KTU NOTES

The learning companion.

KTU STUDY MATERIALS | SYLLABUS | LIVE NOTIFICATIONS | SOLVED QUESTION PAPERS

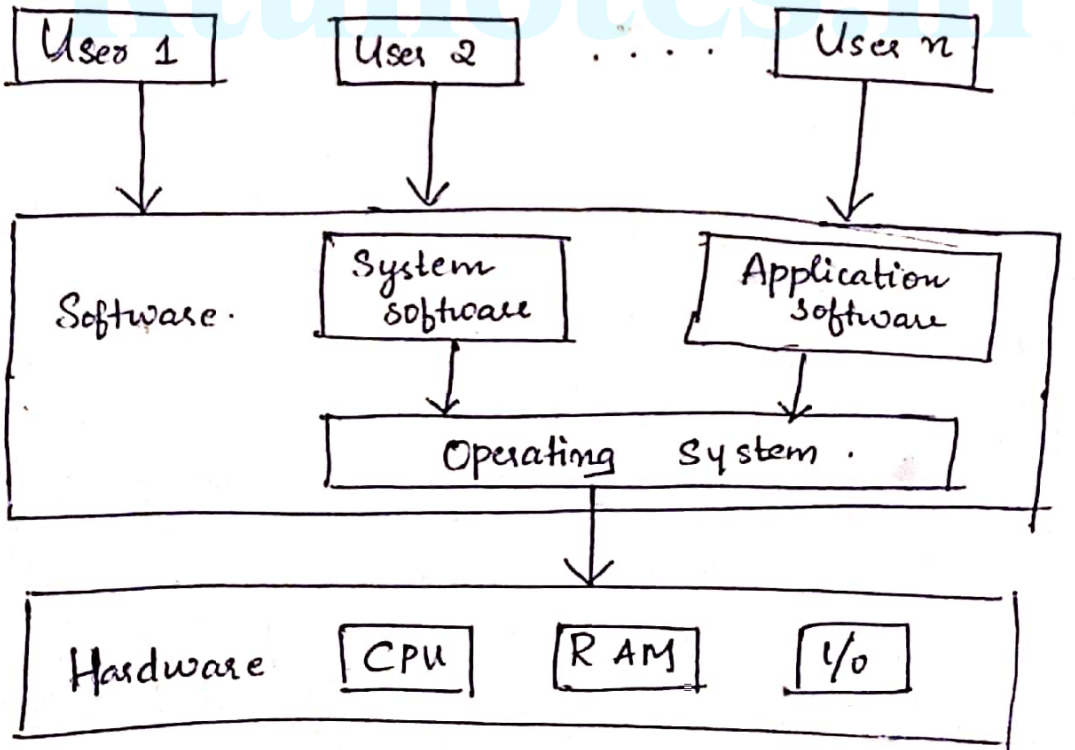
Website: www.ktunotes.in

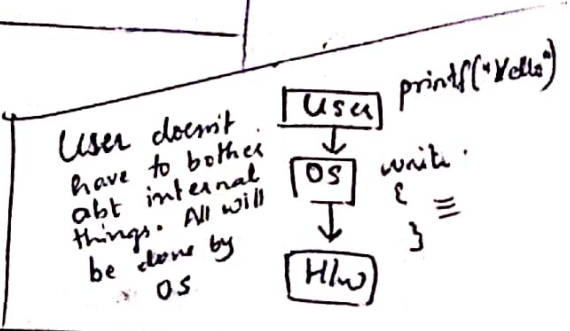
OPERATING SYSTEM

MODULE - 1

* An OS is a pgm that acts as an interface b/w user & computer h/w and controls the execution of all kinds of pgms.
* OS is a system software.
* Software – set of tested pgms with documentation.
  (user manual - how to use s/w)

S/w
- Application s/w.
  - designed for a specific task.
    - Eg: Windows Media player.
- System s/w
  - Provides platform for Appln s/w to run.
  - Operates computer h/w.





Functions of OS

1. Memory Management.

- Management of primary mly or main mly.
- for a pgm to be executed, it must be in the main mly.

* An OS does the foll. activities for Mly mgmt:
  - Keeps track of primary mly ie, what part of it are in use by whom, what part are not in use.
  - In multi programming, the OS decides which process will get memory when & how much.
  - Allocates mly when a process requests it to do so.
  - Deallocates mly when a process no longer needs it or has terminated.

2. Processor Management.

- OS decides which process gets the processor when & for how much time. This ftn is called process scheduling.

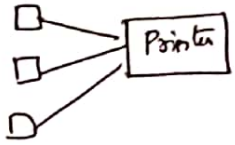
* Functions.
  - Keeps track of processor & status of process. The pgm responsible for this task is known as traffic controller.
  - Allocates the processor (CPU) to a process.
  - Deallocates processor when a process is no longer required.

3. Device Management

- Manages device communication via their respective drivers.

* Functions

- Keeps tracks of all devices. Pgm responsible for this task is known as I/O controller.
- Decides which process gets the device when & for how much time.
- Allocates device in the efficient way.
- Deallocates devices



4. File Management

- A file s/m is organized into directories for easy navigation & usage.

* Functions

- Keeps track of information, location, uses, status etc. The collective facility are known as file s/m.
- Decides who get the resources.
- Allocates the resources.
- Deallocates the resources.

5. Security & Protection

By means of password & similar other techniques, it prevents unauthorized access to pgms & data.

6. Job Accounting

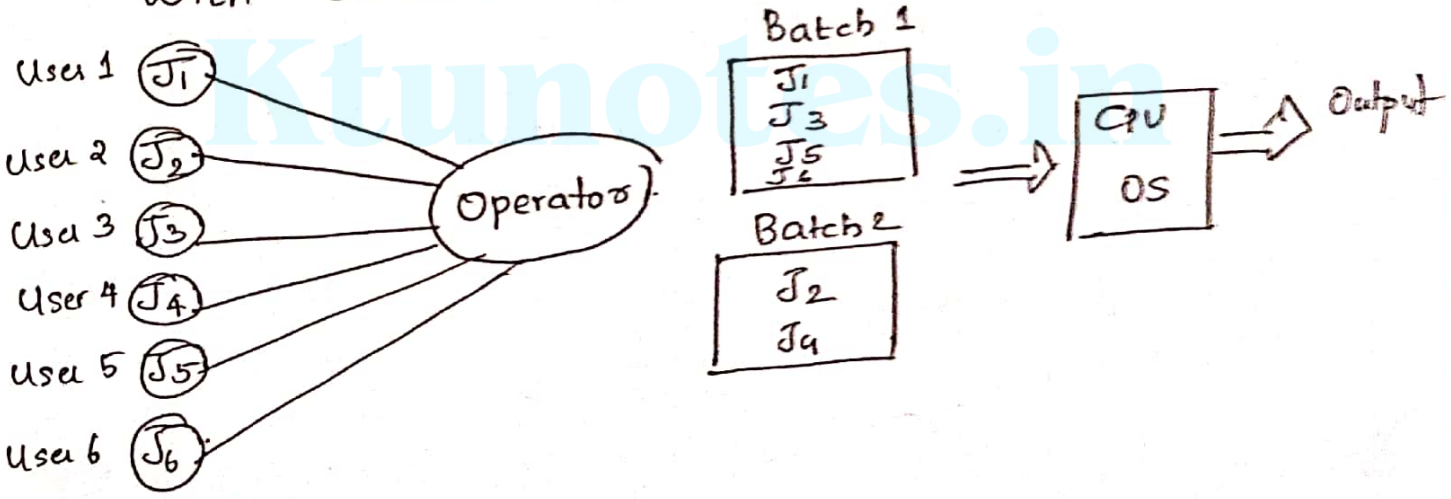
Keeping track of time & resources used by various jobs & users.

7. Error detecting aids — Production of dumps, traces, error msg & other debugging & error detecting aids.

TYPES OF OPERATING SYSTEM.

1. Batch Operating System.

* The users of a batch OS do not interact with the computer directly.
* Each user prepares his job on an off-line device like punch cards & submits it to the computer operator.
* To speed up processing, jobs with similar needs are batched together & run as a group.
* The programmers leave their programs with the operator & the operator then sorts the pgm with similar requirements into batches.



Problems.

* Lack of interaction btw user & the job.
* CPU is often idle, because the speed of I/o devices is slower than the CPU.
* Difficult to provide the desired priority.

2. Multiprogrammed System

* Multiprogramming increases CPU utilization by organizing jobs so that CPU always has one to execute.
* Single user cannot keep either CPU or I/O devices busy all the time.
* So went for multiprogrammed system.
* Idea of multiprogramming.
  - ↳ OS keep several jobs in memory simultaneously.
  - ↳ This set of job is subset of jobs kept in Job Pool.
  - ↳ Since there is limitation in the no. of jobs placed in memory compared to job pool.
  - ↳ OS picks & begins to execute one of the jobs in the memory.

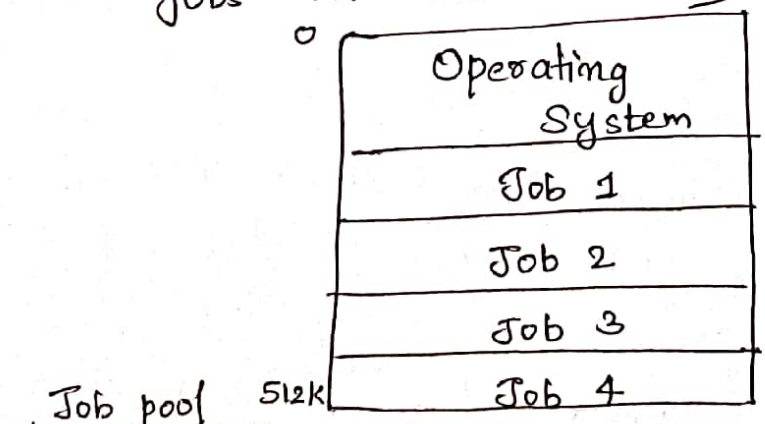
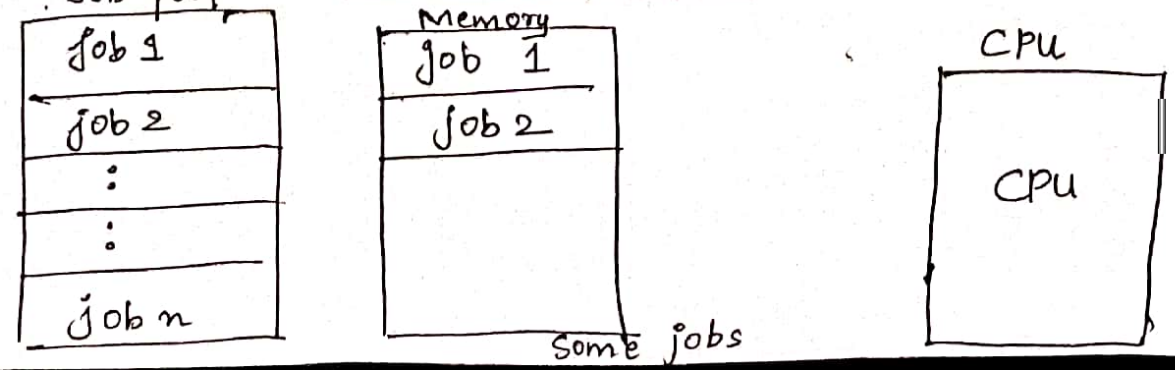


fig: M/y layout for a multi-programming system.



3.* When a job may have to wait for some task such as an I/O operation.

↳ In non-multiprogrammed system, the CPU would sit idle; else the OS simply.

↳ In multiprogramming system, the OS simply switches to & executes another job.

↳ When that job has finished then it is switched to another job & so on.

↳ Eventually, the first job finishes waiting & gets the CPU back.

* Multiprogramming is the first instance where the OS must make decisions for the users.

* Multipoogrammed OS are fairly sophiscated.
  - All the jobs that enters the system are kept in the job pool.
  - This pool consists of all processes residing on disk awaiting allocation of main memory.
  - If several jobs are ready to be brought into memory & if there is not enough room for all of them, then the system must choose among them.
  - Making this decision is job scheduling.
  - If several jobs are ready to run at the same time, then the system must choose among them.
  - Making this decision is CPU scheduling.

3. Multiprocessor System.

* Multiprocessor system are also called as parallel systems or tightly coupled systems.
* Multiprocessor system have more than one processor in close communication, sharing the computer bus, the clock & sometimes memory & peripheral devices.

Advantages.

(i) **Increased throughput.**

By increasing the no. of processors more work done in less time.

(ii) **Economy of scale**

Multiprocessor system can save more money than multiple single processor system → because they share peripherals, mass storage & power supplies.

(iii) **Increased Reliability.**

- If functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slow it down.
- If we have 10 processors & one fails, then the rest of 9 processors must pick up a share of the work of the failed processor.
- This ability to continue providing services proportional to the level of surviving hardware is called **graceful degradation.**

4. Time Sharing System.

* Major disadvantage of batch system & multiprogrammed system was that there is no user interaction.
* Time sharing (or multitasking) is a logical extension of multiprogramming.
* In Time sharing system, the CPU executes multiple jobs by switching among them.
* The switches occurs so frequently that the users can interact with each program while it is running.
* Time shared OS allows many users to share the computer simultaneously.
* As the system switches rapidly from one user to the next, each user is given the impression that the entire computer system is dedicated to their use.
* Each user is given a time slice for executing his job, job continues until time slice ends.
* Time shared OS uses CPU scheduling & multi-programming to provide each user with a small portion of a time shared computer.

5. Real Time Systems.

* A real time system is used when rigid time requirements have been placed on the operation of a processor.
* Processing must be done within the defined constraint or the system will fail.
* Real time systems come in two flavours:
  - ✓ Hard real time system
  - ✓ Soft real time system

(i) Hard Real time System.

* Hard real time system guarantees that critical tasks be completed on time.
* Therefore all delays in the system be bounded, from the retreival of stored data to the time that it takes the OS to finish any request made of it.

(ii) Soft Real Time System

* In Soft Real Time system a critical real-time tasks gets priority over other tasks & retain that priority until it completes.

*

6. Distributed OS.

* Distributed systems uses multiple central processors to serve multiple users.
* The processors communicate with one another through various communication lines. These are referred as loosely coupled system / distributed system.

Advantages.

- With resource sharing facility, a uses at one site may be able to use the resources available at another.
- Speed up the exchange of data with one another via e-mail.
- If one site fails in distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on host computer.
- Reduction of delays in data processing.

OPERATING SYSTEM SERVICES.

OS services are provided for the convenience of the programmer, to make the programming task easier.

(i) Program Execution.

* System must be able to load a pgm into memory & to run that program.
* The program must be able to end its execution either normally or abnormally.

(ii) I/O operation.

* A running pgm may require I/O.
* This I/O involve a file or an I/O device.
* For efficiency & protection, users usually cannot control I/O devices directly.
* OS provide a means to do I/O.

(iii) File System Manipulation.

* Programs need to read & write files.
* Programs also need to create & delete files by name.

(iv) Communication

* One process needs to exchange information with another process.
* Communication can occur in two ways :-
  - ✓ Between processes that are executing on the same computer.
  - ✓ Between processes that are executing on different computer system connected through a network.
* Communications are implemented using shared memory or by message passing in which packets of information are moved between processes by the OS.

(v) Error Detection :

* OS constantly needs to be aware of possible errors.
* Error may occur in the :
  - CPU
  - Memory hardware (memory error / power failure)
  - I/o device (Lack of paper in printer, connection failure on n/w)
  - User pgm
* For each type of error, the OS should take appropriate action to ensure correct & consistent computing.

(vi) Resource allocation.

* When multiple users are logged on the system or multiple jobs are running at the same time, resources must be allocated to each of them.
* Different types of resources are managed by the OS - CPU cycle, main mly & file storage.

(vii) Accounting

* Need to keep track of which user use how many & which kinds of computer resources.
* Record keeping may be used for accounting or for accumulating usage statistics.

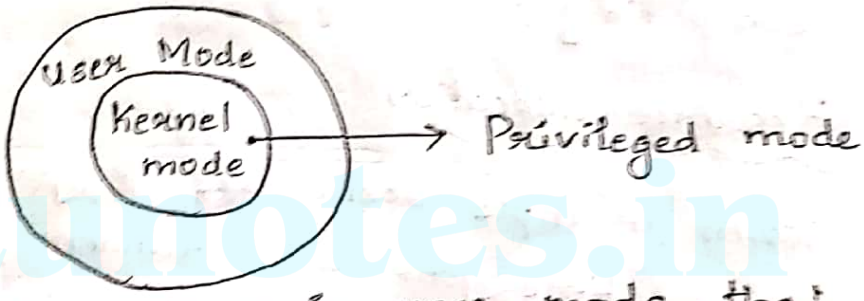
(viii) Protection

* Protection involves ensuring that all access to system resources is controlled.

* Security of the system from outsiders is important.
* Security is done by authenticating the user by means of password.

System Call

* System calls provide an interface to the services made available by an OS.
* Two modes of operation in which pgm can execute:
  - User mode
  - Kernel mode



* If pgm is executing in user mode, that pgm doesn't have direct access to memory, h/w & resources.

If pgm is executing in kernel mode, that pgm has direct access to memory, h/w & resources ie, in privileged mode.

* If pgm is executing in kernel mode & if that pgm happens to crash during pgm execution then the entire system will crash / halt.

If pgm is executing in user mode & if that pgm happens to crash during pgm execution then the entire system will not crash / halt.

* So it is safe to execute pgms in uses mode, but it may need to have access to m/y, hlw & resources
* for that, it ~~makes~~ makes a call to OS that it needs access to resources.
~~At that~~ for an instant, the pgm is switched from uses mode to kernel mode so that it can use that resources. This switching is called context switching.
* The call that the pgm makes to access the resources is called System call.
* System call is the programmatic way in which a computer pgm requests a service from the kernel of the OS.

TYPES OF SYSTEM CALLS

System calls can be grouped roughly into 5 major categories:

1. Process Control.
2. File Manipulation.
3. Device Management
4. Information Maintenance
5. Communications.

① Process Control

* end, abort.
* load, execute
* create process, terminate process.
* get process attribute, set process attribute
* wait for time
* wait event, signal event
* allocate & free m/y.

② File Manipulation.

* Create file, delete file.
* open, close
* read, write, reposition.
* get file attributes, set file attributes.

③ Device Manipulation.

* request device, release device.
* read, write, reposition.
* get device attributes, set device attributes
* logically attach or detach devices.

④ Information Maintenance.

* get time or date, set time or date.
* get system data, set system data.
* get process, file or device attributes.
* set process, file or device attributes.

⑤ Communication.

* Create, delete communication connection.
* send, receive messages.
* transfer status information.
* attach or detach remote devices.

OPERATING SYSTEM STRUCTURE.

Structure specifies the way these components are interconnected & melded into a kernel.

1. Simple Structure.

* Commercial system do not have a well-defined structure.
* OS started as small, simple & limited system & then grew beyond original scope.
* MS-DOS was designed & implemented to provide the most functionality in the least space (because of the limited hardware on which it ran).
* So in MS-DOS it was not divided into modules carefully.

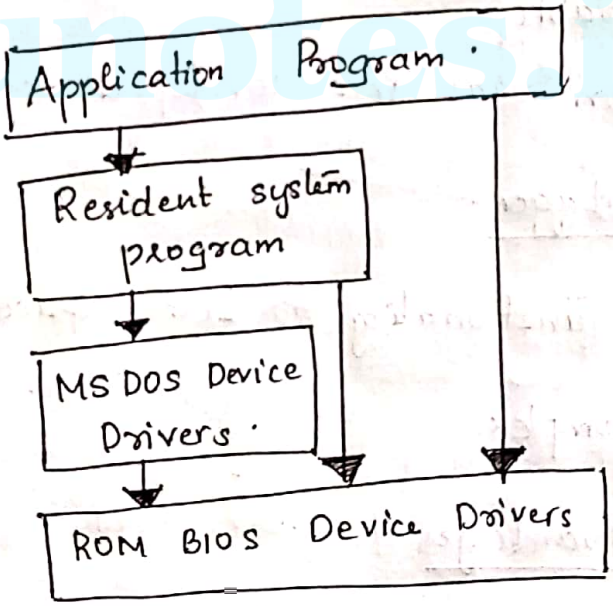


fig: MS DOS structure

* Application Pgm — Programs where users directly interact with them.

Eg: To write document → applm is MS word.

* System Pgm (Eg: File Manipulation)

To create/delete file, system pgm gives system call, manipulation of file done.

User Interface ↔ System Pgm ↔ System Call

* MS DOS Device Drivers - Computer Program that operates or controls hardware devices attached to computer.

* ROM BIOS Device Drivers
  - This can be considered as hardware
  - BIOS - Basic I/O system.
  - Stored in ROM on Mother board.
  - Used to start the system.

Eg: Bank - Cash withdrawal
Person → Write check → Gives to employee to approve → Gives to cashier → Get money from locker → Person.

- Appln pgm direct access to h/w, malicious or harmful pgm can take advantage of this.
- Vulnerable.
- Reason was less developed technology at that time.

Advantages.

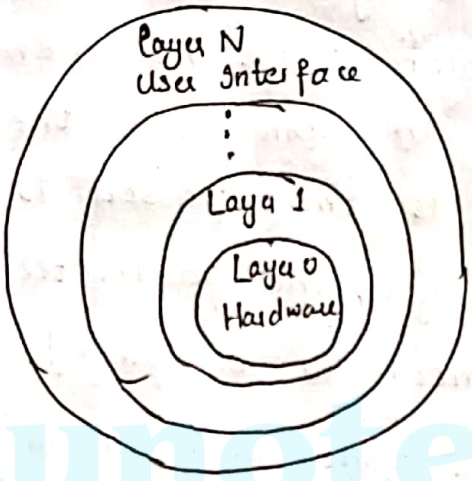
* More functionality in Less Space
* Less Complex

DisAdvantages.

* Direct Access to hardware made it vulnerable to malicious programs.
* If application pgm would get crashed, whole system got affected.

2. Layered Approach.

* One method of modularization is the layered approach, in which the OS is broken up into a number of layers (or levels) each built on top of lower layer.
* Bottom layer (layer 0) is the hardware
* Highest layer (layer N) is the user ~~Interface~~ Interface



* A typical OS layer (layer M) consists of data structures & set of routines that can be invoked by higher level layers.
* Layer M in turn, invokes operation on lower-level layer.
* Layers are selected such that each uses functions & services of only lower-level layers.
* Design & implementation of the system are simplified where the system is broken down into layers.
* A layer does not need to know how these operations are implemented, it needs to know only what these operations do.
* Each layer hides the existence of certain data-structures, operations & hardware from higher-level layer.

Disadvantages

1. Layered approach involves careful definition of the layers, because layers can use only those layers below it.
2. Layered approach tend to be less efficient than other types.

When a user program executes an I/O operation, it executes a system call that is trapped to the I/O layer, which calls the memory management layer, which in turn calls the CPU scheduling layer, which is then passed to the hardware.

At each layer, the parameters may be modified, data may need to be passed.

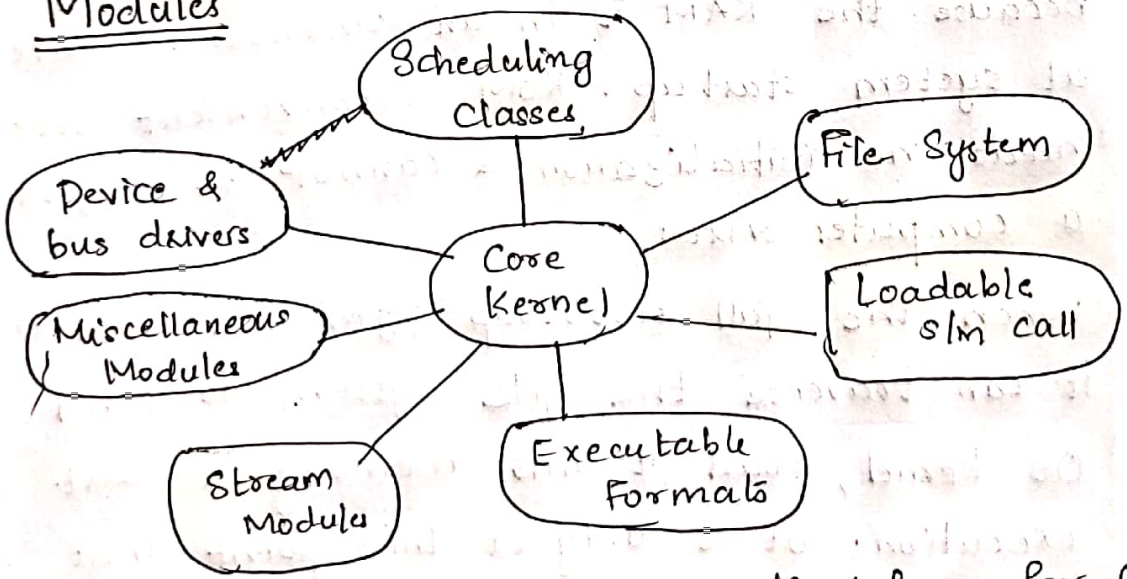
Each layer adds overhead to the system call. Resulting in a system call that takes longer time than on a non-layered system.

3. Microkernels:

* This method structure the OS by removing all non-essential components from the kernel & implementing them as system & user level pgms. Result is a smaller kernel.
* Main function of micro kernel is to provide a communication facility between the client pgm & the various services running in user space.
* Communication is provided by message passing.

* If a client program wishes to access a file, it must interact with the file server.
* Client pgm & the service never interact directly, they communicate indirectly by exchanging messages with the microkernel.
* Benefit of the microkernel approach - ease of extending the OS.
  All new services are added to user space & do not require modification of the kernel.
* The resulting OS is easier to port from one hardware design to another.
* Microkernel provides more security & reliability since most services are running as user - rather than kernel processes.

4. Modules



* This is the best current methodology for OS design.
* Core Kernel will only have core functionalities of kernel & other functionality are present in the form of modules which will be loaded into kernel either at boot time or at run time.

* More flexible than layered system, as any module can call anyother module directly via core kernel.
* More efficient than a microkernel, because modules do not need to invoke message passing inorder to communicate as they can be loaded into kernel either at boot time or run time.

SYSTEM BOOT.

* The procedure of starting a computer by loading the kernel is known as booting the system.
* On most computer systems, a small peece of code known as the bootstrap pgm or bootstrap loader locates the kernel.
* The program is in the form of read-only-m/y (ROM) because the RAM is in an unknown state at system start up. ROM is convenient because it needs no initialization & cannot be infected by a computer virus.
* When the full bootstrap pgm has been loaded, it can traverse the file system to find the OS kernel, load it into memory & start its execution. It is only at this point that the system is said to be Running.