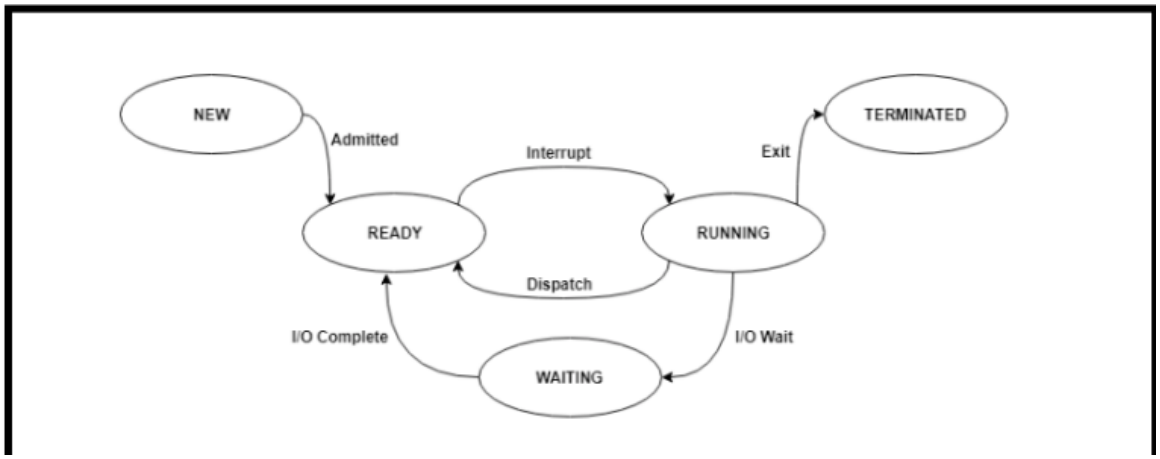
Answer in Brief

1. **What is Process Synchronization with its four conditions in details**

A process is an active program i.e a program that is under execution. It is more than the program code as it includes the program counter, process stack, registers, program code etc.

Process States



New - The process is in the new state when it has just been created.

• Ready - The process is waiting to be assigned the processor by the shortterm scheduler.

• Running - The process instructions are being executed by the processor.

• Waiting - The process is waiting for some event such as I/O to occur.

• Terminated - The process has completed its execution.

2. **Draw and Explain Distributed Architecture of file System**

ARCHITECTURE OF DISTRIBUTED FILE SYSTEM

- File servers and File clients interconnected by a communication network.

- **Two most important components:**
  1. **Name Server**: map logical names to stored object's (files, directories) physical location.
  2. **Cache Manager**: perform file caching. Can present on both servers and clients.
     1. Cache on the client deals with network latency
     2. Cache on the server deals with disk latency

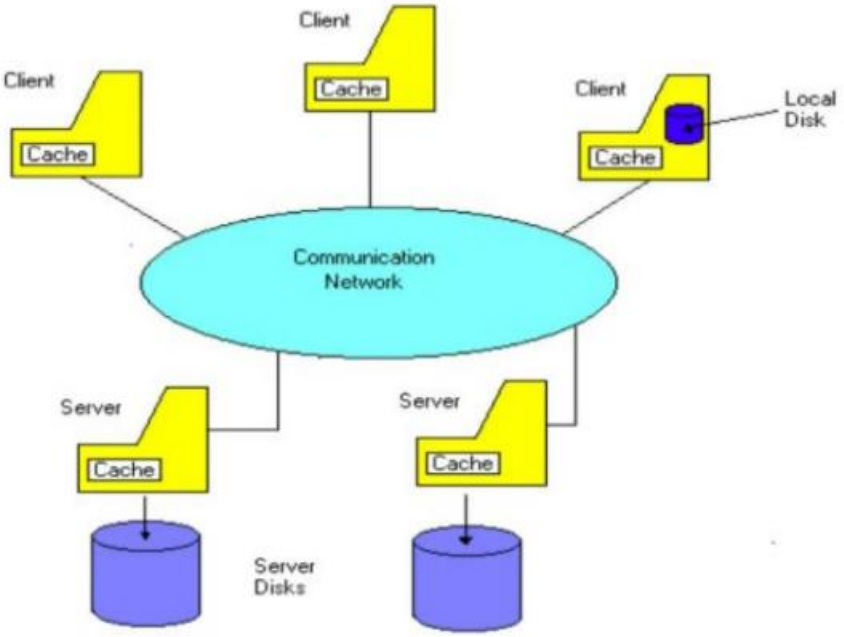


Figure:
Architecture of a Distributed File System

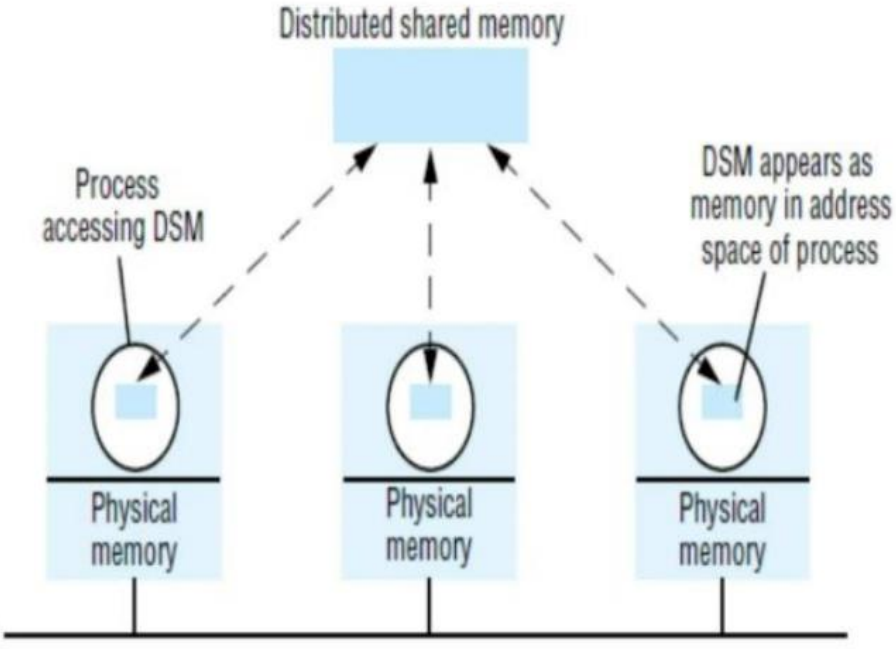
STEPS TO ACCESS DATA

1. check client cache, if present, return data.
2. Check local disk, if present, load into local cache, return data.
3. Send request to file server
4. ...
5. server checks cache, if present, load into client cache, return data
6. disk read
7. load into server cache
8. load into client cache
9. return data

3. Explain Shared Memory with implementation of Algorithm

DISTRIBUTED SHARED MEMORY

- Idea of distributed shared memory is to provide an environment where computers support a shared address space that is made by physically dispersed memories.
- It refers to shared memory paradigm applied to loosely coupled distributed memory systems. **It gives the systems illusion of physically shared memory.**
- Memory mapping manager is responsible for mapping between local memories and the shared memory address space.
- Any processor can access any memory location in the address space directly.
- Chief responsibility is to keep the address space coherent at the times.



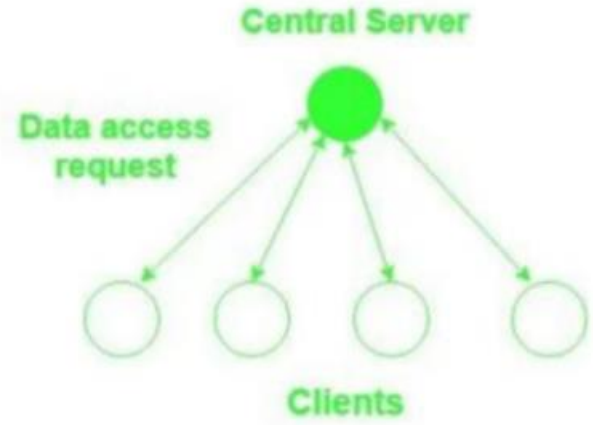
- Each node of the system consist of one or more CPUs and memory unit.
- Nodes are connected by high speed communication network.
- Simple message passing system for nodes to exchange information.
- Main memory of individual nodes is used to cache pieces of shared memory space.
- **Shared memory exist only virtually.**
- Memory mapping manager routine maps local memory to shared virtual memory.

Algorithm for implementation of DSM

1. The Central Server Algorithm
2. The Migration Algorithm
3. The Read-Replication Algorithm
4. The Full–Replication Algorithm

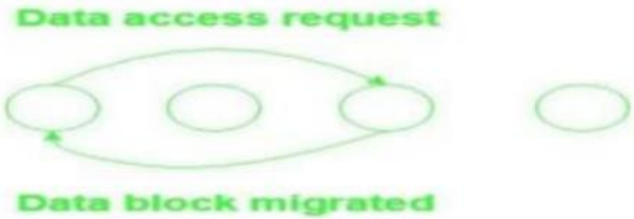
1. The Central Server Algorithm

- **Central server maintains all shared data:**
  - Read request: returns data item.
  - Write request: updates data and returns acknowledgement message.



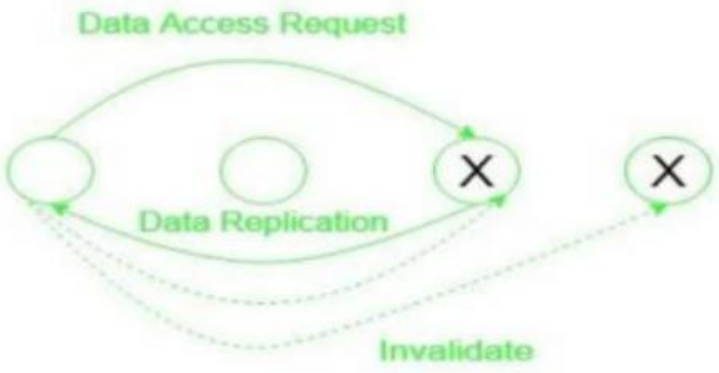
2. The Migration Algorithm

- Every data access request is forwarded to location of data while in this data is shipped to location of data access request which allows subsequent access to be performed locally.
- It allows only one node to access a shared data at a time.
- The whole block containing data item migrates instead of individual item requested.
- This algorithm provides an opportunity to integrate DSM with virtual memory provided by operating system at individual nodes.



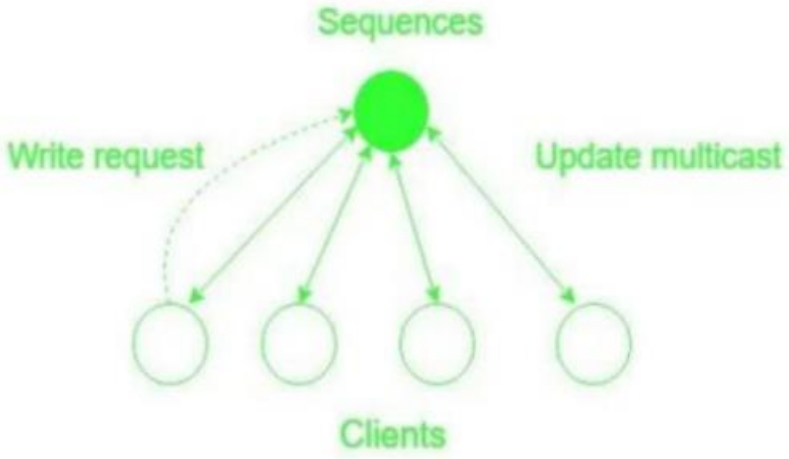
3. The Read-Replication Algorithm

- This extends the migration algorithm by replicating data blocks to multiple nodes and allowing multiple nodes to have read access or one node to have both read write access.
- After a write, all copies are invalidated or updated.
- DSM has to keep track of locations of all copies of data objects.
- Advantage:
  - The read-replication can lead to substantial performance improvements if the ratio of reads to writes is large.
  - It improves system performance by allowing multiple nodes to access data concurrently.



4. The Full–Replication Algorithm

- It is an extension of read replication algorithm which allows multiple nodes to have both read and write access to shared data blocks.
- Issue: Since many nodes can write shared data concurrently, the access to shared data must be controlled to maintain it's consistency.



4. Explain Deadlocks with its Strategy in details

Introduction to Deadlock

Every process needs some resources to complete its execution. However, the resource is granted in a sequential order.

1. The process requests for some resource.
2. OS grant the resource if it is available otherwise let the process waits.
3. The process uses it and release on the completion.

A Deadlock is a situation where each of the computer process waits for a resource which is being assigned to some another process.

In this situation, none of the process gets executed since the resource it needs, is held by some other process which is also waiting for some other resource to be released.

Strategies for handling Deadlock

Deadlock Ignorance

In this approach, the Operating system assumes that deadlock never occurs. It simply ignores deadlock. This approach is best suitable for a single end user system where User uses the system only for browsing and all other normal stuff.

In these types of systems, the user has to simply restart the computer in the case of deadlock. Windows and Linux are mainly using this approach.

Deadlock prevention

Deadlock happens only when Mutual Exclusion, hold and wait, No preemption and circular wait holds simultaneously. If it is possible to violate one of the four conditions at any time then the deadlock can never occur in the system.

Deadlock avoidance

the operating system checks whether the system is in safe state or in unsafe state at every step which the operating system performs.

The process continues until the system is in safe state. Once the system moves to unsafe state, the OS has to backtrack one step.

The OS reviews each allocation so that the allocation doesn't cause the deadlock in the system.

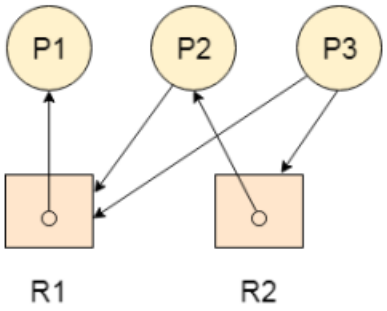
the process should declare the maximum number of resources of each type it may ever need. The Deadlock avoidance algorithm examines the resource allocations so that there can never be a circular wait condition.

Example

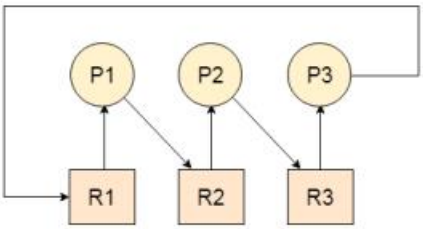
3 processes P1, P2 and P3, and two types of resources R1 and R2. The resources are having 1 instance each.

According to the graph, R1 is being used by P1, P2 is holding R2 and waiting for R1, P3 is waiting for R1 as well as R2.

The graph is deadlock free since no cycle is being formed in the graph.



5. Explain necessary conditions for Deadlock with diagram



Necessary conditions for Deadlocks

1. **Mutual Exclusion**

   A resource can only be shared in mutually exclusive manner. It implies, if two process cannot use the same resource at the same time.

2. **Hold and Wait**

   A process waits for some resources while holding another resource at the same time.

3. **No preemption**

   The process which once scheduled will be executed till the completion. No other process can be scheduled by the scheduler meanwhile.

4. **Circular Wait**

   All the processes must be waiting for the resources in a cyclic manner so that the last process is waiting for the resource which is being held by the first process.

6. **Explain Distributed Operating system with its types**

Distributed Operating Systems

Distributed systems require a network to connect all of its elements (devices, hardware, or software) so that they may exchange messages and interact.

A distributed system's ability to consistently convey messages, whether they're delivered, received, acknowledged, or how a node retries after a failure, is a key aspect.

Services and applications required to scale, and additional computers necessary to be added and managed, therefore distributed systems were born.

It connects multiple computers via a single communication channel. Furthermore, each of these systems has its own processor and memory.

Types of Distributed Operating System

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

Client-Server System

This type of system requires the client to request a resource, after which the server gives the requested resource. When a client connects to a server, the server may serve multiple clients at the same time.

Client-Server Systems are also referred to as "Tightly Coupled Operating Systems".

Client-Server Systems function as a centralized server since they approve all requests issued by client systems.

Server systems can be divided into two parts:

1. **Computer Server System**

This system allows the interface, and the client then sends its own requests to be executed as an action. After completing the activity, it sends a back response and transfers the result to the client.

2. **File Server System**

It provides a file system interface for clients, allowing them to execute actions like file creation, updating, deletion, and more.

Peer-to-Peer System

The nodes play an important role in this system. The task is evenly distributed among the nodes. Additionally, these nodes can share data and resources as needed. Once again, they require a network to connect.

The Peer-to-Peer System is known as a "Loosely Couple System".

This concept is used in computer network applications since they contain a large number of processors that do not share memory or clocks.

Middleware

Middleware enables the interoperability of all applications running on different operating systems. Those programs are capable of transferring all data to one other by using these services.

Three-tier

The information about the client is saved in the intermediate tier rather than in the client, which simplifies development. This type of architecture is most commonly used in online applications.

N-tier

When a server or application has to transmit requests to other enterprise services on the network, n-tier systems are used.

7. Explain Strategy of Handling Deadlock with example

Strategies for handling Deadlock

Deadlock Ignorance

In this approach, the Operating system assumes that deadlock never occurs. It simply ignores deadlock. This approach is best suitable for a single end user system where User uses the system only for browsing and all other normal stuff.

In these types of systems, the user has to simply restart the computer in the case of deadlock. Windows and Linux are mainly using this approach.

Deadlock prevention

Deadlock happens only when Mutual Exclusion, hold and wait, No preemption and circular wait holds simultaneously. If it is possible to violate one of the four conditions at any time then the deadlock can never occur in the system.

Deadlock avoidance

the operating system checks whether the system is in safe state or in unsafe state at every step which the operating system performs.

The process continues until the system is in safe state. Once the system moves to unsafe state, the OS has to backtrack one step.

The OS reviews each allocation so that the allocation doesn't cause the deadlock in the system.

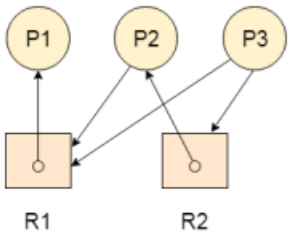
the process should declare the maximum number of resources of each type it may ever need. The Deadlock avoidance algorithm examines the resource allocations so that there can never be a circular wait condition.

Example

3 processes P1, P2 and P3, and two types of resources R1 and R2. The resources are having 1 instance each.

According to the graph, R1 is being used by P1, P2 is holding R2 and waiting for R1, P3 is waiting for R1 as well as R2.

The graph is deadlock free since no cycle is being formed in the graph.



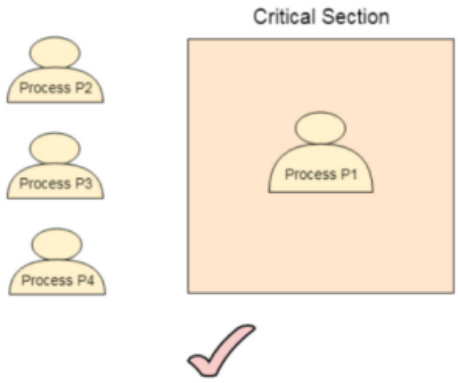
8. Explain Distributed shared Memory in Brief

Answer in One Sentence

1. What is Mutual Exclusion

1. Mutual Exclusion

Our solution must provide mutual exclusion. By Mutual Exclusion, we mean that if one process is executing inside critical section then the other process must not enter in the critical section.



2. **Define Process States**

   Define Process: A process is an active program i.e a program that is under execution. It is more than the program code as it includes the program counter, process stack, registers, program code etc.

3. **What is Co-operative process**

   Cooperating processes Execution of one process affects the execution of the other. Thus, it is necessary that these processes are synchronized in order to guarantee the order of execution.

4. **What is independent process**

   Independent process is the process that can not affect or be affected by the other processes. Independent processes does not share any data like temporary or persistent with any other process.

5. **What is race condition in operating system**

   Race Condition in OS

   When more than one processes execute the same code or access the same memory/shared variable, it is possible that the output or value of the shared variable is wrong.

   In this condition, all processes race ahead in order to prove that their output is correct. This situation is known as race condition.

   When multiple processes access and manipulate the same data concurrently the outcome depends on the order in which these processes accessed the shared data. When the output of multiple thread execution differs according to the order in which the threads execute, a race condition occurs.

6. **What is deadlock in operating System**

A Deadlock is a situation where each of the computer process waits for a resource which is being assigned to some another process. In this situation, none of the process gets executed since the resource it needs, is held by some other process which is also waiting for some other resource to be released.

7. **What is Distributed File System**

Distributed File Systems

- File System work as the resource management component, which manages the availability of files in distributed system.
- A common file system that can be shared by all the autonomous computers in the system. i.e. files can be stored at any machine and the computation can be performed at any machine.

8. **What are the two types of Processes**

Processes are two types based on their types of categories.

1. **Independent process.**
2. **Cooperating process.**

Independent process is the process that can not affect or be affected by the other processes. Independent processes **does not share any data** like temporary or persistent with any other process.

Cooperating processes Execution of one process affects the execution of the other. Thus, it is necessary that these processes are synchronized in order to guarantee the order of execution.

9. **What is critical section in operating system**

The Critical Section Problem

The critical section problem is used to design a set of protocols which can ensure that the Race condition among the processes will never arise.

Critical Section is the part of a program which tries to access shared resources. That resource may be any resource in a computer like a memory location, Data structure, CPU or any IO device.

10. **What is bounded waiting condition in synchronization**

3. Bounded Waiting

We should be able to predict the waiting time for every process to get into the critical section. The process must not be endlessly waiting for getting into the critical section.

MCQs

1. **When several processes access the same data concurrently and the outcome of the execution depend on the particular order in which the access take place is called _____**
2. **If a process is executing in it critical section then no other processes can be executing in critical section what is process called _____**
3. **Which process can be affected by the other processes executing in the system is called _____**
4. **Mutual Exclusion can be provided by the _____**
5. **Process Synchronization can be done on _____**
6. **Client- Server system is also known as _____**
7. **The Peer-to- Peer system is known as _____**
8. **Agreement is always required to achieve a _____ goal in distributed System**
9. **Memory Mapping manager is responsible for mapping between Local Memory and Shared Memory Address Space**