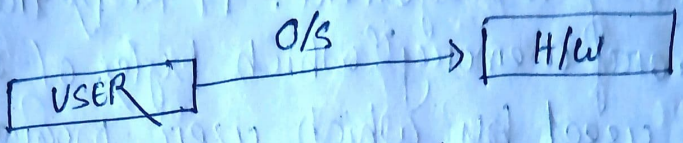
Operating System

Chapter-1

Introduction to O/S

-> It is a program that acts as an medium between user of the computer & computer hardware.



-> It is a program that controls the execution of application program.

-> Also acts as an interface between application & H/w.

Main Objective of O/S :-

① Convenience → It makes the computer more ~~efficient~~ convenient to use.

② Efficiency → It allows the computer system resources to be used in efficient manner.

③ Ability to Evolve → An os should be constructed in such a way that to permit the effective development, testing and intr" of new system function at the same time without interfering the service.

Functions of O/S

① Security

② Control over system performance

③ Job accounting : Keeps track of time and resource by various users.

④ Error detecting aids → detect and avoid error & other manufacturing fault.

⑤ Co-ordination between other s/w and user

⑥ Memory management: It keeps track of primary memory i.e. which bytes of memory are used by which user program.

⑦ Processor Management: the os decide the order in which processes have access to the processor, and how much processing time each process has.
⇒ called Process scheduling.

⑧ Device Management :- Keeps track of all devices connected to the system.

⑨ File Management :- Keeps track of where information is stored, user access setting and status of every file, and more.

Characteristics of an operating System.

① Multi-User ⇒ Two (or) more users have individual accounts that allow them to work with programs and peripheral devices at the same time.

② Multi-Tasking ⇒ The computer is capable of operating multiple applications at the same time.

⑤ Multiprocessing ⇒ The o/s can support two (or more) CPUs.

④ Multithreading ⇒ A Program can be broken into smaller parts that are loaded as needed by the o/s.

⇒ multithreading allows individual programs to be multitasked.

Structures

① Simple Batch
② Multiprogrammed
③ Time shared
④ Personal Computer
⑤ Parallel
⑥ Distributed
⑦ Real time

✱ Serial Processing System

⇒ Earliest Computing

⇒ No o/s is present then:

⇒ The programmer directly interacted with the computer s/w

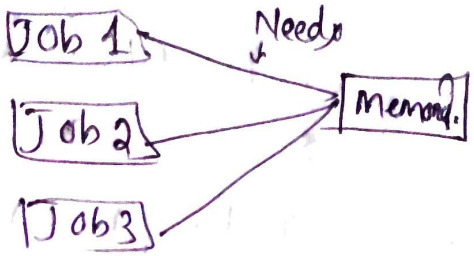
⇒ Computer ran from a console with display light, toggle, switches, some I/p devices & a printer.

⇒ User had access to the computer in series.

(One process after one)

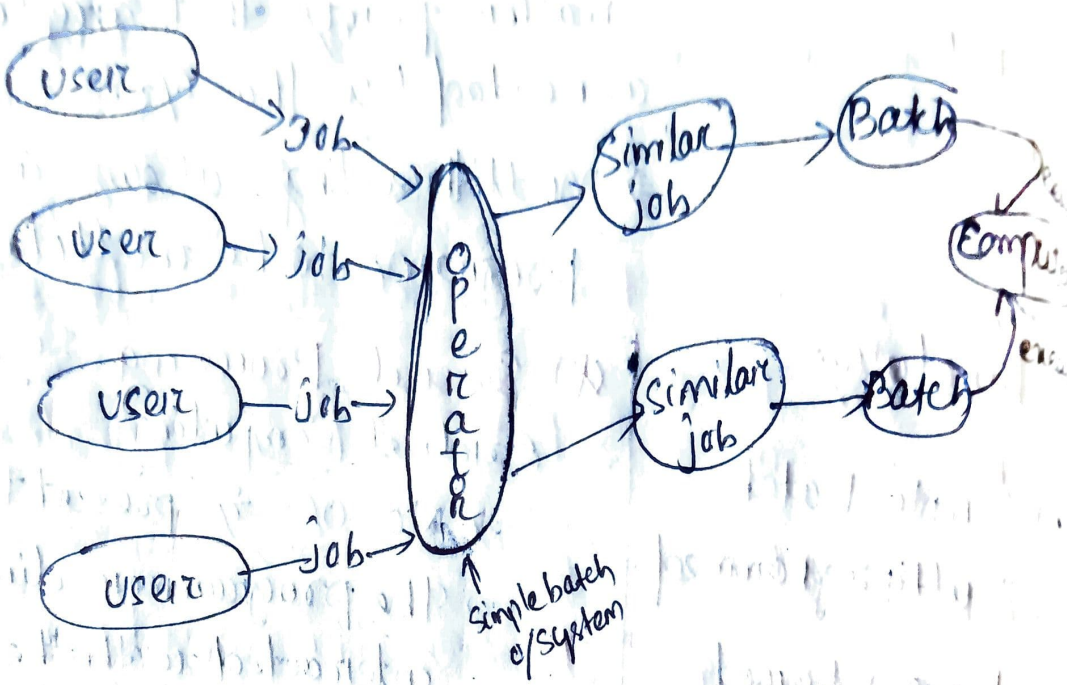
① Simple Batch System

Batch → Set of jobs with similar needs.



⇒ In Batch processing same type of jobs batch together & execute at a time.

fig.



⇒ The is a special program called monitor that manages the program on each program that run in the batch.

⇒ There is no direct interaction b/w user & compute

Adv

⇒ Pay roll run of company

⇒ Gas & electricity bill produce by batching.

disadvantages

⇒ No interaction user → com

⇒ No mechanism to prioritise the proces.

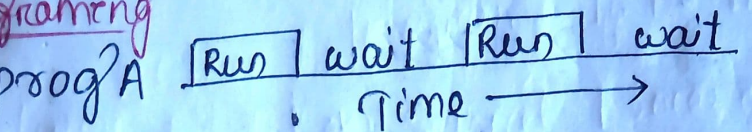
⇒ cpu is often idle, bcz the speed of I/O devices is slower tha cpu.

② Multi-Programmed System

⇒ In this os, picks & begins to execute one job from memory.

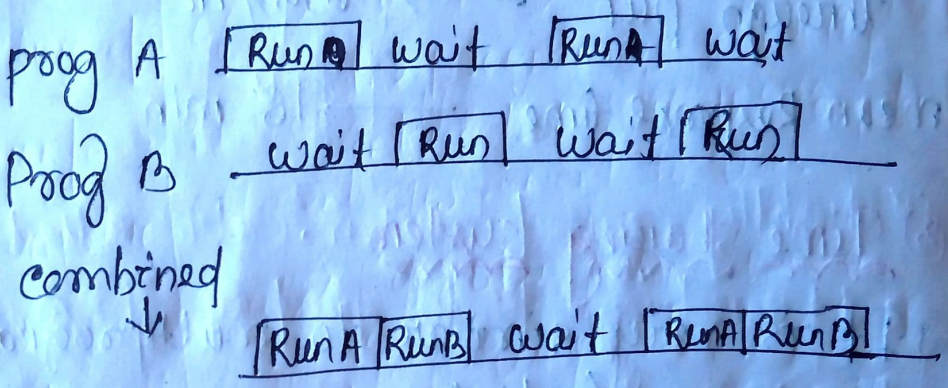
⇒ In this system, CPU will never be idle & keeps on processing.

Uniprogramming

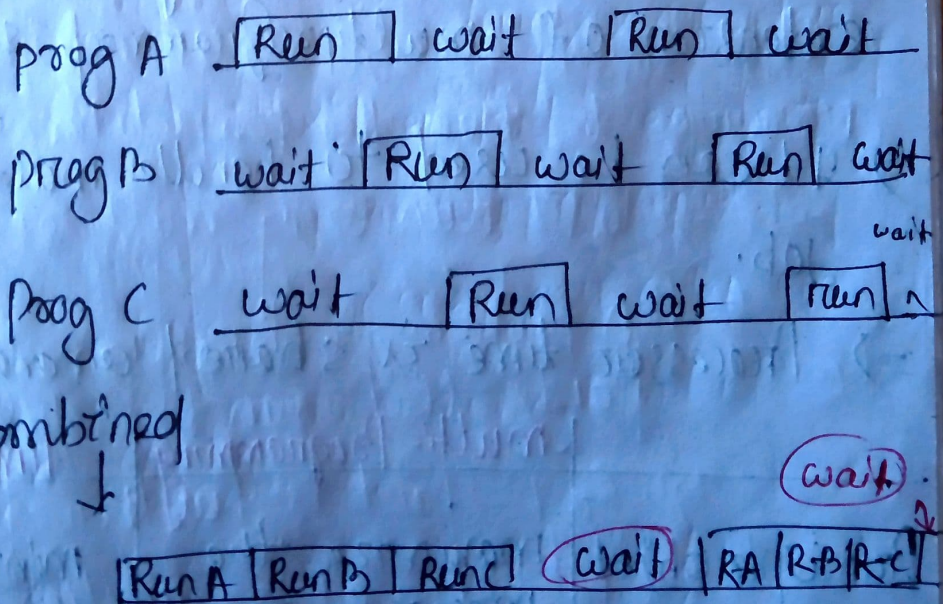


multi-Programming

2 programs



3 programs



⇒ It is also known as multi-tasking.

Effects of multiprogramming or Resource utilization

	Uni-Programming	Multiprogramming
Processor use →	20%	40%
Memory use →	33%	67%
Disk use →	33%	67%
Printer Use →	33%	67%
Elapsed time →	30 min	15 min
through put →	6 jobs/hr	12 jobs/hr
mean Response time →	18 min	10 min

(3) Time sharing system

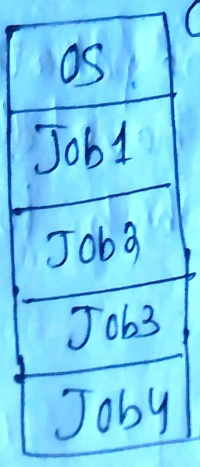
⇒ It is very similar to multiprogramming batch system. In fact time sharing system are an extension of multiprogrammed system.

⇒ It can be used to handle multiple interactive job.

⇒ Processor time is shared among multiple users

	multi-Programming	Time sharing
Principle objective	maximize the processor use	minimize response time.
Source of directive to OS	Job control language commands provided with the job.	Commands entered at the terminal

Time sharing system



⇒ The user can recieve an immediate response. (bcz the job are executed so frequently)

Adv

① Provide quick response
② Avoidy duplication of s/w.
③ Reduces the cpu idle time

Disadvantage

① problem of reliability
② question of security
③ Problem of data communication.

④ Personal Computing System

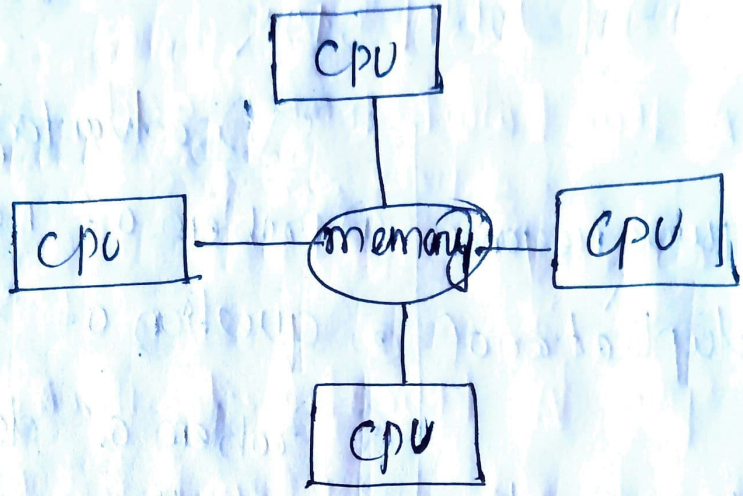
⇒ Single user system & it's portable.
⇒ I/O devices → keyboard, mouse, display on monitor, small printers etc.
⇒ Types → Laptops, smart cards, wireless devices.
⇒ Single user system may not need advanced cpu utilization or protection features

adv

⇒ user convinience
⇒ Responsiveness

⑤ Parallel System

⇒ A system is said to be parallel system in which multiple process have direct access to shared memory which form a common address space.



(*) Tightly coupled system. (also called)

⇒ It is designed to speed up the execution of program.

⇒ It is simultaneous use of multiple compute resources to solve the computational problems.

⇒ To be run using multiple CPU's.

⇒ A problem is broken into discrete parts that can be solved concurrently.

⇒ Each part is further broken down to series of instruction.

⇒ Instruction from each part executes simultaneously on different CPU's.

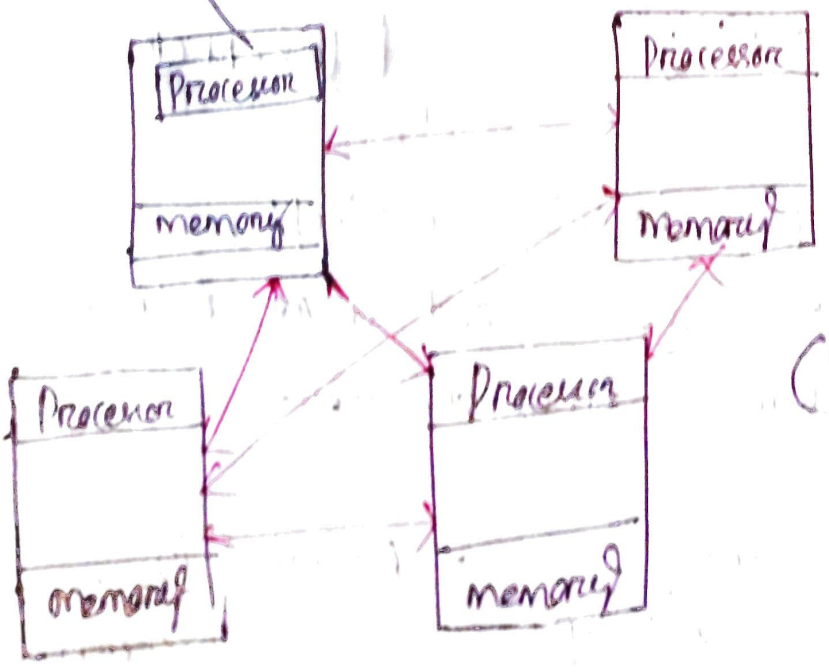
Application

1. Parallel computing could be two servers that share the workload of routing mail, solving mathematical Problem.
2. Super Computers are usually placed in parallel system architecture.
3. Terminals are connected to single server.

Advantages	dis-advantage
⇒ Do multiple things at same time.	⇒ Lack of scalability bet memory & CPU
⇒ Cost saving, time saving	⇒ Programmer / user responsibility for correct access of global memory.
⇒ over coming memory constraints.	

6. Distributed System

⇒ It is collection of independent computers, interconnected via network, capable of collabrating on a task.

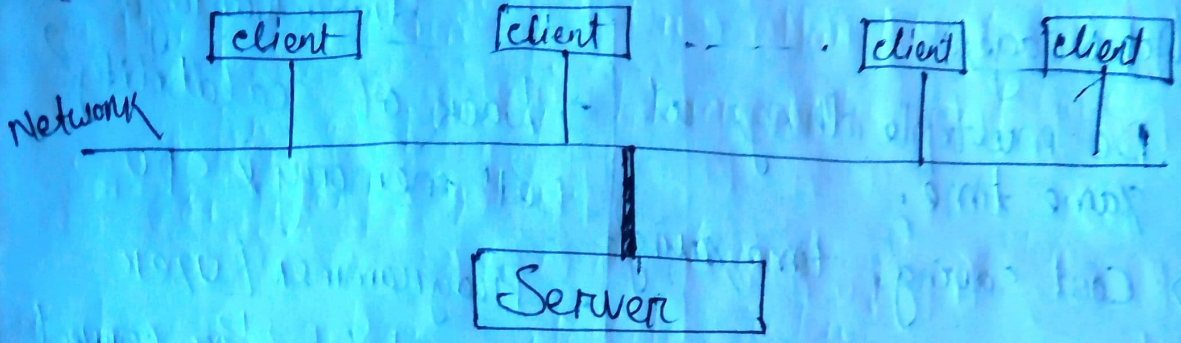


Examples

⇒ Telephone network & cellular network.
⇒ Computer network such as internet.
⇒ ATM machine.
⇒ Mobile computing etc.

⇒ This en follow the client-server system.

Client-Server System



⇒ Server broadly catagorized

① computer Server
↓
It provides an interface to which client can send request to perform an action & Server send back result to client.

② File Server
⇒ It provides a file system interface where client create, update, read & delete file present in the fileserver.

⇒ Distributed is loosely coupled:-
BCZ there is not shared memory, Various communication lines are there.

Adv
→ Resource sharing ⇒ Reliability
→ Computation Speed up ⇒ Communication ⇒ so-enal

Difference between Parallel & Distributed System

	Parallel	Distributed
memory: →	Tightly coupled system. (shared memory)	loosely coupled system. distributed memory (own)
control: →	global clock control	No global clock control. (due to independent system)
processor interconnection: →	order of Tbps	order of Gbps
main focus →	Performance, scientific computing.	Performance (cost, scalability), Reliability, resource sharing

4) Real-time O/s (Imp)

⇒ A Real time system is defined as a data processing system in which the time interval required to process & response to input/output is so small that it controls the Environment.

⇒ The time time taken from I/p to O/p task is Response time.

⇒ Real time O/s responds to I/p immediately. (Real time)

⇒ Here the task is completed within a specified time delay.

⇒ In real life situation like controlling traffic signals or a nuclear reaction (or

an aircraft.

⇒ The os has respond quickly.

⇒ A system is said to be Real time if it is required to complete the work & deliver its service on time.

Ex: Flight control System.

Types of Real Time os

① Hard Real-Time System:- Hard Real time system is purely deterministic and time constraint system.

Ex ⇒ missile Launching system.
⇒ Satellite System.
⇒ AirBag controls in Car.

② Soft-Real Time System:- In soft real time time system, the meeting of deadline is not compulsory for everytime for every task but process should get processed and give the result.

Ex:- Personal Computer.
Audio & video System etc.

Characterstic of Real Time os

- Consistency
- Reliability
- Predictibility
- Performance
- Scalability

Function of RTOS

⇒ Task management
⇒ Scheduling
⇒ Resource Allocation
⇒ Interrupt Handling.

Operating System Services

→ Operating System provided services to both program as well as user.

→ OS provide an environment to program to execute.

→ OS provide services to user to execute program in convenient manner.

OS common services

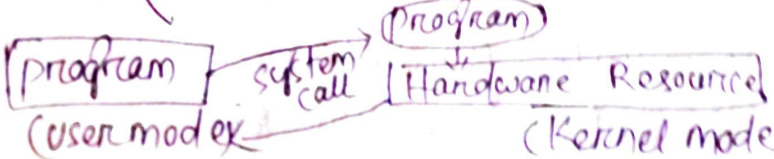
1. Program execution
2. I/O operation
3. File system manipulation
4. Communication.
5. Error detection
6. Resource Allocation.
7. Protection.

System Call

→ It's a programmatic way in which a computer program request a service from the kernel of the operating System.

program executes in 2 modes
- Usermode (cannot access any h/w resource)
- Kernel mode (can access hardware resource like, ROM, printer)

⇒ When program needs any hardware resource, it needs to make a call to the kernel.



⇒ Due to security reasons, user applications are not given access to hardware resources.
⇒ When they need to do any I/O or require some memory, it requests Os to use all these.
⇒ This request is made through system calls.

Types of System Calls

1. Process Control
2. File Management
3. Device management.
4. Information management
5. Communication.

1. Process Control

⇒ Those system calls deals with process.

Ex
1. end, abort
2. load, execute, (Fork system call) imp
3. create process, terminate process
4. get process attributes, set process attributes
5. wait for time
6. wait event, signal event
7. allocate & free memory

2. File management

⇒ Those system calls are responsible for file manipulation such as:

(i) create file, delete file
(ii) open, close file
(iii) reads, write, reposition
(iv) get & set file attributes.

③ Device Management

⇒ A process may used several resources to execute main memory, disk drives, access files, I/O device & so on.

⇒ If these resources are free, they will be allocated to process otherwise the process have to wait until resource available.

⇒ These system calls are responsible for device manipulation such as

(i) request device, release device

(ii) read, write

(iii) get device attributes, set device attributes

(iv) logically attach or detach devices.

④ Information Management -

⇒ These system calls handle information & its transfer between the O/S and the user program.

Ex 1. Most system have a system call to returns the current date and time.

2. the number of current users, the version of OS.

3. the amount of free memory (or disk space & so on.

⇒ Examples

(i) to get time or date, set time or date

(ii) get system date, set system data.

(iii) get & set process, file or device attributes.

⑤ Communication

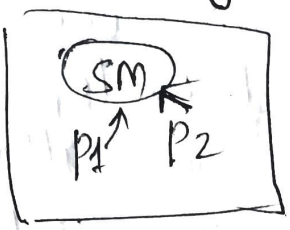
These system calls are useful for inter process communication.

IPC → message passing model
→ shared memory model.

⇒ In message passing model, the communication processes exchange message with one another to transfer information.

$P_1 \rightarrow P_2$
→ send (x, P_2)

⇒ In shared memory model, processes use shared memory to exchange information by reading & writting on it.



Examples

1. create, delete communication connection.
2. Send, Receive messages.
3. transfer status information
4. attach & detach remote devices

Process Management

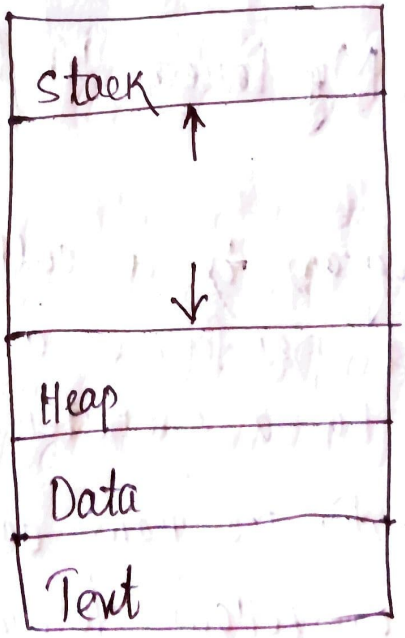
Process Concept

→ A process is basically a program in execution. The execution of a process must progress in a sequential fashion.

→ We can say, we write our computer programs in a text file and when we execute this program, it becomes a process which performs all the tasks mentioned in the program.

→ When a program is loaded into the memory and it becomes a process, It can be divided into four section - stack, heap, text and data.

diagram



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

2 Heap : → This is dynamically allocated memory to a process during it's run time.

3. Text: → This includes the current activity represented by the value of program counter and the contents of the Processors registers.

4. Data: → This section contains the global and static variables.

Program

→ A Program is a piece of code which may be a single line or millions of lines. A computer Program is usually written by a computer Programmer in a programming language.

Eg
```
#include<stdio.h>
Void main()
{
 Printf("My Paresh Singh");
}
```

⇒ A computer program is a collection of instruction that performs a specific task, when executed by a computer.

⇒ A part of computer program or we can say the step we follow to write a program and the logic behind that is known as algorithm.

⇒ A collection of computer programs, Libraries and related data are referred to as a software.

Processes and Threads

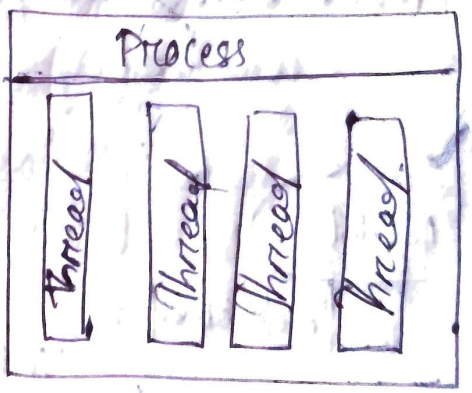
Process

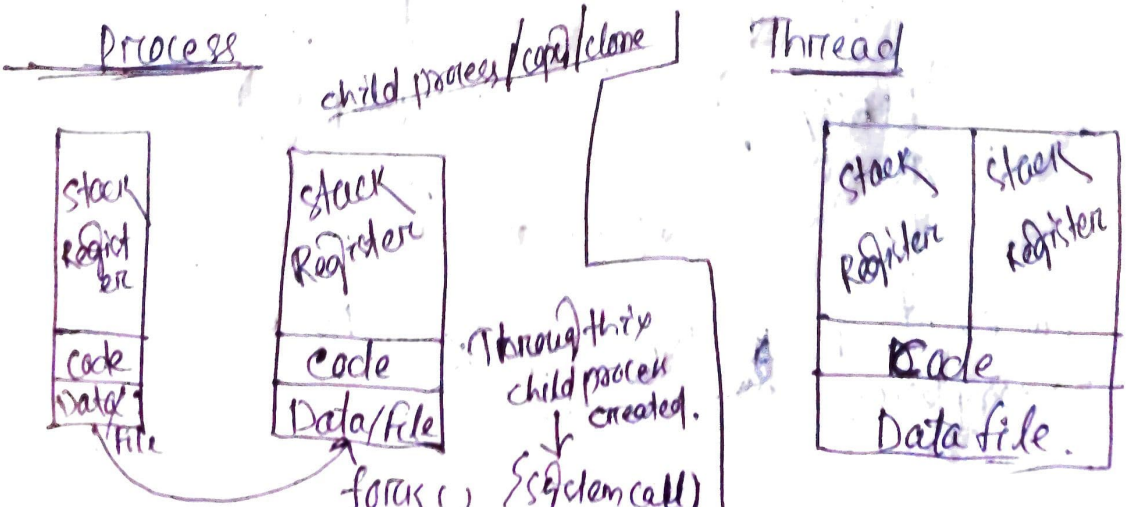
1. System calls involved in process.
2. OS treats different process differently.
3. Different process have different copies of data, file, code.
4. context switching is slower.
5. Blocking a process will not block another.
6. Independent

Threads

1. There is no system call involved.
2. All user level threads treated as single task for OS.
3. Threads share same copy of code and data.
4. context switching is faster.
5. Blocking a thread will block another.
6. Inter-dependent.

We can show like this:





Process state

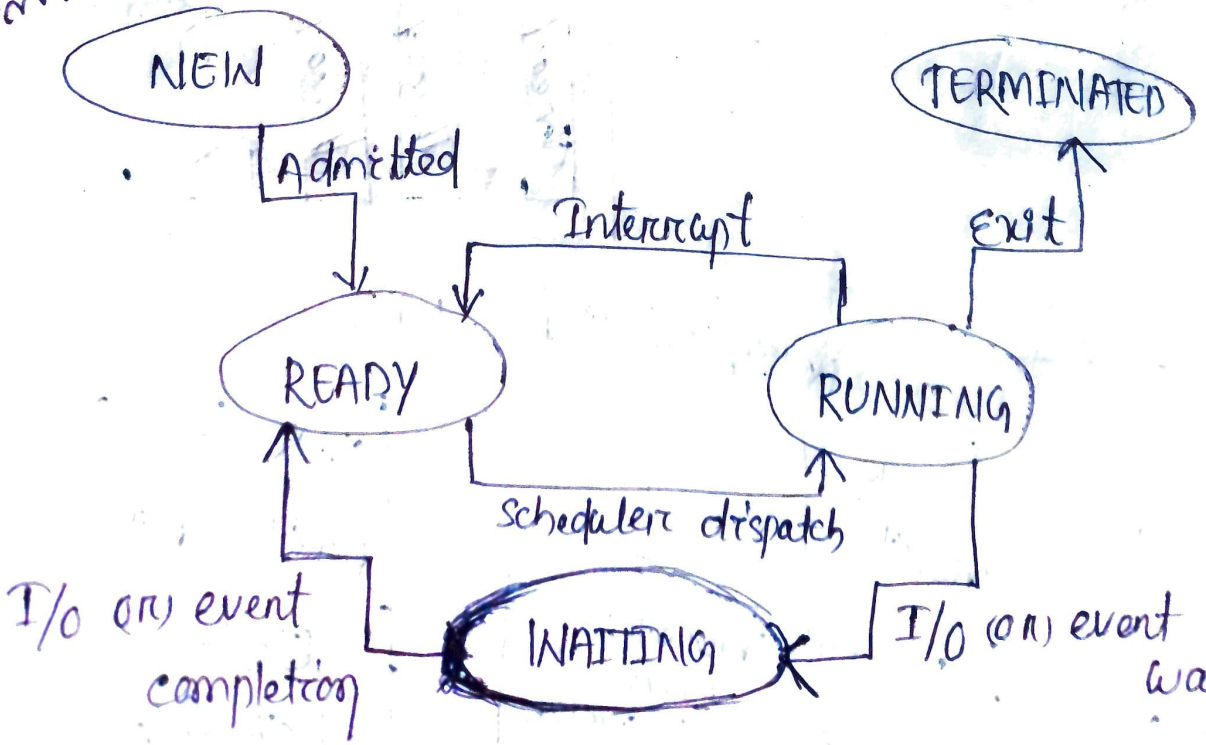
→ As a process executes, it changes state.

⇒ The state of a process is defined in part by the current activity of that process.

(A) Each process may be in one of the following states:

- **NEW** → The process is being created.
- **RUNNING** → Instructions are being executed.
- **WAITING** → The process is waiting for some event to occur. (such as I/O completion or reception of a signal).
- **READY** → The process is waiting to be assigned to a processor.
- **TERMINATING** → The process has finished execution.

Diagram



Operation on Process → Process creation
→ Process termination.

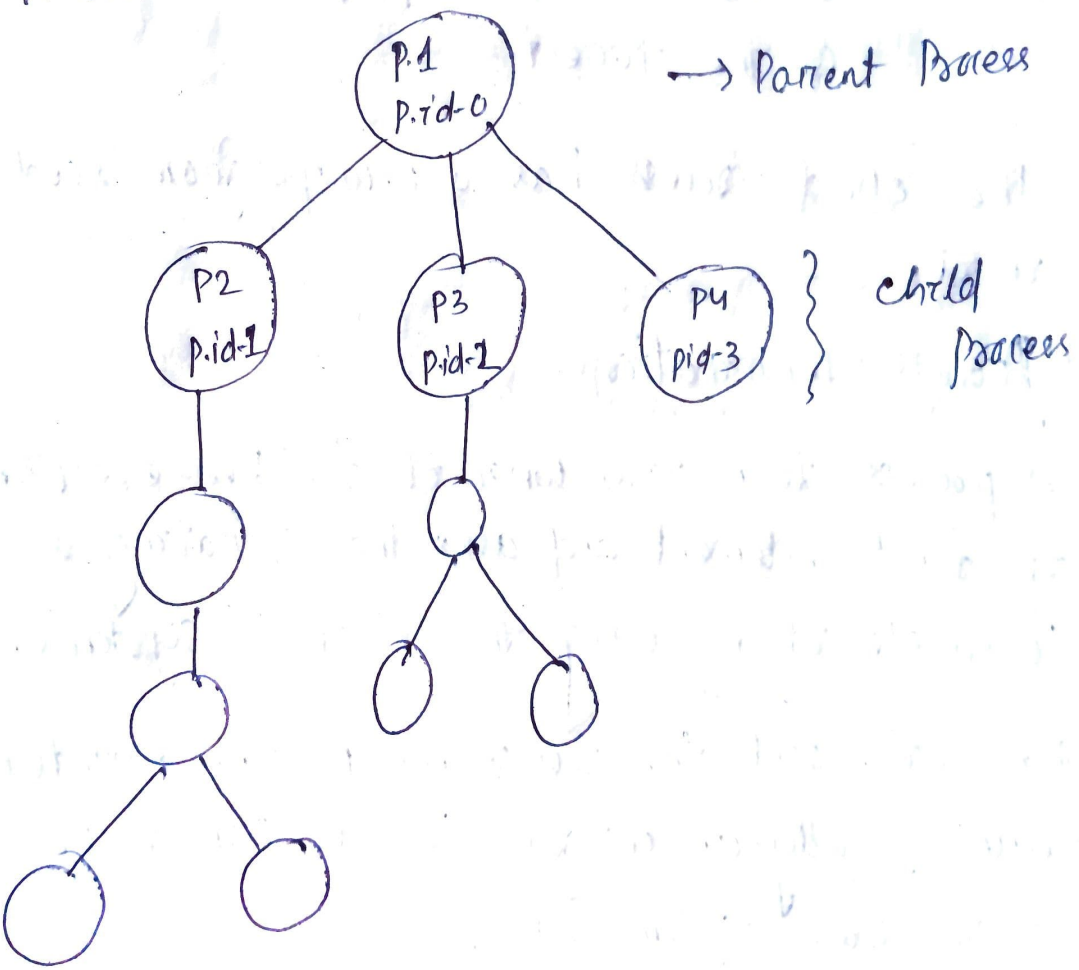
(1) Process Creation

→ A process may create several new processes, via a create-process system call, during the course of execution.

→ The creating process is called a parent process, and the new processes are called the children proces of that Process.

→ Each of these new processes may in turn create other processes, forming a tree of processes.

Figure: A tree of process on typical solaris system.



(*) When a process creates a new process, two possibilities exist in terms of execution:

1. The parent continues to execute concurrently with its children.
2. The parent waits until some or all of its children have terminated.

(*) There are also two possibilities in terms of the address space of the new process:

1. The child process is a duplicate of the parent process. (It has the same program and data as the parent.)
2. The child process has a new program loaded in it.

(2) Process Termination

→ A process terminates when it finishes executing its final statement and asks the operating system to delete it by using the exit() system call.

→ At that point, the process may return a status value (typically an integer) to its parent process (via the wait() system call).

→ All the resources of the process - including physical and virtual memory, open files, and I/O buffers - are deallocated by the operating system.

Termination can occur in other circumstances as well:

- A process can cause the termination of another process via an appropriate system call.
- Usually, such a system call can be invoked only by the parent of the process that is to be terminated.
- Otherwise, users could arbitrarily kill each other's jobs.

* A parent may terminate the execution of one of its children for a variety of reasons, such as these:

- The child has exceeded its usage of some of the resources that it has been allocated. (To determine whether this has occurred, the parent must have a mechanism to inspect the state of its children).
- The task assigned to the child is no longer required.
- The parent is exiting, and the operating system does not allow a child to continue if its parent terminates.

Process Control Block

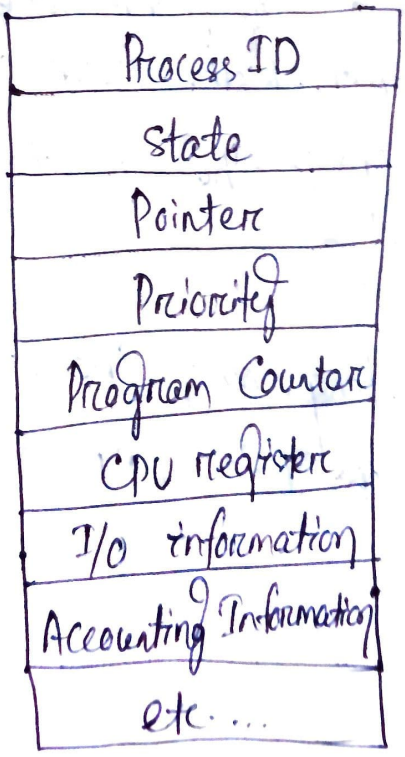
→ A Process control block is a data structure maintained by the operating system for every Process.

→ The PCB is identified by an integer process ID (PID).

→ A PCB keeps all the information needed to keep track of a process as listed below in the table.

→ The architecture of a PCB is completly dependent on operating system and may contain different information in different operating system.

→ Here is a simplified diagram of a PCB -



→ The PCB is maintained for a Process throughout its lifetime, and is deleted once the process

form rnotes.

(1) Process state :- The current state of a process i.e whether it is ready, running, waiting on whatever.

(2) Process Privilege :- This is required to allow / disallow access to system resources.

(3) Process ID :- Unique identification for each of the Process in the operating system.

(4) Pointer :- A pointer to parent process.

(5) Program Counter :- A Program counter (PC) is a pointer to the address of the next instruction to be executed for this process.

(6) CPU register :- Various cpu registers where process 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 ~~information~~ amount of CPU used for process execution, time limits, execution ID etc.