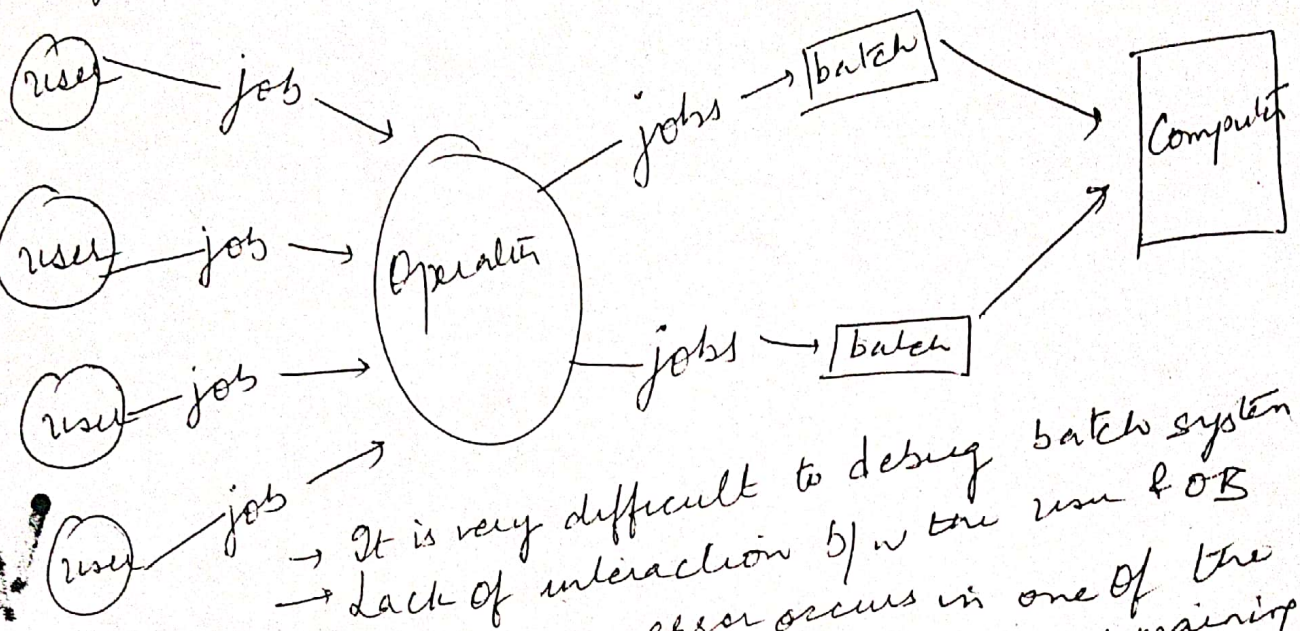
Concept of Operating System: Operating system is a program that acts as an intermediary b/w the user of a computer and the computer h/w. OS provides an environment in which a user can execute programs in a convenient & efficient manner.



→ It is very difficult to debug batch system
→ Lack of interaction b/w the user & OS
→ Suppose an error occurs in one of the jobs of a batch. Then all the remaining jobs get affected i.e. they have to wait until the error is resolved
→ pgms that have require long execution time such pgms can be performed even in the absence of humans

Multiprogramming: It permits the multiple programs to be loaded into the memory and execute the pgm concurrently. A pgm in execution is called a process. The concurrent execution of programs → improving the utilization of system resources and thus enhancing the system throughput

Types of Operating System

Batch operating System → To speed up processing operator batched together jobs with similar needs an ran them through the computer as a group. The programmer leave their programs with the operator.

Memory layout for a batch system

permanently occupied by resident portion of the OS

This area dynamically used to load the transient programs for execution. When one program is over, the new program is loaded into the same area

Multiprogramming OS → Improving system throughput and resource utilisation

Time Sharing Operating System (Multitasking) is a logical extension of multiprogramming. It allows many users to simultaneously share the computer resources. CPU switches rapidly from one user to another, each user is given an impression that he has his own computer while it is actually one computer shared among many users.

Real time Operating Systems It is used when rigid time requirements have been placed on the operation of a processor or the flow of data. RTS has well defined constraints or the system will fail. Heard

Objective — is to provide quick event response time
Meet the scheduling deadlines.

Hard — short term integrity of data — Data file sizes small/medium

Soft — long term integrity of data — large

Operating System Services -:

An operating system provides services to the program and to the users of the programs. It provides an environment for the execution of the program. The services provided by one operating system is different from the another operating system. OS makes the programming task easier.

Services provided by the operating system -:

(1) Program Execution -: OS loads a pgm into memory and executes the pgm. The pgm must be able to end its execution either normally or abnormally.

(2) I/O operation -: Pgm may require any I/O device while running. So OS must provide the required I/O.

(3) File System Manipulation -: Pgm needs to read or write a file. The OS gives the permission to the program for operation on file.

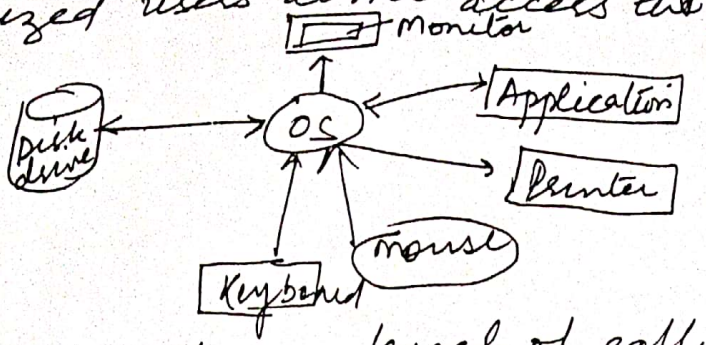
(4) Communication -: Data transfer b/w two ~~programs~~ processes is required for some time. The both processes are on the one computer or on different computers but connected through computer network. Communication may be implemented by 2 methods :- shared memory or message passing.

(5) Error detection -: Error may occur in CPU, I/O device or in the memory. The OS constantly needs to be aware of possible errors. It should take the appropriate action to ensure correct and consistent computing.

OS with multiple users provided the following services -:

(1) Resource Allocation → OS is a resource allocator. OS keeps track of the status of each resource and decides who gets a resource, for how long and when. OS makes sure that different programs running at the same time but do not interfere with each other.

It is also responsible for security, ensuring that unauthorized users do not access the system.



View of OS with components

* An OS is a lower level of software that user programs run on. OS is built directly on the Hardware interface and provides an interface b/w the hardware and the user program. It share characteristics with both h/w and s/w.
* The primary objective of OS is to increase productivity of a processing resource such as computer hardware or users.
* The OS is the first program run on a computer when the computer boots up. The services of the OS are invoked with a system call.

→ Accounting ① Log of each user must be kept. It is also necessary to keep record of which users uses how much and what kinds of computer resources. This log is used for accounting purposes. The accounting data may be used for statistics or for billing. It is also used to improve system efficiency.

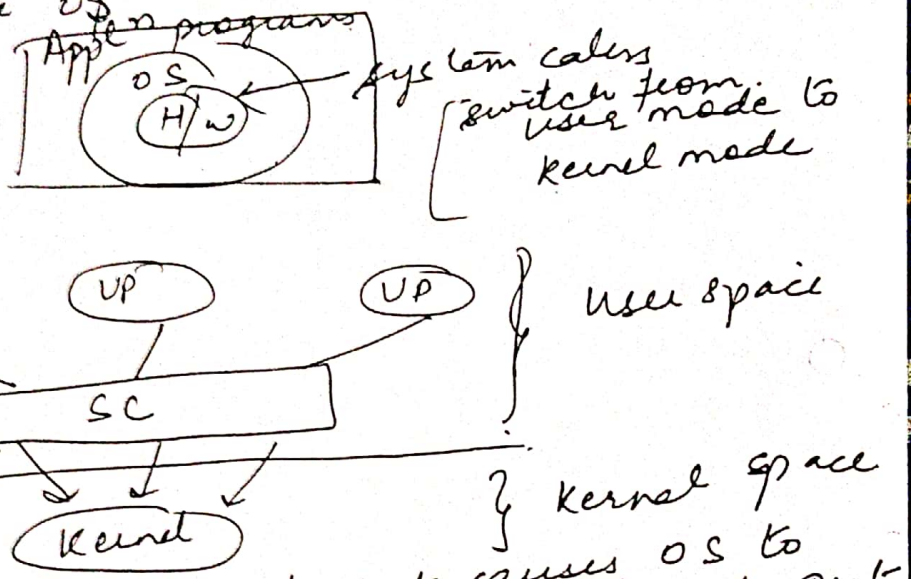
System Calls :- for performing any operation, a user must have to request a service call which is also known as system call.

Ordinary User mode → all user processes are executed

Kernel/System mode → All privileged operations are executed

Heart of OS → The user programs and kernel functions are being executed in their respective space allotted in the main memory partitions

User mode programs need to execute some privileged operations which are not permitted in user mode, user mode programs must use an interface which forms only permitted interface b/w user mode and kernel mode. The interface is called system calls. So system calls is an interface b/w the user programs and the OS.



SC generates an interrupt that causes OS to gain control of the CPU. The OS uses finds out the type of SC and the corresponding interrupt handler routine is executed to perform the

Making a system calls:-

Executing the system calls:-

There is a sequence of steps to execute a system call for this there is a need to pass parameters of system calls to the OS. Methods to pass parameters

(1) Register Method -: where in the parameters are stored in registers of the CPU

(2) If parameters are not in number, compared to the size of registers a block of memory is used and the address of that block is stored in registers

(3) Stack Method -: parameters are pushed on to the stack

Sequence in which SC are executed:

(1) In the user program, the SC is executed, Del- of all its parameters are pushed on to the stack & later on saved in processor registers.
(2) The library procedure for the SC is executed
(3) There is a particular code for every SC by which the kernel identifies which system call function or handler needs to be executed.
(4) Then the library procedure traps to the kernel by executing interrupt instruction. With this interrupt execution, the user mode switches to kernel mode

Types of SC

(1) Process control SCs -: A process is a basic entity in the system. The processes in the system need to be created, deleted and aborted.
fork ( ) — create a process
exit ( ) — Terminate a process
kill ( ) — Terminate a process abnormally.
nice ( ) — Increase a priority of a process.
(2) File Mgmt System calls -: creation, deleting, opening, closing, read, write
create ( ), open ( ) close ( ) read ( ) write ( )
(3) Device Mgmt system calls.
(1) Request of device (2) Release of device.
(4) Information Maintainence SC →
(5) Communication systems calls -:

Multithreading -: A thread is a path of execution within a process. A process can contain multiple threads. Thread is also known as lightweight process. The idea is to achieve parallelism by dividing a process into multiple threads

Process Vs thread - (1) Threads within the same process run in a shared memory space while processes run in separate memory spaces. (2) Threads are not independent of one other like processes.

CPU Scheduling

Multiprogramming :- The objective of multiprogramming is to have some process running at all the times, in order to maximize CPU utilization

Uniprocessor System :- Only one process may run at a time, any other processes must wait until the CPU is free and can be rescheduled.

In a simple computer system, CPU would then sit idle, all this waiting time is wasted. With multiprogramming, we try to use this time productively. Several processes are kept in memory at one time. When one process has to wait, the operating system takes the CPU away from that process and gives the CPU another process. This pattern is scheduling.

Scheduling → the process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

It can not be scheduled

Non Preemptive → when a process switches from the running state to the waiting state or when a process terminates

Under non preemptive, once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.

Preemptive :- Processes are preempted or you can say it can be scheduled

Preemptive :- The scheduling which takes place when a process switches from running state to ready state or from waiting state to ready state

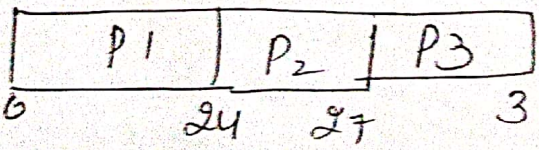
Scheduling Criteria

1. CPU Utilization → The time for which CPU is busy.
2. Throughput → No. of processes executed or completed by CPU in a particular time. All time CPU must be allocated with some process.
3. Burst time → No. of CPU cycles req'd for process to run or execute. (time)
4. Waiting → The time for which the process spends waiting in the ready queue.
5. Turnaround time → (Total time to complete the process)
6. Response Time → The amount of time it takes to start responding, but not the time that it takes to o/p that response.

Objective of CPU scheduling — 1. Maximum CPU utilisation
2. Max Throughput
3. Min waiting time
4. Min turnaround time

FCFS

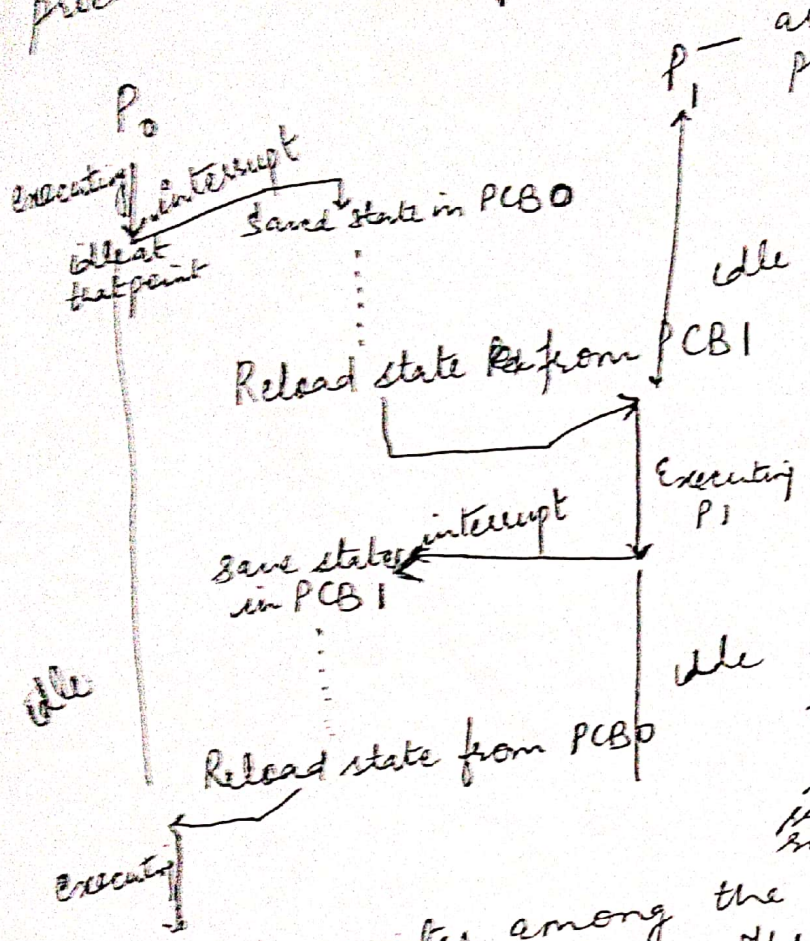
Process	Burst time	waiting time
P1	24	P1 = 0
P2	3	P2 = 24
P3	3	P3 = 27



Average waiting time = 0 + 24 + 27 / 3

= 17 ms

Context switching → Switching from one process to another in a system requires saving state of old process and loading saved state for new process.



Context - It is the stuff on the CPU which needs to be saved so that the CPU can restart execution at the current point at some later time (usually after interrupt)

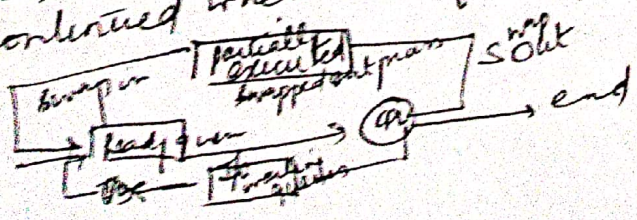
Switching → Switching b/w runnable process remove the running process and replace it with another runnable process

A process migrates among the various scheduling queues throughout its lifetime. The OS must select for scheduling purpose, process from these queues. The selection process is carried out by Scheduler

Long TS - JS → selects processes from the pool and loads them into main memory for execution

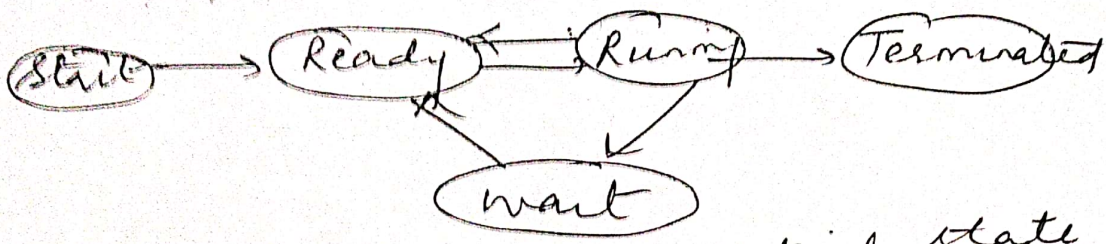
STS → CPU scheduler → selects processes from a pool that contains all processes that are waiting to be executed.

MTS → used to remove process from memory to reduce the degree of multiprogramming. Later the process can be reintroduced into memory and its execution can be continued where it left off



Process → A process is basically a program in execution.

Program → A program is a piece of code which may be single line or millions of lines. When we execute this program it becomes a process which performs all the tasks mentioned in a program



Start → This is the initial state when a process is first started/created

Ready → The process is waiting to be assigned to a processor. 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

Running → Once the process has been assigned to a processor by the OS scheduler

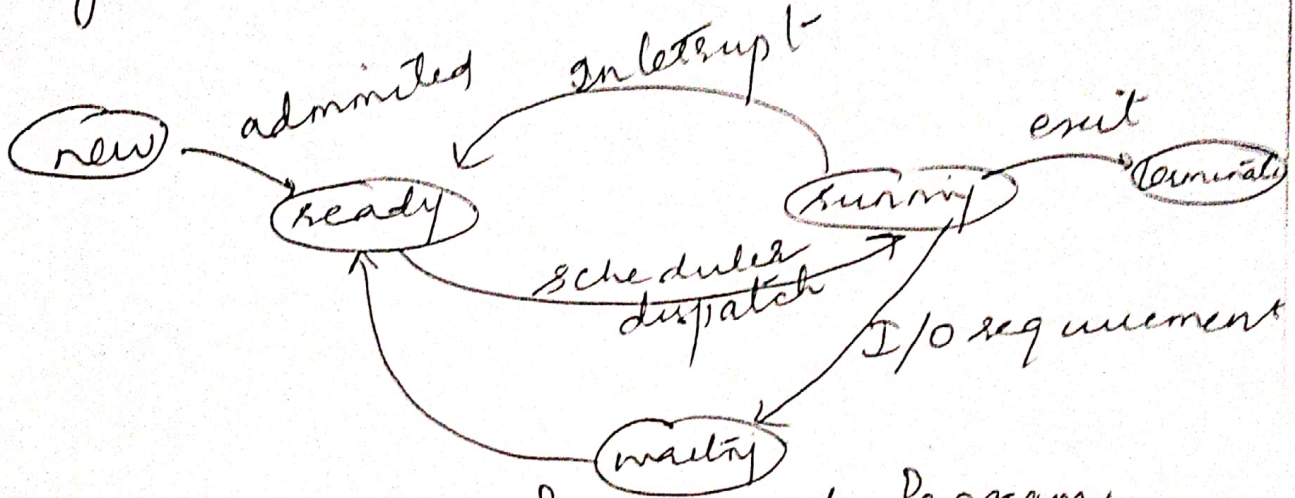
Waiting → If a process needs to wait for a resource such as waiting for user i/p or waiting for a file to become available

Terminated → Once the process finishes its execution, or it is moved to the terminated state where it waits to be removed from the main memory.

PCB → Process Control Block → It is maintained by operating system for every process. It keeps all the information needed to keep track of a process.

Process
Dynamic

Program
Static



Difference between Process and Program

Program is a set of instruction to perform a particular task.

Process is a program in execution

Process	Program
② Process is a active entity	② Program is a passive entity
③ Limited.	③ Longer
④ Process holds resources such as CPU, memory address, disk I/O etc	④ Program is stored in disk. Some files does not require any other resources

Process

1. It is a heavyweight task.
2. Process switching needs interaction with OS
3. In multiple processing environment each process executes the same code but has its own memory & file resources
4. If one process is blocked then no other process execute until the first unblocked
5. Each process operates independently to one another.

Thread

1. It is a lightweight task.
2. Thread switching does not interact with OS
3. All threads can share same set of open files, child processing.
4. While one thread is blocked or waiting, a second thread in same task can run
5. One thread can read write or change another thread data

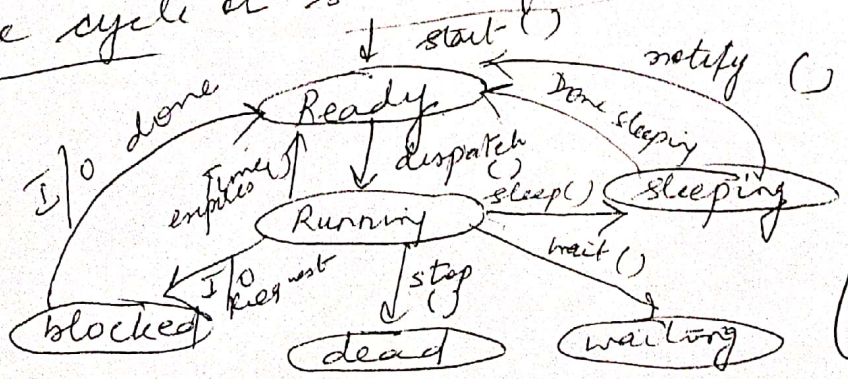
User level thread

1. These are managed by User level library.
2. Fast
3. Context switching is faster
4. If one user level thread blocked then entire process get blocked

Kernel level Thread

1. These threads all managed by OS — we have to use system calls
2. Slow
3. Slow
4. If one kernel level thread blocked, no effect on others

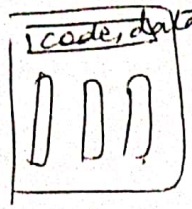
Life cycle or states of a thread

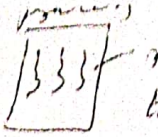


User prgms → This is the highest layer in the layered operating system. This layer deals with the many user prgms and applications that run in an OS such as word processor games, browsers.

Thread → light weight process

It is the basic unit of CPU execution consisting of – Program counter
– thread id
– Stack } to store temporary variables
– set of registers

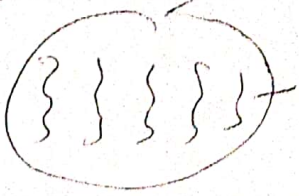




example of Word document

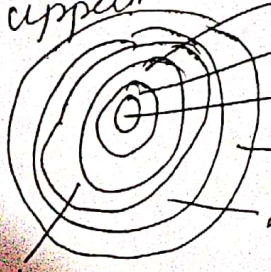
different tasks or threads
- Spell checker is a thread
- Saving
- animations

This thread will share the resources of word document.



Layered Structure of an Operating System –

Layering provides a distinct advantage in an operating system. All the layers can be defined separately and interact with each other as required. Also, it is easier to create, maintain and update the system if it is done in the form of layers. Change in one layer specification does not affect the rest of the layers. Each of the layers interact with the layers that are above and below it. The lowest layer handles the h/w and the uppermost layer deals with the user applications.



6 Six Layers

Layer 1 H/w → This layer interacts with the system hardware and coordinates with all the peripherals devices used such as printer, mouse, keyboard. The H/w layer is the lowest layer in the layered OS architecture

Layer 2 CPU scheduling → It deals with scheduling the processes for the CPU. There are many scheduling queues that are used to handle processes like job queue, ready queue.

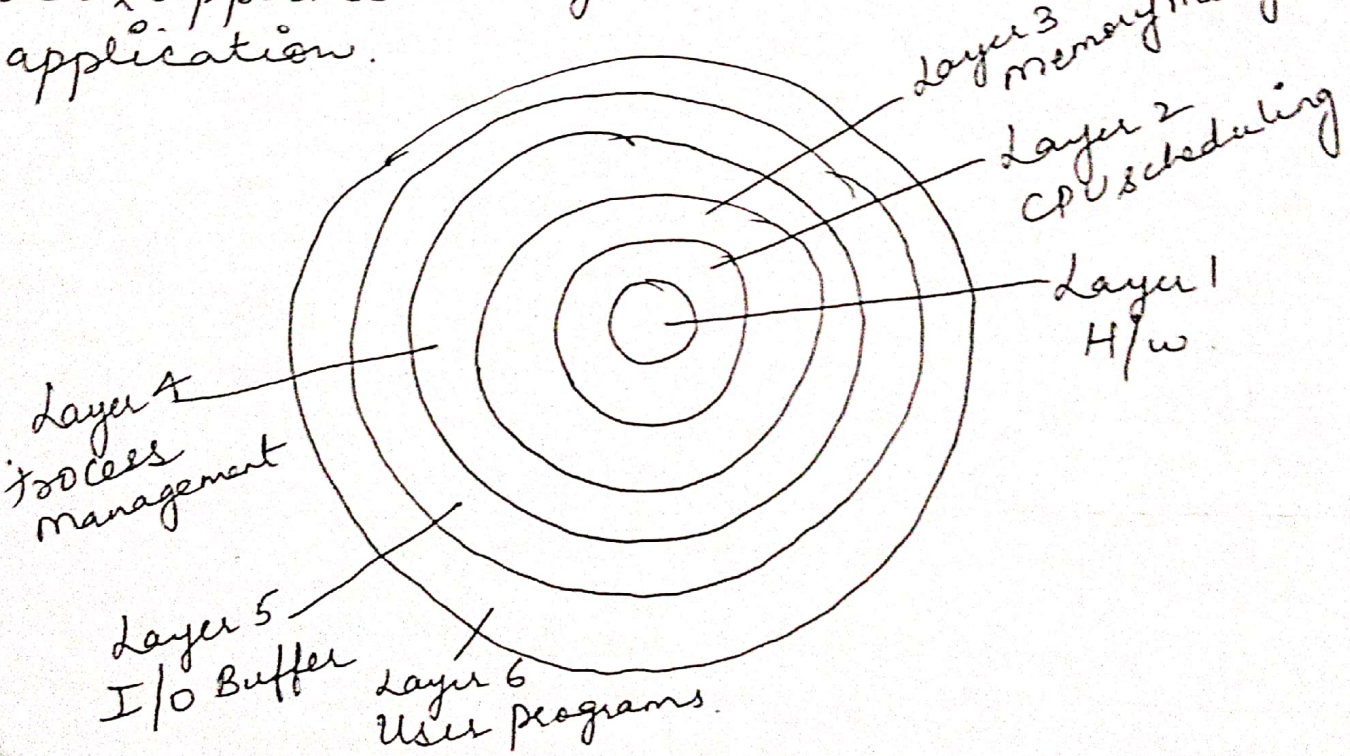
Layer 3 Memory mgmt → It deals with memory and the moving of processes from disk to primary memory for execution and back again. This is handled by the 3rd layer of OS

Layer 4 Process mgmt → This layer is responsible for managing the processes i.e assigning the processor to a process at a time. This is known as process scheduling.

Layer 5 I/O Buffer → I/O devices are very important in the computer systems. They provide users with the means of communicating with the system. This layer handles the buffers for the I/O devices and make sure that they are working correctly.

Layered Structure of an Operating System -: (1)

Layering provides a distinct advantage in an operating system. All the layers can be defined separately and interact with each other as required. Also it is easier to create, maintain and update the system if it is done in the forms of layers. Change in one layer specification does not affect the rest of the layers. Each of the layers in the OS can only interact with the layers that are above and below it. The lowest layer handles the H/w and the uppermost layer deals with the user application.



(1) Layer 1 (H/w) → This layer interacts with the system hardware and coordinates with all the peripherals devices used such as printer, mouse, keyboard. The H/w layer is the lowest layer in the layered OS architecture.

2) Layer 2 (CPU scheduling) → It deals with scheduling the process for the CPU. There are many scheduling queues that are used to handle processes like job queue and ready queue.

3) Layer 3 (memory Mgmt) → It deals with the memory and the moving of processes from disk to primary memory for execution and back again.

4) Process Management (layer 4) → This layer is responsible for managing the processes i.e. assigning the processor to a process at a time. This is known as process scheduling. It includes FCFS, SJF, RR etc.

5) I/O Buffer → I/O devices are important in the computer systems. They provide users with the means of interacting with the system. This layer handles the buffers for the I/O devices and make sure that they are working correctly.

6) User Programs :- This is the highest layer in the layered OS. This layer deals with the many user programs and applications that run in the OS such as word processors, games, browsers etc

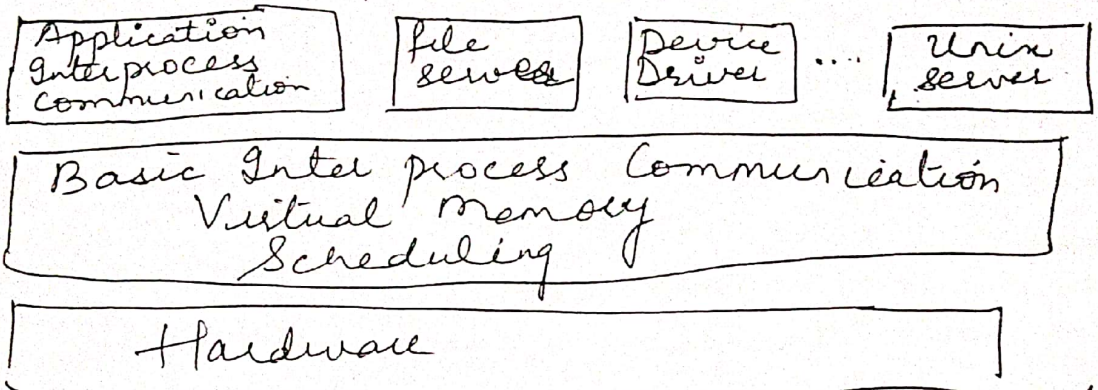
Monolithic OS → The entire operating system works in the kernel space in the monolithic system. This increases the size of the kernel as well as the operating system. This is different from the microkernel system where the minimum software that is required to correctly implement an operating system is kept in the kernel.

Disadvantage

(1) If any service fails in the monolithic kernel it leads to failure of the entire system.

(2) To add any new service, the entire OS needs to be modified by the user.

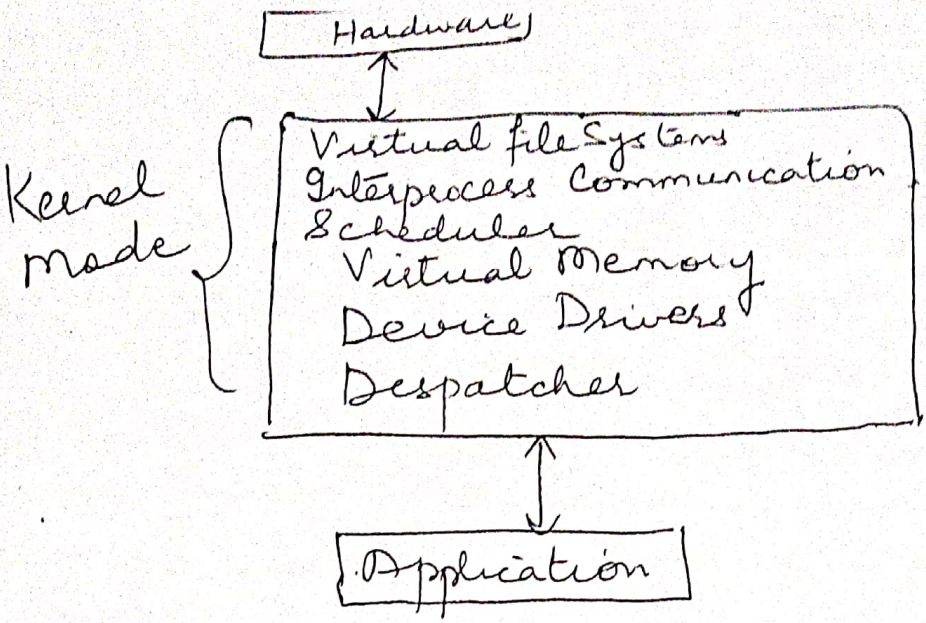
Microkernel OS —: It is the minimum s/w that is required to correctly implement an OS. This includes memory, process scheduling mechanisms and basic interprocess communication.



In the above diagram, the microkernel contains basic requirements such as memory, process scheduling mechanism and basic interprocess communication. The only s/w executing at the priviledged level i.e kernel mode is the microkernel. The other functions of the OS are removed from the kernel mode and run in the user mode. These functions may be device drivers file servers etc. The microkernel make sure that the code can be easily managed because the services are divided in the user space. This means that there is less code running in the kernel mode which results in increased security & stability. Essential Components in a microkernel

A microkernel contains only the core functionalities of the system —:

1) Memory Management mechanisms like address spaces are included in the microkernel. This also contains memory protection features



The kernel provides various services such as memory management, file management, process scheduling etc using function calls. This makes the execution of the OS quite fast as the services are implemented under the same address space.

Difference between Microkernel & Monolithic

1) Microkernel is smaller in size as compared to the monolithic.
2) The microkernel is easily extensible whereas this is quite complicated for monolithic.
3) The execution of microkernel is slower as compared to monolithic.
4) Much more code is required to write a microkernel than monolithic.

Advantages of monolithic OS → (1) The execution of monolithic kernel is quite fast as the services such as memory management, file management, process scheduling etc are implemented under the same address space.
(2) A process runs completely in a single address space in the monolithic.

(2) Processor scheduling mechanisms are also necessary in the microkernel. This contain process and thread schedulers. (5)

(3) Interprocess communication is important - as it is needed to manage the servers that run their own address space

Performance of a Microkernel System → Providing services in a microkernel system are much more expensive than in a normal monolithic system. The service is obtained by sending an interprocess communication message to the server and getting one in return. This means a context switch or a function call if the drivers are implements as processes respectively. So performance can be complicated in microkernel systems and may lead to some problems.

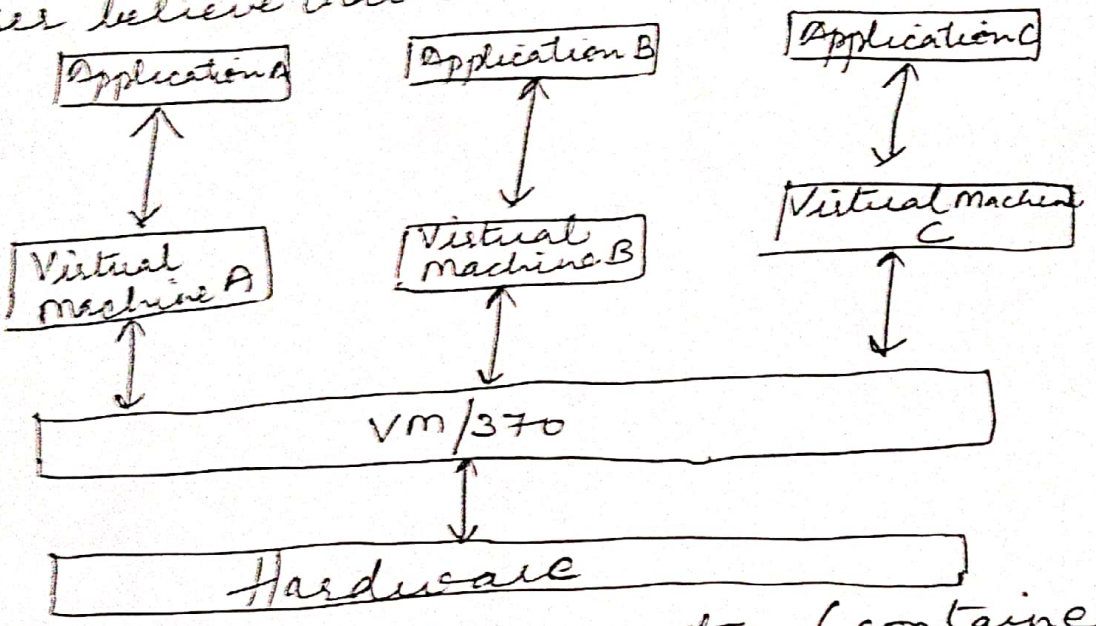
Benefits of Microkernels →

(1) Microkernels are modular and the different modules can be replaced, reloaded, modified, changed etc as required. This can be done without even touching the kernel

(2) Microkernels are quite secure as only those components are included that would disrupt the functionality of the system otherwise.

(3) Microkernels contain fewer system crashes as compared to monolithic. Also the crashes that occur can be handled easily due to the modular structure of microkernels

VIRTUAL MACHINE (VM) :- The fundamental idea behind a virtual machine is to abstract the hardware of a single computer (CPU, memory) into several different execution environments, thereby creating the illusion that each separate execution environments is running its own private computer. The term virtual machine indicates a machine, which does not physically exist and yet makes the user believe that there is a machine.



This virtual machine monitor (contained in VM/370) runs on the actual Hardware and performs the multiprogramming functions.

Implementation :- ① Although the VM concept is useful, it is difficult to implement. Much work is required to provide an exact ~~copy~~ duplicate of the underlying machine. Remember that underlying machines has 2 modes :- user mode & and kernel mode. The virtual machine Software can run in kernel mode, since it is operating system. The virtual machine itself can execute in only user mode. Consequently, virtual user mode and virtual kernel mode, both of which run in physical user mode. Those actions that can cause a transfer from user mode to kernel mode on a real machine.

must also cause a transfer from virtual kernel mode on a virtual machine.

(2) Such a transfer can be accomplished as follows: When a system call, for example, is made by a program running on a virtual machine in virtual user mode.

Benefits — In this environment, there is complete protection of various system resources. Each virtual machine is completely isolated from all other virtual machines, so there is no protection problems. At the same time, there is no direct sharing of resources. Two approaches to provide sharing have been implemented. First, it is possible to share a minidisk and thus to share files modeled after a physical shared disk but is implemented by software. Second, it is possible to define a network of virtual machines, each of which can send information over the virtual communication n/w.

(2) Virtual machine system is a perfect vehicle for operating system research and development.

Q What is the main advantage for an OS designer of using a virtual machine architecture? What is the main advantage for a user?

Ans The system is easy to debug and security problems are easy to solve. Virtual machines also provide a good platform for operating system research since many different operating systems may run on one physical system.