★ History of Operating Systems

Generations

- First Generation (1945 - 1955) - Vacuum Tubes and Plugboards.
- Second Generation (1955 - 1965) - Transistors and Batch Systems.
- Third Generation (1965 - 1980) - Integrated Circuits and Multiprogramming.
- Fourth Generation (1980 - Present) - Personal Computers.
- The Fifth Generation (1990–Present) - Mobile Computers, Tablets, Smart Phones.

★ What is an Operating System ?

An operating system acts as an intermediary between the user of a computer and the computer hardware.

The purpose of an operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner

★ Operating systems use two components to manage computer programs and applications:

1. The kernel is the core inner component that processes data at the hardware level.

It handles input-output management, memory and process management.

2. The shell is the outer layer that manages the interaction between the user and the OS.

The shell communicates with the operating system by either taking the input from the user or a shell script.

A shell script is a sequence of system commands that are stored in a file.

★ Types of Operating Systems

1. Batch OS
2. Time-sharing or multitasking OS
3. Distributed OS
4. Network OS
5. Real-time OS
6. Mobile OS

1. Batch OS

The batch operating system does not have a direct link with the computer. A different system divides and allocates similar tasks into batches for easy processing and faster response.

The batch operating system is appropriate for lengthy and time-consuming tasks. To avoid slowing down a device, each user prepares their tasks offline and submits them to an operator.

Ex - managing payroll systems, data entry and bank statements

Advantages	Disadvantages
Many users can share batch systems. There is little idle time for batch operating systems.	Some notable disadvantages are: Batch operating systems are challenging to debug.
It becomes possible to manage large workloads.	Any failure of the system creates a backlog.
It's easy to estimate how long a task will take to be completed.	It may be costly to install and maintain good batch operating systems.

2. Time-sharing or multitasking OS

The time-sharing operating system, also known as a multitasking OS, works by allocating time to a particular task and switching between tasks frequently.

Unlike the batch system, the time- sharing system allows users to complete their work in the system simultaneously.

It allows many users to be distributed across various terminals to minimize response time

Advantages	Disadvantages
There's a quick response during task performance.	The user's data security might be a problem.
It minimizes the idle time of the processor.	System failure can lead to widespread failures.
All tasks get an equal chance of being accomplished.	Problems in data communication may arise.
It reduces the chance of software duplication.	The integrity of user programs is not assured.

3. Distributed OS

This system is based on autonomous but interconnected computers communicating with each other via communication lines or a shared network. Each autonomous system has its own processor that may differ in size and function.

A distributed operating system serves multiple applications and multiple users in real time. The data processing function is then distributed across the processors.

Ex - telecommunication networks, airline reservation controls and peer-to-peer networks.

Advantages	Disadvantages
They allow remote working.	If the primary network fails, the entire system shuts down.
They allow a faster exchange of data among users.	They're expensive to install.
Failure in one site may not cause much disruption to the system.	They require a high level of expertise to maintain.
They reduce delays in data processing.	
They minimize the load on the host computer.	
They enhance scalability since more systems can be added to the network.	

4. Network OS

Network operating systems are installed on a server providing users with the capability to manage data, user groups and applications.

This operating system enables users to access and share files and devices such as printers, security software and other applications, mostly in a local area network.

Ex - Microsoft Windows, Linux and macOS X

Advantages	Disadvantages
Centralized servers provide high stability.	They require regular updates and maintenance.
Security issues are easier to handle through the servers.	Servers are expensive to buy and maintain.
It's easy to upgrade and integrate new technologies.	Users' reliance on a central server might be detrimental to workflows.
Remote access to the servers is possible.	

5. Real-time OS

The response time between input, processing and response is tiny, which is beneficial for processes that are highly sensitive and need high precision.where delays may lead to loss of life and property

Ex - operating missile systems, medical systems or air traffic control systems, scientific experiments, medical imaging, robotics and air traffic control operations.

Advantages	Disadvantages
They use device and systems maximally, hence more output.	They have a low capacity to run tasks simultaneously.
They allow fast shifting from one task to another.	They use heavy system resources.
The focus is on current tasks, and less focus is put on the queue.	They run on complex algorithms that are not easy to understand.
They can be used in embedded systems.	They're unsuitable for thread priority because of the system's inability to switch tasks.
Real-time systems are meticulously programmed, hence free of errors.	
They allow easy allocation of memory.	

6. Mobile OS

Mobile operating systems run exclusively on small devices such as smartphones, tablets and wearables.

Ex - Android OS, Apple and Windows mobile OS.

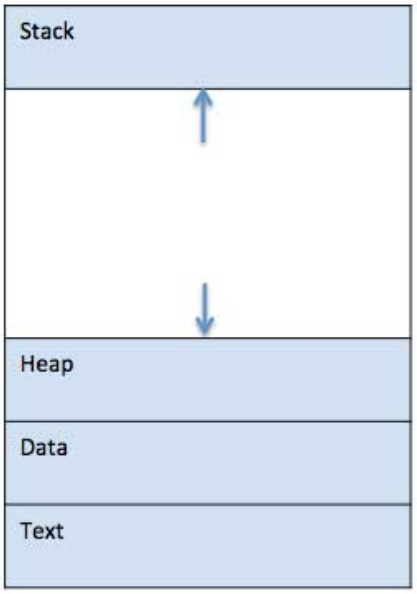
Advantages	Disadvantages
Most systems are easy for users to learn and operate.	Some mobile OS put a heavy drain on a device's battery, requiring frequent recharging.
	Some systems are not user-friendly.

★ What is the Process ?

A process is the instance of a computer program that is being executed by one or many threads.

A process is defined as an entity which represents the basic unit of work to be implemented in the system.

★ Process in Memory



Stack

The process Stack contains the temporary data such as method/function parameters, return address and local variables.

Heap

This is dynamically allocated memory to a process during its run time.

Text

This includes the current activity represented by the value of Program Counter and the contents of the processor's registers. (The program code, also called text section)

Data

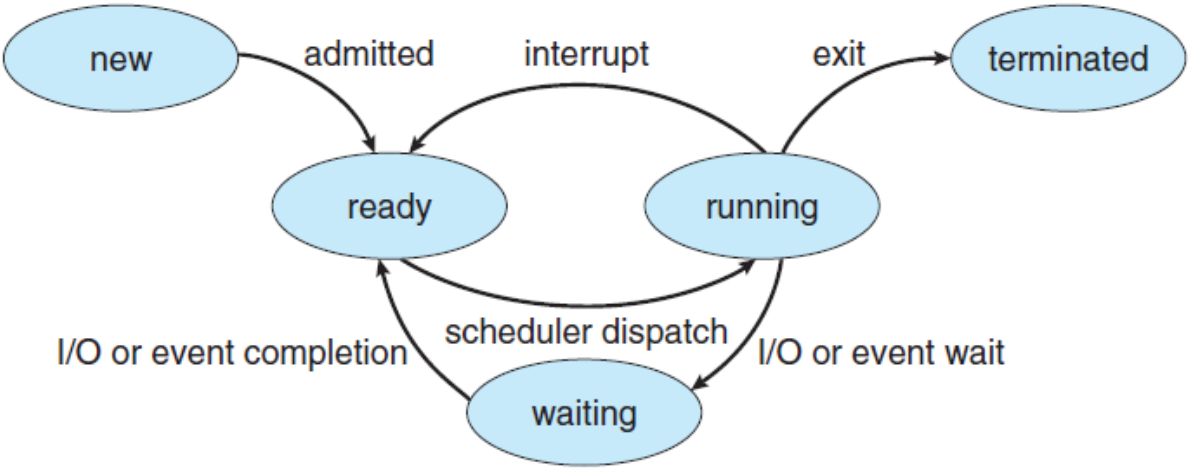
This section contains the global variables.

★ Process State

As a process executes, it changes state

- **new:** The process is being created
- **running**: Instructions are being executed
- **waiting:** The process is waiting for some event to occur
- **ready:** The process is waiting to be assigned to a processor
- **terminated:** The process has finished execution

Diagram of Process State (Process life cycle)



1.New/ Start
This is the initial state when a process is first started/created.

2.Ready The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run. Process may come into this state after Start state or while running it by but interrupted by the scheduler to assign CPU to some other process.

3.Running Once the process has been assigned to a processor by the OS scheduler, the process state is set to running and the processor executes its instructions.

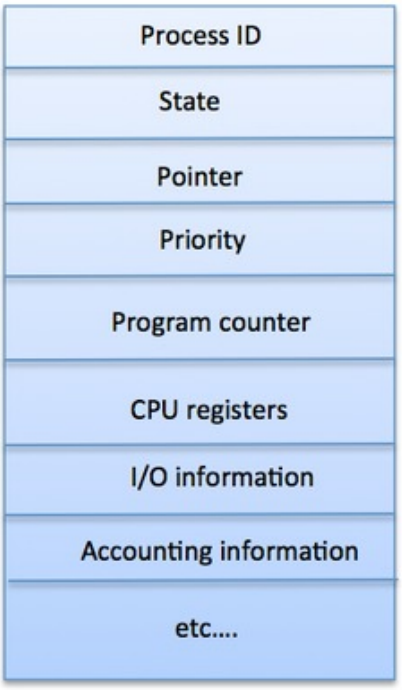
4.Waiting Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.

5.Terminated or Exit Once the process finishes its execution, or it is terminated by the operating system, it is moved to the terminated state where it waits to be removed from main memory.

★ **PCB (Process Control Block)** (also called task control block)

A Process Control Block is a data structure maintained by the Operating System for every process.

Figure 3.3 Process control block (PCB).



S.N.	Information & Description
1	**Process State** The current state of the process i.e., whether it is ready, running, waiting, or whatever.
2	**Process privileges** This is required to allow/disallow access to system resources.
3	**Process ID** Unique identification for each of the processes in the operating system.
4	**Pointer** A pointer to the parent process.
5	**Program Counter** Program Counter is a pointer to the address of the next instruction to be executed for this process.
6	**CPU registers** Various CPU registers where processes need to be stored for execution for running state.
7	**CPU Scheduling Information** Process priority and other scheduling information which is required to schedule the process.
8	**Memory management information** This includes the information of page table, memory limits, Segment table depending on memory used by the operating system.
9	**Accounting information** This includes the amount of CPU used for process execution, time limits, execution ID etc.
10	**IO status information** This includes a list of I/O devices allocated to the process.

★ What are Threads ?

Thread is a sequential flow of tasks within a process.

But the threads of a single process might share the same code and data/file.

Eg: While playing a movie on a device the audio and video are controlled by different threads in the background.

A thread has the following three components:

1. Program Counter
2. Register Set
3. Stack space

Why do we need Threads?

- Since threads use the same data and code, the operational cost between threads is low.
- Creating and terminating a thread is faster compared to creating or terminating a process.
- Context switching is faster in threads compared to processes.

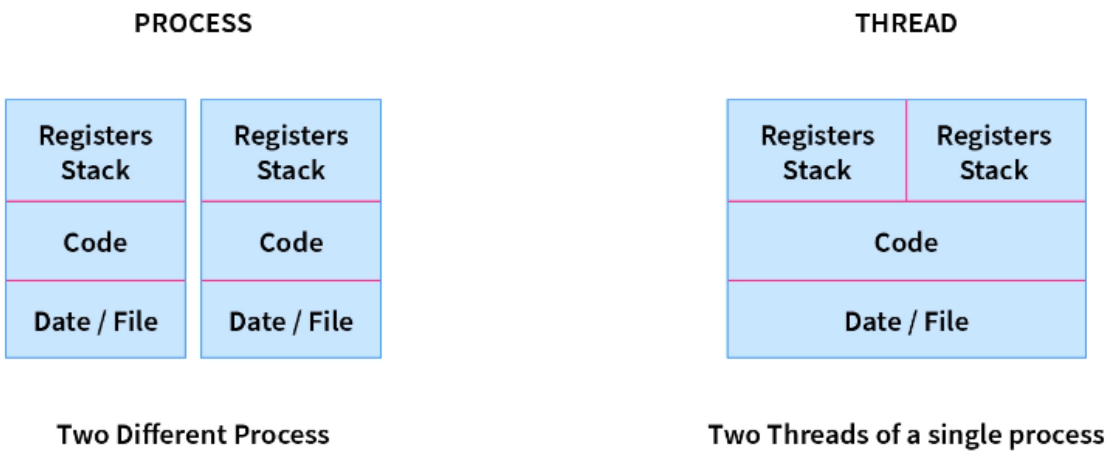
  (A context switch is a procedure that a computer's CPU (central processing unit) follows to change from one task (or process) to another while ensuring that the tasks do not conflict.)

★ What is Multithreading ?

Multithreading enables a program to create smaller task units (threads) that execute in parallel.

★ Process vs Thread

Process	Thread
Creation and termination times of process are slower	Creation and termination times of threads are faster compared to processes.
Processes have their own code and data/files.	Threads share code and data/file within a process.
The process takes more time to terminate	The threads take less time to terminate.
It takes more time for creation.	It takes less time for creation.
Context switching in processes is slower.	Context switching in threads is faster.
Communication between processes is slower.	Communication between threads is faster.
Processes are independent of each other	Threads on the other hand are independent.
Processes use more resources and hence they are termed as heavyweight processes.	Threads share resources and hence they are termed as lightweight processes.
Eg- opening two different browsers	Eg - opening two tabs in the same browsers.



★ Types of Thread

1. User Level Thread - (management done by user-level threads library)

User-level threads are implemented and managed by the user and the kernel is not aware of it.

- User-level threads are implemented using user-level libraries and the OS does not recognize these threads.
- User-level thread is faster to create and manage compared to kernel-level thread.
- Context switching in user-level threads is faster.
- If one user-level thread performs a blocking operation then the entire process gets blocked.

Three primary thread libraries:
- POSIX Pthreads
- Windows threads
- Java threads

2. Kernel level Thread - (Supported by the Kernel)

Kernel level threads are implemented and managed by the OS.

- Kernel level threads are implemented using system calls and Kernel level threads are recognized by the OS.
- Kernel-level threads are slower to create and manage compared to user-level threads.
- Context switching in a kernel-level thread is slower.
- Even if one kernel-level thread performs a blocking operation, it does not affect other threads.
  Eg: Window Solaris.

virtually all general purpose operating systems, including:
- Windows
- Solaris
- Linux
- Tru64 UNIX
- Mac OS X

★ Advantages of Threading

1. Threads improve the overall performance of a program.
2. Threads increases the responsiveness of the program
3. Context Switching time in threads is faster.
4. Threads share the same memory and resources within a process.
5. Communication is faster in threads.
6. Threads provide concurrency within a process.
7. Enhanced throughput of the system.

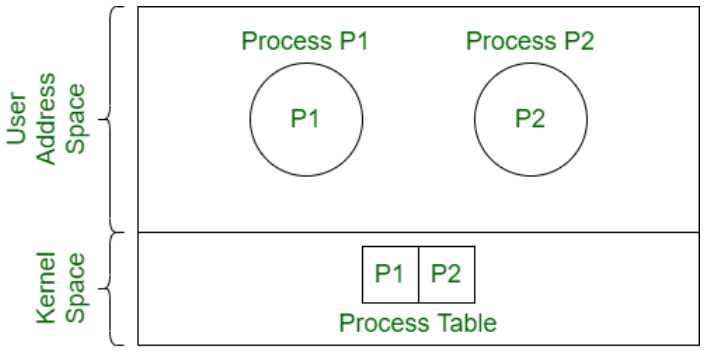
★ Thread Models in Operating System

Each thread contains a Stack and a Thread Control Block. There are four basic thread models

1. User Level Single Thread Model
2. User Level Multi Thread Model
3. Kernel Level Single Thread Model
4. Kernel Level Multi Thread Model

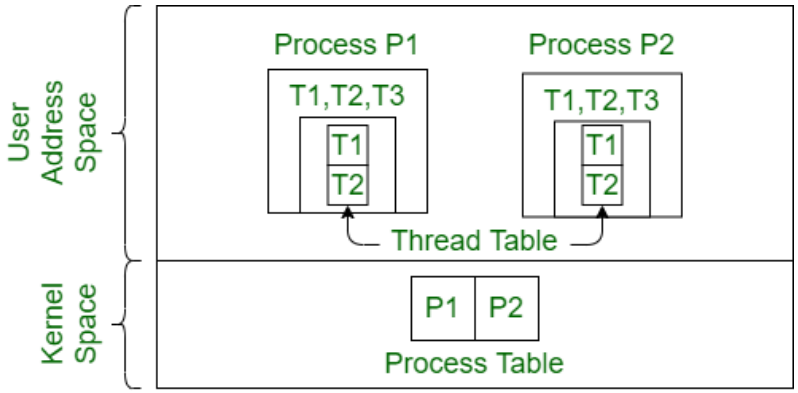
1. **User Level Single Thread Model**

- Each process contains a single thread.
- Single process is itself a single thread.
- The process table contains an entry for every process by maintaining its PCB.



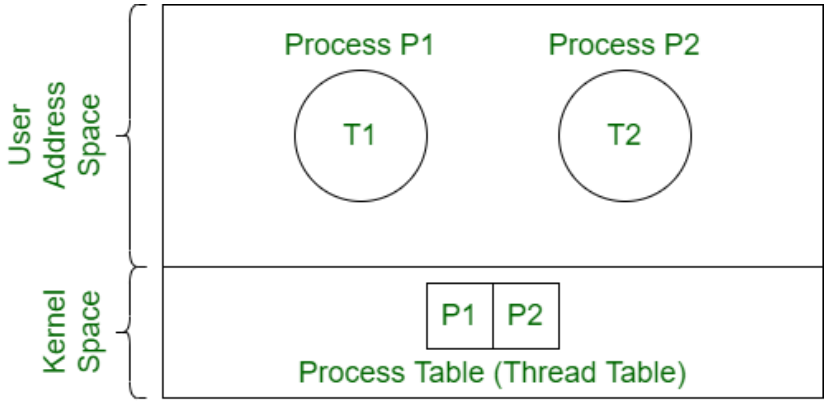
2. **User Level Multi Thread Model**

- Each process contains multiple threads.
- All threads of the process are scheduled by a thread library at user level.
- Thread switching can be done faster than process switching.
- Thread switching is independent of the operating system which can be done within a process.
- Blocking one thread makes blocking of entire process.
- Thread table maintains Thread Control Block of each thread of a process.



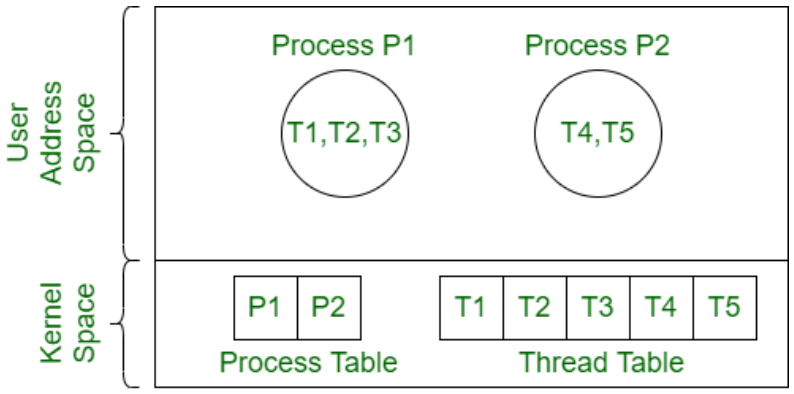
3. **Kernel Level Single Thread Model**

- Each process contains a single thread.
- Thread used here is kernel level thread.
- Process table works as a thread table.



4. Kernel Level Multi Thread Model

- Thread scheduling is done at kernel level.
- Fine grain scheduling is done on a thread basis.
- if a thread blocks, another thread can be scheduled without blocking the whole process.
- Thread scheduling at Kernel process is slower compared to user level thread scheduling.
- Thread switching involves switching.



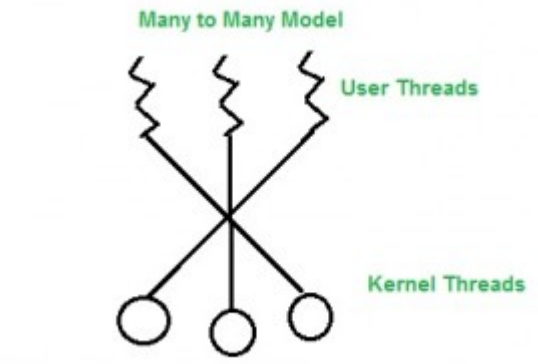
★ Multi Threading Models

Multithreading models are three types.

1. Many to many models.
2. Many to one model.
3. One to one model.

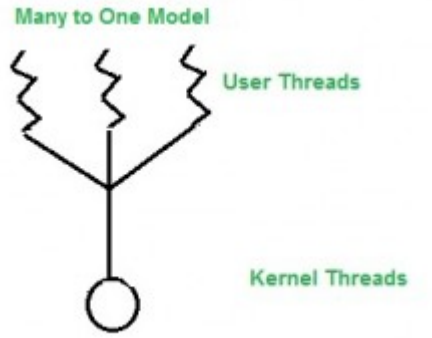
1. Many to many models.

In this model, we have multiple user threads multiplexed to the same or lesser number of kernel level threads. Number of kernel level threads are specific to the machine, the advantage of this model is if a user thread is blocked we can schedule other user threads to other kernel thread. Thus, System doesn't block if a particular thread is blocked.



2. Many to one model.

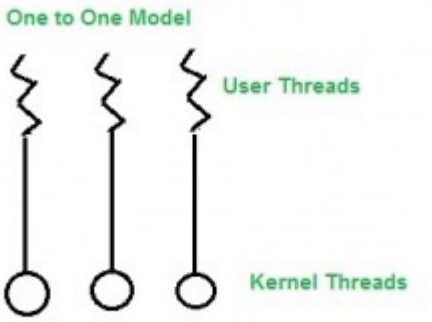
In this model, we have multiple user threads mapped to one kernel thread. In this model when a user thread makes a blocking system call entire process blocks. As we have only one kernel thread and only one user thread can access the kernel at a time, so multiple threads are not able access the multiprocessor at the same time.



3. One to one model.

In this model, one to one relationship between kernel and user thread. In this model multiple threads can run on multiple processors. Problem with this model is that creating a user thread requires the corresponding kernel thread.

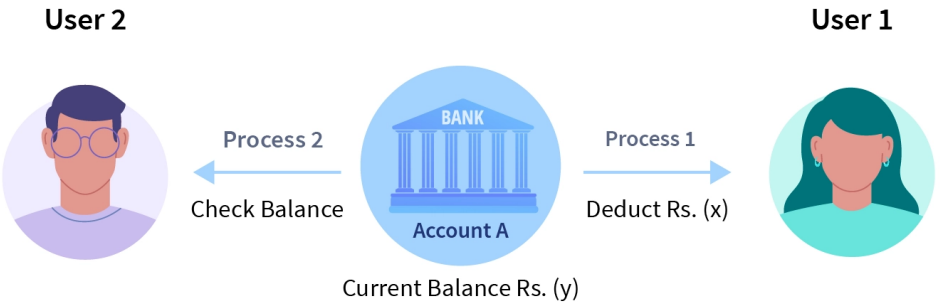
As each user thread is connected to a different kernel , if any user thread makes a blocking system call, the other user threads won't be blocked.



★ Process Synchronization

Process Synchronization is the coordination of execution of multiple processes in a multi-process system to ensure that they access shared resources in a controlled and predictable manner.

Process synchronization is the task of coordinating the execution of processes in such a way that no two processes can access the same shared data and resources.



On the basis of synchronization, processes are categorized as one of the following two types

1. Independent Process: The execution of one process does not affect the execution of other processes.

2. Cooperative Process: A process that can affect or be affected by other processes executing in the system.

Process synchronization problem arises in the case of Cooperative processes also because resources are shared in Cooperative processes.

Advantages & Disadvantages of Process Synchronization

1 . Advantages of Process Synchronization

- Ensures data consistency and integrity
- Avoids race conditions
- Supports efficient and effective use of shared resources
- Prevents inconsistent data due to concurrent access

2. Disadvantages of Process Synchronization

- Adds overhead to the system
- Increases the complexity of the system
- Can cause deadlocks if not implemented properly.
- This can lead to performance degradation

What is the main objective of process synchronization in a multi-process system?

A multi-process system's process synchronization goal is to govern and predict shared resource access. It prevents race situations and other synchronization concerns to maintain data integrity and avoid deadlocks.

What are the key requirements that any solution to the critical section problem must satisfy?

The crucial section problem solution must meet three criteria:

- Mutual Exclusion - Only one process can run in its critical section.
- Progress - Only processes not in their remainder sections can select the next process to enter the critical section if no process is in it and others are waiting.
- Bounded Waiting - The number of times a process can enter its crucial phase after making a request must be limited before it is granted.

★ Process Scheduling

Process Scheduling is the process of the process manager handling the removal of an active process from the CPU and selecting another process based on a specific strategy.

There are three types of process schedulers:

1. Long term or Job Scheduler
2. Short term or CPU Scheduler
3. Medium term Scheduler

Why do we need to schedule processes?

Process Scheduling allows the OS to allocate CPU time for each process. Another important reason to use a process scheduling system is that it keeps the CPU busy at all times. This allows you to get less response time for programs.

★ CPU scheduling

CPU scheduling is the process of deciding which process will own the CPU to use while another process is suspended. The main function of the CPU scheduling is to ensure that whenever the CPU remains idle, the OS has at least selected one of the processes available in the ready-to-use line.

There are mainly two types of scheduling methods:

1. Preemptive Scheduling:

   Preemptive scheduling is used when a process switches from running state to ready state or from the waiting state to the ready state.

2. Non-Preemptive Scheduling:

   Non-Preemptive scheduling is used when a process terminates , or when a process switches from running state to waiting state.

About CPU scheduling algorithms in operating systems:

1. First Come First Serve(FCFS).
2. Shortest Remaining Time First.

1. First Come First Serve(FCFS).

   FCFS is considered to be the simplest of all operating system scheduling algorithms. First come first serve scheduling algorithm states that the process that requests the CPU first is allocated the CPU first and is implemented by using FIFO queue.

   Characteristics of FCFS

   - Tasks are always executed on a First-come, First-serve concept.
   - FCFS is easy to implement and use.
   - FCFS supports non-preemptive and preemptive CPU scheduling algorithms.
   - This algorithm is not much efficient in performance, and the wait time is quite high.

   Advantages of FCFS
   - Easy to implement.
   - First come, first serve method.

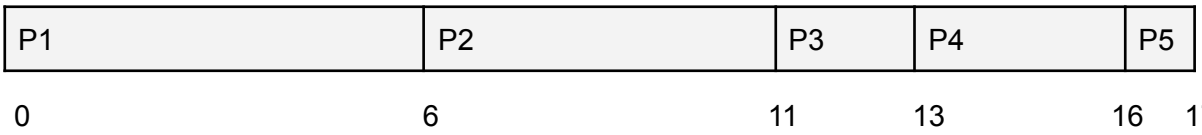
   Disadvantages of FCFS:
   - FCFS suffers from the Convoy effect.
   - The average waiting time is much higher than the other algorithms.
   - FCFS is very simple and easy to implement and hence not much efficient.
   -

1) Consider the following four processes to run in a single CPU. All times are given in milliseconds.

Process	Arrival Time	Burst Time
P1	0	6
P2	1	5
P3	2	2
P4	4	3
P5	5	1

a). Draw the Gantt chart to show the execution of the above process for **First Come First Serve (FCFS)**

Gantt chart



b) Calculate the **average completion time** when scheduling these processes according to FCFS algorithms.

Process	Arrival Time	Burst Time	completion time
P1	0	6	6
P2	1	5	11
P3	2	2	13
P4	4	3	16
P5	5	1	17

Average CT = 6+11+13+16+17 / 5 = 12.6 milliseconds.

c) Draw the Gantt chart to show the execution of the above process for **Shortest Remaining Time First (SRTF)** algorithms.

d) Calculate the average completion time when scheduling these processes according to **SRTF** algorithms.

Process	Arrival Time	Burst Time	completion time
P1	0	6	
P2	1	5	
P3	2	2	
P4	4	3	
P5	5	1	

★ What is the Deadlock ?

A deadlock is a situation in which two computer programs sharing the same resource are effectively preventing each other from accessing the resource, resulting in both programs ceasing to function.

Deadlock can arise if the following four conditions hold simultaneously (Necessary Conditions):

- **Mutual Exclusion:** Two or more resources are non-shareable (Only one process can use at a time).
- **Hold and Wait:** A process is holding at least one resource and waiting for resources.
- **No Preemption:** A resource cannot be taken from a process unless the process releases the resource.
- **Circular Wait:** A set of processes waiting for each other in circular form.

Methods for handling deadlock

There are three ways to handle deadlock .

- Deadlock prevention or avoidance:

  Prevention: Stop deadlocks by managing resource allocation to ensure the necessary conditions for deadlock can't happen.

  Avoidance (Banker's Algorithm): Safely allocate resources by analyzing future requests to avoid potential deadlocks.

- Deadlock detection and recovery

  Allow dreadlocks, periodically check for them, and take action (like killing processes or releasing resources) to resolve them.

- Deadlock ignorance

  Ignorance (Ostrich Method): Ignore the deadlock issue and assume they're rare or won't happen, without actively preventing or resolving them.

Banker's Algorithm

Consider a system with five processes P0 through P4 and three resources of type A, B, C. Resource type A has 10 instances, B has 5 instances and type C has 7 instances. Suppose at time t0 following snapshot of the system has been taken

Process	Allocation	Max	Available
	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2
P_1	2 0 0	3 2 2	
P_2	3 0 2	9 0 2	
P_3	2 1 1	2 2 2	
P_4	0 0 2	4 3 3	

Answer -

Banker's algorithm for deadlock avoidance | An example.

Process	Allocation	Max	Available (Work)	Need (Max - Allocation)
	A B C	A B C	A B C	A B C
p_0	0 1 0	7 5 3	~~3 3 2~~	7 4 3
p_1	2 0 0	3 2 2	~~5 3 2~~	1 2 2
p_2	3 0 2	9 0 2	~~7 4 3~~	6 0 0
p_3	2 1 1	2 2 2	~~7 4 5~~	0 1 1
p_4	0 0 2	4 3 3	~~7 5 5~~	4 3 1
			10 5 7	

Need $_i$ ≤ work , work = work + allocation

P_0 7 4 3 ≤ 3 3 2 X

P_1 1 2 2 ≤ 3 3 2 ✔

w = w + allocation
= 3 3 2 + 2 0 0
= 5 3 2

P_2 6 0 0 ≤ 5 3 2 X

P_3 0 1 1 ≤ 5 3 2 ✔

w = w + allocation
= 5 3 2 + 2 1 1
= 7 4 3

P_4 4 3 1 ≤ 5 3 2 ✔

w = w + allocation
= 7 4 3 + 0 0 2
= 7 4 5

P_0 7 4 3 ≤ 7 4 5 ✔

w = w + allocation
= 7 4 5 + 0 1 0
= 7 5 5

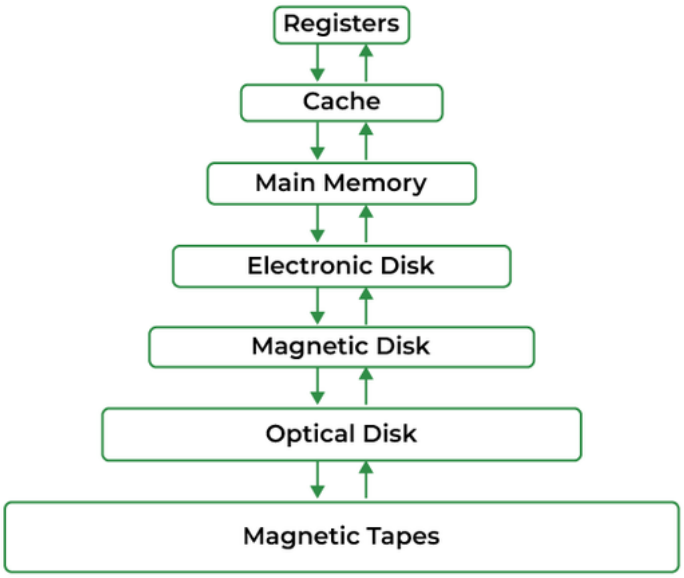
P_2 6 0 0 ≤ 7 5 5 ✔

w = w + allocation
= 7 5 5 + 3 0 2
= 10 5 7

safe sequence is a, < P1, P3, P4, P0, P2>

★ **Memory management**

Memory management is the process of controlling and coordinating a computer's main memory.



Why is Memory Management Required?

- Allocate and deallocate memory before and after process execution.
- To keep track of used memory space by processes.
- To minimize fragmentation issues.
- To proper utilization of main memory.
- To maintain data integrity while executing the process.

Address Spaces

1. Logical Address Space

   An address generated by the CPU is known as a “Logical Address”. It is also known as a Virtual address. Logical address space can be defined as the size of the process. A logical address can be changed.

2. Physical Address Space

   An address seen by the memory unit (i.e the one loaded into the memory address register of the memory) is commonly known as a “Physical Address”. A Physical address is also known as a Real address. A physical address is computed by MMU (Memory Management Unit). The physical address always remains constant.

Virtual Memory vs Physical Memory

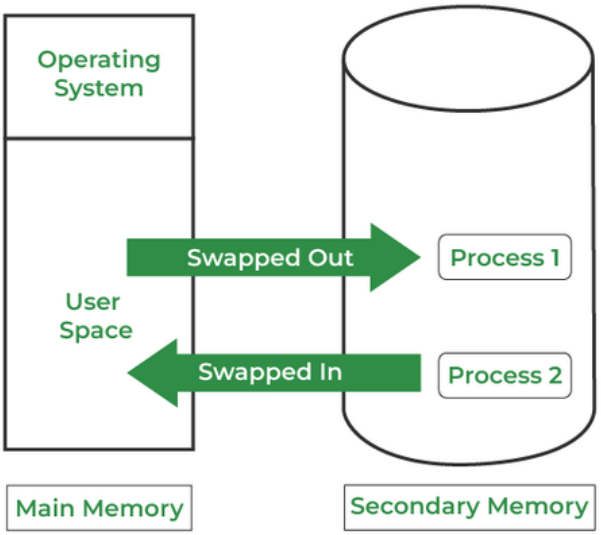
Virtual Memory	Physical Memory (RAM)
Speed is slower	Speed is faster
Uses the paging technique	Uses the swapping technique
Limited by the size of the physical memory	Limited to the size of the RAM chip
No direct access to the CPU	Can directly access the CPU
Limited by the size of the computer's hard drive	Can add RAM by installing more RAM chips

★ Swapping

Swapping is a process of swapping a process temporarily into a secondary memory from the main memory, which is fast compared to secondary memory. A swapping allows more processes to be run and can be fit into memory at one time.

The procedure by which any process gets removed from the hard disk and placed in the main memory or RAM commonly known as Swap In.

On the other hand, Swap Out is the method of removing a process from the main memory or RAM and then adding it to the Hard Disk.



The advantages/benefits of the Swapping technique

- The swapping technique mainly helps the CPU to manage multiple processes within a single main memory.
- This technique helps to create and use virtual memory.
- The CPU can perform several tasks simultaneously. Thus, processes need not wait too long before their execution.
- This technique is economical.
- This technique can be easily applied to priority-based scheduling in order to improve its performance.
-

Disadvantages of Swapping

- Shared resources accessed by multiple processes involved in swapping can cause inefficiencies.
- Inadequate swapping algorithms can increase page faults, hampering overall processing performance.
- During heavy swapping, sudden power loss may result in losing all program-related information.

★ Virtual Memory

Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of the main memory.

Virtual memory is a memory management technique where secondary memory can be used as if it were a part of the main memory.

Virtual memory is implemented using

1. Demand Paging
2. Demand Segmentation.

Advantages of Virtual Memory

- It has twice the capacity for addresses as main memory.
- It makes it possible to run more applications at once.
- Memory isolation has increased security.
- Memory allocation is comparatively cheap.
- It doesn't require outside fragmentation.
- Automatic data movement is possible.

Disadvantages of Virtual Memory

- It can slow down the system performance
- It can increase the risk of data loss or corruption
- It can increase the complexity of the memory management system

★ Paging in Operating System

Paging is a static memory allocation method that allows a process's physical address space to be of a non-contiguous type. The physical memory is divided into fixed-size blocks called page frames.The mapping between logical pages and physical page frames is maintained by the page table.which is used by the memory management unit to translate logical addresses into physical addresses. The page table maps each logical page number to a physical page frame number.

Paging is a storage mechanism used in OS to retrieve processes from secondary storage to the main memory as pages.

The Physical Address Space is conceptually divided into a number of fixed-size blocks, called **frames.**

The Logical Address Space is also split into fixed-size blocks, called **pages.**

★ Segmentation in Operating System

Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.

A table stores the information about all such segments and is called a Segment Table.

Types of Segmentation in Operating System

1. Virtual Memory Segmentation

   Each process is divided into a number of segments, but the segmentation is not done all at once. This segmentation may or may not take place at the run time of the program.

2. Simple Segmentation

   Each process is divided into a number of segments, all of which are loaded into memory at run time, though not necessarily contiguously.

Advantages of Segmentation in Operating System

- No Internal fragmentation.
- Segment Table consumes less space in comparison to Page table in paging.
- As a complete module is loaded all at once, segmentation improves CPU utilization.

- Segmentation is a method that can be used to segregate data from security operations.

Disadvantages of Segmentation in Operating System

- Causes wasted memory due to small unused spaces between segments.
- Managing different segment sizes adds system complexity.
- Sharing data among segments or processes becomes complex.
- Retrieving data might take longer due to scattered segments.

★ File System

A File System is a data structure that stores data and information on storage devices (hard drives, floppy disks, etc.), making them easily retrievable.

Some common types of file systems include:

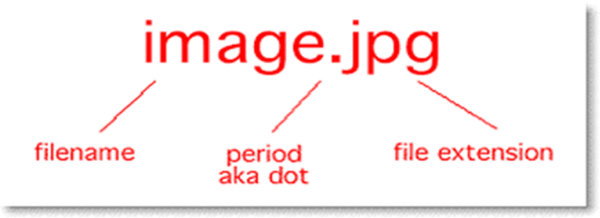
- **FAT (File Allocation Table):** An older file system used by older versions of Windows and other operating systems.
- **NTFS (New Technology File System):** A modern file system used by Windows. It supports features such as file and folder permissions, compression, and encryption.
- **ext (Extended File System):** A file system commonly used on Linux and Unix-based operating systems.
- **HFS (Hierarchical File System):** A file system used by macOS.
- **APFS (Apple File System):** A new file system introduced by Apple for their Macs and iOS devices.

The advantages of using a file system

- Organization: A file system allows files to be organized into directories and subdirectories, making it easier to manage and locate files.
- Data protection: File systems often include features such as file and folder permissions, backup and restore, and error detection and correction, to protect data from loss or corruption.
- Improved performance: A well-designed file system can improve the performance of reading and writing data by organizing it efficiently on disk.

Disadvantages of using a file system

- Compatibility issues: Different file systems may not be compatible with each other, making it difficult to transfer data between different operating systems.
- Disk space overhead: File systems may use some disk space to store metadata and other overhead information, reducing the amount of space available for user data.
- Vulnerability: File systems can be vulnerable to data corruption, malware, and other security threats, which can compromise the stability and security of the system.



Examples of web file extensions:

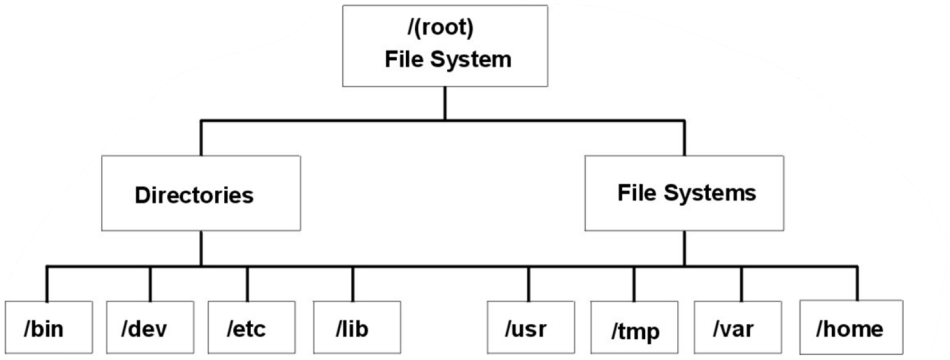
- Web pages: .html .htm .php .jsp
- Web image file types: .png .jpg .svg .gif
- CSS and JavaScript files: .css .js
- Other file extension:

- MS Word: .docx
- Excel: .xlsx

Operations on the File

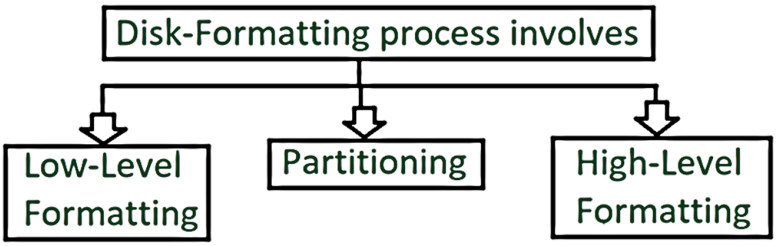
- Create operation
- Read operation
- Write operation
- Open operation
- Delete operation
- Rename operation

Linux file system



★ Input / Output management and Disk scheduling

Disk Formatting



1. Low-level Formatting

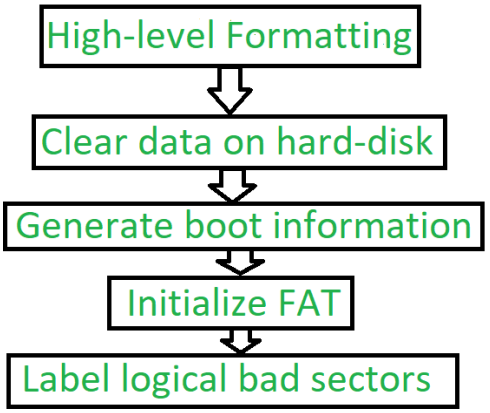
Low level formatting is a type of physical formatting. It is the process of marking cylinders and tracks of the blank hard-disk. After this there is the division of tracks into sectors with the sector markers. Now-a-days low-level formatting is performed by the hard-disk manufacturers themselves.

2. Partitioning

As suggested from the name, partitioning means divisions. Partitioning is the process of dividing the hard-disk into one or more regions. The regions are called partitions. It can be performed by the users and it will affect the disk performance.

3. High-level Formatting

High-level formatting is the process of writing. Writing on a file system, cluster size, partition label, and so on for a newly created partition or volume. It is done to erase the hard-disk and again install the operating system on the disk-drive.



Disk scheduling

Disk scheduling is done by operating systems to schedule I/O requests arriving for the disk. Disk scheduling is also known as I/O Scheduling.

★ Unix, Linux, and Android

Linux

Linux is a group of open-source Unix-like operating systems which was developed by Linus Torvalds. It is a package of Linux distribution.

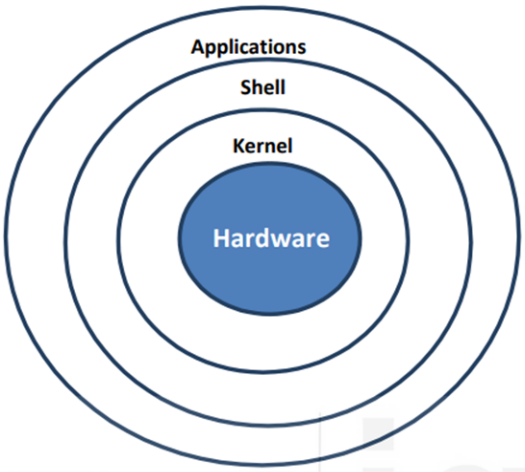


Fig 1: Linux Architecture

Android

Android is a mobile operating system that is provided by Google. It is based on the modified version of the Linux kernel and other open-source software.

Unix

Unix is a portable, multi-tasking, bug-fixing, multi-user operating system developed by AT&T.

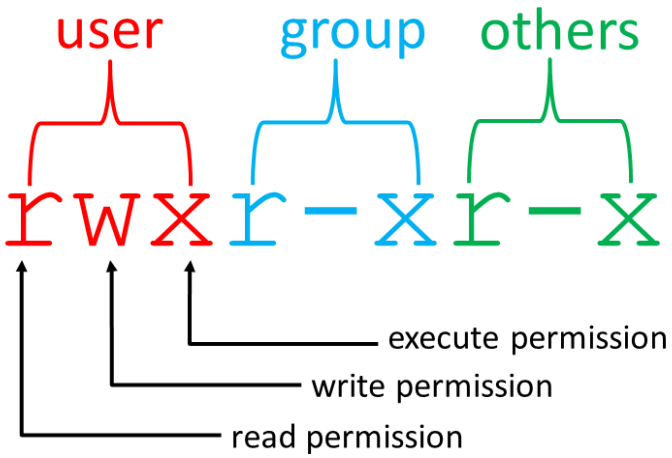
Features of UNIX Operating System

- Multitasking:
- Multi-user:
- Portability:
- File Security and Protection:
- Command Structure:
- Communication:
- Open Source:
- Accounting:
- UNIX Tools and Utilities:

★ **Linux Security**

Mention the various requirements of security.

- Authorization
- Authenticity
- Privacy
- Integrity, and Availability




```
rwx  : read/write/execute permission for owner (user)
r-x  : read/execute only permission for group members
r-x  : read/execute only permission for all others
```