OS kernel allows for multiple isolated user space instances
- A user space instance in general contains a proper view of the file system, which is completely isolated, and separate IP addresses, software configurations, and access to devices
- It is an efficient solution for server consolidation scenarios

Example: *chmod* in unix (Solaris and OpenVZ)

```
[test_dev@ra-net ~]$ ls test2
alias.sh  cat.sh       loop.sh      special.sh  variable.sh
array.sh  first2.sh    sayhellow.sh test.txt
awk.sh    function.sh  sed.sh       unix.sh
[test_dev@ra-net ~]$
[test_dev@ra-net ~]$
[test_dev@ra-net ~]$ chmod 755 test2/test.txt
```

8. Explain Xen architecture with suitable diagram.

- an open-source initiative implementing a virtualization platform based on para-virtualization, under GPL2
- Xen now has a large open-source community backing it
- It is used for either desktop virtualization or server virtualization
- Virtualisation system supporting both para-virtualisation and hardware-assistant full virtualisation
  - Para-virtualisation: High performance, high scalability and uses a modified OS
  - Hardware-assisted full virtualisation: enhance virtualisation in x86, uses an unmodified OS
- Offers high performance and secure architecture



Xen architecture and guest OS management

- Xen hypervisor: runs in the highest privileged mode and controls the access of guest operating system to the underlying hardware

- Guest OS are executed within domains, which represent virtual machine instances
- All guest OSs are maintained and executed in a special domain called Domain 0
- HTTP server serves requests for virtual machine creation
- Xen implemented virtualization by executing the hypervisor in Ring 0, Domain 0