Rutvi Patel
SOCET-CE-E
Roll no. 15

1. Introduction. (10%)

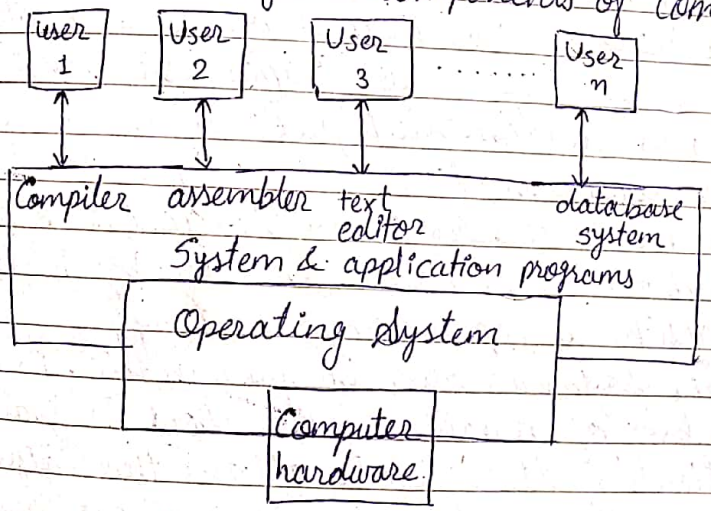
- Computer System overview
- Architecture, Goals & Structure of OS.
- Basic functions
- Interaction of OS and hardware architecture, System call
- Batch, Multiprogramming.
- Multitasking, timesharing, Parallel, distributed and real time OS.

⇒ What is OS ?

An Operating System is a program that manages a computer's hardware. It also provides a basis for applications programs and acts as an intermediary between the computer user and computer hardware.

⇒ Computer System & Role of OS in Computer System

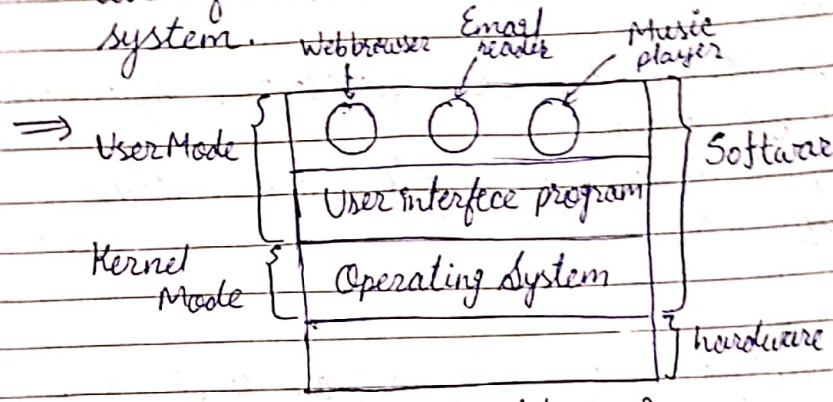
- Abstract view of the components of Computer System.



A computer system can be divided roughly into four parts : (i) hardware (ii) OS (iii) Application Programs (iv) Users

→ The hardware : the CPU, the memory and I/O devices provides the basic resources for the system.
→ The application programs such as word processor, spreadsheets, compiler and web browsers - define the ways in which these resources are used to solve user's computing ~~systems~~ problems.
→ The OS controls the hardware and coordinates its use among the various application programs for users.

→ The operating system provides the means for proper use of these resources in the operation of computer system.

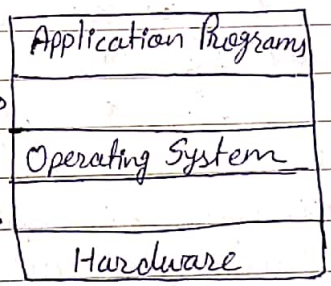


Where the OS fits in ?

→ Hardware lies at the bottom and it consists of chips, boards, disks, keyboard, monitor and similar physical objects.
→ On the top of hardware is the software.
→ Most computers have two modes of operations
① User Mode ② Kernel Mode or privileged Mode (or Supervisor Mode)
→ OS (the most fundamental piece of software) runs in kernel mode.
→ In this mode it has complete access to all the hardware and can be executed by instructions the machine is capable of executing.
→ The other softwares run in user mode, in which only subset of machine instructions is available.
→ In particular, those instructions that affects control of the machine or do I/O are forbidden to user-mode programs.
→ Finally, above the system programs are the application programs. These programs are purchased or written by the users to solve their particular problems, such as word processing, spreadsheets, web browsers or music players.
→ To hide complexity of hardware, an OS is provided. It consists of a layer of software that hides the hardware & gives the programmer a more convenient set of instructions to work with.

⇒ The Operating System as an Extended Machine.

→ Architecture of a computer at machine level language is called primitive.
→ Architecture of computer consists of instruction set, memory organization, I/O devices and bus structure.
→ ~~The~~ primitive architecture is awkward to program as specially I/O ~~devices~~ operations.
→ Users don't want to be involved in programming of storage devices.
→ Operating System provides a simple, high level abstraction such that these devices contain a collection of named files.
→ Such files consists of useful piece of information like a digital photo, e-mail messages or web pages.
→ OS provides a set of basic commands or instructions to perform various operations like read, write, open, search, close etc.
→ OS hides the complexity of hardware and presents a beautiful interface to the user.



→ Just as the OS shields the programmer from the disk hardware & presents a simple file oriented interface, it also conceals a lot of unpleasant business concerning interrupts, timers, memory management and other low level features.
→ The abstraction offered by the operating system is simpler & easier to use than that offered by the underlying hardware.
→ In this view the function of OS is to present the user ~~and easier to use~~ with the equivalent of an extended machine or virtual machine that is easier to work with than the underlying hardware.
→ The OS provides a variety of services that programs can ~~be~~ obtain using special instructions called system calls.

⇒ Give the view of OS as a Resource Manager.

→ The concept of an OS as providing abstraction to application program is a top down view.
→ Alternatively, bottom up view holds that the OS is there to manage all pieces of a complex system.
→ A computer consists of a set of resource such as processor, memories, timers, disks, printers etc.
→ The OS manages the resources and allocates them to specific programs.
→ As a resource manager, OS provides controlled allocation of the processor, memories, I/O devices among various programs.
→ The processor itself is a resource and the OS decides how much processor time should be given for the execution of a particular user program.
→ Operating System also manages memory and I/O devices when multiple users are working.
→ The primary task of OS is to keep the track of which programs are using which resources, to grant resource requests, to account for usage and to resolve conflicting requests from different programs and users.
→ An OS is a controlled program. A control program controls the execution of user programs to prevent errors and improper use of computer.
→ Resource management includes multiplexing resource in two ways: in time & in space.
→ When a resource is time multiplexed, different programs or users take turns using it. First one of them gets to use the resource, then another and so on.
→ Eg. CPU and printer are time multiplexed resources. OS decides who will use it and for how long.
→ The other kind of multiplexing is space multiplexing, instead of the customers taking turns, each one gets part of the resource.
→ For Eg. both primary & secondary memories are space multiplexed. OS allocates them to user programs and keeps the track of it.

⇒ History of Operating System:

→ OS have been evolving through the years.

- First Generation (1945-1955)
  - First Generation of OS was invented during 1945-1955.
  - Vaccum Tubes and Plugboards were used in the first generation of OS.
  - Input/output timing was too much (Around 1 day) and the speed of process & execution was too low.
  - During this generation computers were generally used to solve simple mathematical calculations,
  - OS were not necessarily needed.
  - All programming was done in absolute machine level language, often by wiring up plugboards to control the machine's basic functions.

- Second Generation (1955-1965)
  - The second generation was developed during 1955-1965.
  - The vaccum tubes and plugboards were replaced by transistors and punch cards.
  - Input/Output timings were reduced to 15-20 mins from 1 Day. (Speed became higher comparatively)
  - Batch Processing was invented during this period of time.
  - These machines were known as mainframes and were locked in AC rooms with the operating team.
  - In this generation computers were used for scientific and engineering calculations.

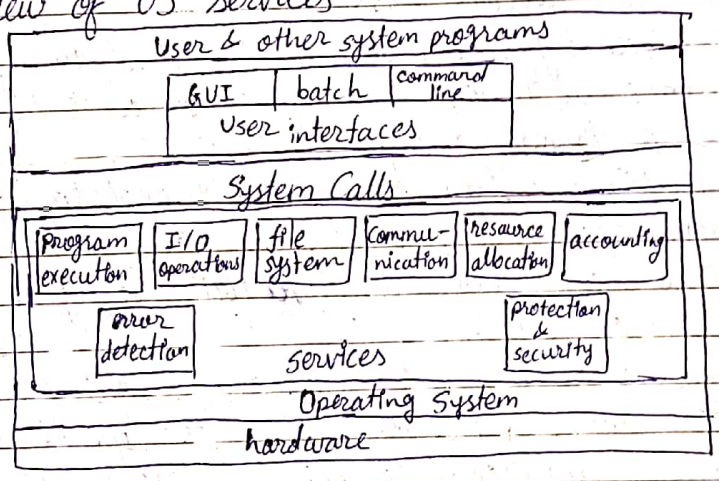
- Third Generation (1965-1980)
  - Third generation of OS was developed during 1965-1980
  - OS 360 was invented by IBM during this generation
  - Multiprogramming was introduced in the 3rd Gen.
  - I.C.s were used in OS
  - Size became smaller & performance speed became faster comparatively.

• Fourth Generation (1980 - Present)
→ LSI (Large Scale Integrated Circuits) and VLSI (Very Large Scale ICs) were started to in use.
→ Computers became cheaper and small in size.
- In LSI & VLSI, chips contained thousands of transistors on a ~~thousand~~ square centimeter of silicon.

⇒ Services / Tasks of OS :

① ~~Program~~ ~~Development~~
An Operating System provides an environment for the execution of programs. It provides certain services to programs and to the users of those programs.
→ The OS services are provided for the convinence of the programmer to make programming.

• A view of OS services:



② User Interface : Almost all the OS has UI. UI has several forms. Such as
- Command line interface (CLI) : uses text commands & a method for entering them.
- Batch interface : commands and directives to control those commands are entered into files & those files are executed.
- Graphical User interface (GUI) : The interface is a window system with a pointing device to direct I/O, choose from menus & make selections & keyboard to enter texts.

② Program Execution : The system must be able to load a program into memory and to run that program. The program must be able to end its execution, either normally or abnormally (indicating error).

③ I/O operations : A running program may require I/O, which may involve a file or an I/O devices. For specific devices, special functions may be desired. For efficiency and protection, users usually can't control I/O devices directly. Therefore, the OS must provide a mean to do I/O.

④ File system manipulation : The file system is of particular interest. They also need to create and delete them by name, search for a given file and list file information. Finally some OS include permissions management to allow or deny access to files or directories based on file ownership. Many OS provides a variety of file systems, sometimes to allow personal choice and sometimes to provide specific features or performance characteristics.

⑤ Communication : There are many circumstances in which one process needs to exchange information with another process. Such communication may occur between processes that are executing on the same computer or between processes that are executing on the same computer systems tied together by a computer network. Communication may be implemented via shared memory in which two or more processes read & write to a shared section or message passing, in which packets of information in predefined formats are moved between processes by the OS.

⑥ Error detection : The OS needs to be detecting and correcting error constantly. Errors may occur in CPU & memory hardware in I/O devices and in the user program. For each type of error, the OS should take the appropriate action to ensure correct & consistent computing.

Sometimes, it has no choice but to halt the system. At other times, it might terminate an error-causing process or return an error code to a process for the process to detect and possibly correct.

(7) Resource Allocation : When there are multiple users or multiple jobs running at the same time, resource must be allocated to each of them. The OS manages many different types of resources. CPU cycles, main-memory, file storage etc. may have special allocation code, whereas others (Eg. I/O devices) may have much more general requests and release code. For instance in determining how best to use the CPU, OS have CPU scheduling routines that take into account the speed of the CPU, the jobs that must be executed, the no. of register available and other factors. They may also be routines to allocate printers, USB storage drives and other peripheral devices.

(8) Accounting : We want to keep track of which users use how much and what kinds of computer resources. This record keeping may be used for accounting (so that users can be billed) or simply for accumulating usage statistics. Usage statistics may be a valuable tool for researchers who wish to reconfigure the system to improve computing services.

(9) Protection and security : The owners of information stored in a multiuser or network computer system may want to control use of that information. When several separate processes execute concurrently it should not be possible for one process to interfere with the others or with the OS. Protection involves ensuring that all access to system resource is controlled. Security of the system from outsiders is also important. Such security starts with requiring each user to authenticate himself/herself by means of password to gain access to system resource. It extends to defending external I/O devices, including network adaptors, from

invalid access attempts and to recording all such connections for detecting break-ins. If a system is to be protected and secure, precautions must be instituted throughout it. A chain is only as strong as its weakest link.

⇒ Types of Operating System.

① Mainframe OS:
- → At the end are the OS for mainframes, those room-sized computers still found in major corporate data centers which differs from PC in terms of I/O capacity
- → A mainframes are making something of a comeback as high end Web servers, servers for large scale electronic ~~devices~~ ecommerce sites and servers for business-to-business transactions.
- → It offers 3 kind of services:
  - (i) Batch System: It is one that processes routine jobs without any interactive user present. Eg. Sales reporting for a chain of stores
  - (ii) Transaction processing system: It handles large numbers of small requests. Each unit of work is small but the system must handle hundreds or thousands per secong. Eg. airline reservation
  - (iii) Timesharing System: It allow multiple remote users to run jobs on the computer once, i.e. querying a big database.
- → Eg. of mainframe OS: OS/390, a descendant of OS/360.
- → Mainframe OS are gradually being replaced by UNIX variants such as Linux.

② Server OS:
- → Server OS run on servers, which are either very large PCs, Workstations or even mainframes.
- → They serve multiple users at once over a network and allow the users to share hardware & software resources. Servers can provide print service, file-service or Web service.
- → Typical Server OS are Solaris, Free BSD, Linux & Windows Server 201x.

③ Multiprocessor OS.

→ An increasingly common way to get major league computing power is to connect multiple CPUs in a single system.

→ Depending on precisely how they are connected and what is shared, these systems are called parallel computers, multicomputers or multiprocessors.

→ They need special OS, but often these are variations on the server OS with special features for communication, connectivity and consistency.

→ Windows 8, Linux and so many other OS run on multiprocessor.

④ Personal Computer OS.

→ Modern ones all support multiprogramming, often with dozens of programs start up at boot time.

→ Their job is to provide good support to a single user.

→ They are widely used for word processing, spreadsheets & Internet access. Eg. Linux, Free BSD, Windows Vista and Macintosh OS.

→ ~~Personal Computer OS are so widely known that probably little introduction is needed.~~

⑤ Handheld Computer OS

→ Continuing on down to smaller and smaller system, we come to handheld computers or PDA (Personal Digital Assistant) which is a small computer that fits in a shirt pocket and performs a small number of functions such as an electronic address book and memo pad.

→ PDAs & mobile phones have essentially merged, differing mostly in size, weight and user interface.

→ Almost all of them are based on 32 bit CPUs with protected mode & run a sophisticated OS.

→ The OS that run on these handhelds are increasingly sophisticated with the ability to handle telephony, digital photography & other functions.

→ Two of the most popular OS for handhelds are Symbian OS and Palm OS.

(6) ~~Embeded~~ Embedded OS.

→ Embedded systems run on the computers that control devices that are not generally thought of as computers and which donot accept user-installed software.
→ Typically examples are microwave ovens, TV sets, cars, DVD, cell phones, MP3 players.
→ The main property which distinguishes embedded systems from handhelds is the certainly that no untrusted software will ever run on it.
→ You can't download new app to oven-all the software is in ROM.
→ This means that there is no need for protection bet" apps, leading to some simplification.
→ Systems such as QNX and VxWorks are popular in this domain.

(7) Sensor Node OS

→ Networks of tiny sensors nodes are being deployed for numerous purposes. These nodes are small battery powered computers with built in radios that communicate with each other and with a base station using wireless communication.
→ These sensor networks are used to protect the perimeters of buildings, guard national borders, detect fires in forestes etc. Sensors have limited power & must work for long periods of time unattended outdoors, frequently in environmentally harsh conditions.
→ Each sensor node is a real computer with a CPU, RAM, ROM & one or more environmental sensors.
→ It runs a small, but real OS usually one that is event driven, responding to external events or making measurements periodically based on an internal clock.
→ As with embedded systems, all the programs are loaded in advance, users don't suddenly start programs they downloaded from the Internet, which makes the design much simpler.
→ TinyOS is a well known OS for a sensor node.

→ Advantages :→ Easy to Use → User Friendly →Quick response time

→ Disadvantages : If any problem affects the OS, you may lose all the contents which have stored already
→ Unwanted user can use your own system in case if proper security options are not available.

(8) Real-Time OS
→ These systems are characterized by having time as a key parameter.
→ Real time OS has well defined, fixed time constraints, processing must be done within define constraints or system will fail.
→ Types of Real time OS:
- Hard real time OS:
  - Many of these are found in industrial process control, avionics & military and similar application areas.
  - These system must provide absolute guarantees that a certain action will occur be a certain time.
- Soft real time OS:
  - Missing on occasional deadline, while not desirable is acceptable and does not cause any permanent damage.
  - Digital audio, digital telephone & multimedia systems fall into this catagory.

→ An Eg. of Real time OS is e-Cos.

(9) Smart Card OS:
→ The smallest OS run on smart cards, which are credit card sized devices containing a CPU chip.
→ They have very ~~serve~~ severe processing power and memory constraints.
→ Some of them can handle only a single function such as electronic payment but others can handle multiple functions on the same card.
→ Often these are proporietary systems.

(10) Multitasking OS
→ A single tasking system can only run one program at a time, while a multitasking OS allows more than one program to be running in concurrency.
→ This is achieved by time-sharing, where the available processor time is divided between multiple processes.
→ Multi-tasking may be characterized in preemptive ~~multitask~~ and

co-operative types.
→ In preemptive multitasking, the OS slices the CPU ti
and dedicates a slot to each of the programs.
→ Unix like OS such as Solaris & Linux as well as non-
like such as Amiga OS supports preemptive multitaskin
→ Cooperative multitasking is achieved by relying on e
process to provide time to the other processes in a
defined manner.
→ 16 bit version of Microsoft Windows used cooperative mu
tasking & 32 bit version of both Windows NT and Win9
used preemptive multitasking.

(11) Distributed OS : → Advantages: → Reduction of delays in data processing
→ Better service to customers
→ Reduction of the load on the host computer.
→ Speedup the exchange of data without going via
→ A distributed OS manages a group of distinct
computers and makes them appear to be a single
computer.
→ The development of network computers that could b
linked and communicate with each other gave rise
to distributed computing.
→ Distributed computations are carried out on more
than one machine. When computers in a group wo
in cooperation, they form a distributed system.

(12) Multiprogramming OS
→ Sharing the processor, when two or more programs
reside in memory at the same time, is referred as
multiprogramming.
→ Multiprogramming assumes a single shared processor.
→ Multiprogramming increases CPU utilization by organizi
jobs so that the CPU always has one to execute.
→ Multiprogramming OS monitors the state of all active
programs and system resources using memory
management programs to ensures that the CPU is neve
idle, unless there are no jobs to process.

(13) Parallel OS :
→ Parallel OS are the interface bet" parallel computers and
the applications that are executed on them.
→ They translate the hardware's capabilities into concepts usable

by programming language.
→ The architecture of the software is often a unix-based platform, which allows it to coordinate distributed loads between multiple computers in a network.
→ Parallel OS are able to use software to manage all of the different resources of the computers running in parallel, such as memory, caches, storage, space and processing power.
→ Parallel OS also allow a user to directly interface with all of the computers in the network.

(14) ~~Time sharing OS~~ → Advantages: Move much of the work of the operator to the computer. → Increase performance since it was possible for job to start as soon as the previous job finished.
→ Disadvantages: Large Turnaround time
→ More difficult to debug program.

(14) Batch OS
→ The users of batch OS don't interact with the computer directly.
→ Each user prepares his job on an off-line devices like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group.
→ Thus, the programmers left their programs with the operator. The operator then sorts programs into batches with similar requirements...

continue...

(15) Interactive OS
→ In an interactive OS, the user interacts directly with the OS to supply commands and data as the application program executes and the user receives the result of processing immediately.
→ The user is in direct two way communication with the computer.

→ The operator then loaded a special program which read the first job from magnetic tape & run it.
→ The O/P was written onto a second magnetic tape, instead of being printed.
→ After each job finished, the OS automatically read the next job from the tape & began running it.
→ When the whole batch was done, the operator removed the I/P & O/P tapes, replaced the input tape with the next batch & brought the O/P tape for offline printing.
→ With the use of this type of OS, the user no longer has direct access to

(16) Time-Sharing OS

→ Time Sharing is a logical extension of multiprogramming.
→ Multiple jobs are executed simultaneously by switching the CPU back and forth among them.
→ The switching occurs so frequently that the users can't identify the presence of other users or programs.
→ Users can interact with his program while it is running timesharing mode.
→ Processor's time is shared among multiple users. An interactive or hands on computer system provides online communication between the user & the system
→ A time shared OS uses CPU scheduling and multiprogramming to provide each user with a small portion of a time shared computer.
→ Each user has at least one separate program in memory.
→ A time shared OS allows many users to share computer simultaneously. Since each action or command in a time shared system tends to be short, only a little CPU time is needed for each user.

Advantages:
- → Easy to use.
- → User friendly
- → Quick response time.

Disadvantages:
- → If any problem affects the OS, may lose all the contents which have stored already.
- → Unwanted user can use your own system in case if proper security options are not available.

⇒ Objective of Operating System

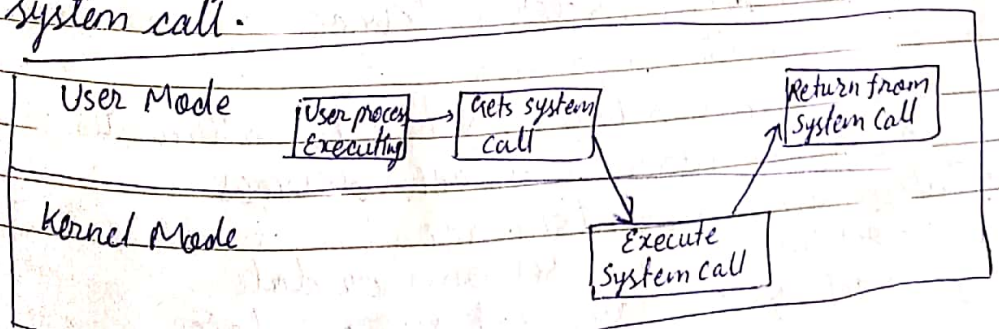
- To make the computer system convenient to use in an efficient manner
- To hide the details of the hardware resources from the users.
- To provide users a convenient interface to use the computer system.
- To act as an intermediary between the hardware and its users, making it easier for the users to access and use other resources.
- To manage the resources of a computer system.
- To provide efficient & fair sharing of resources among users & programs.

⇒ Shell.

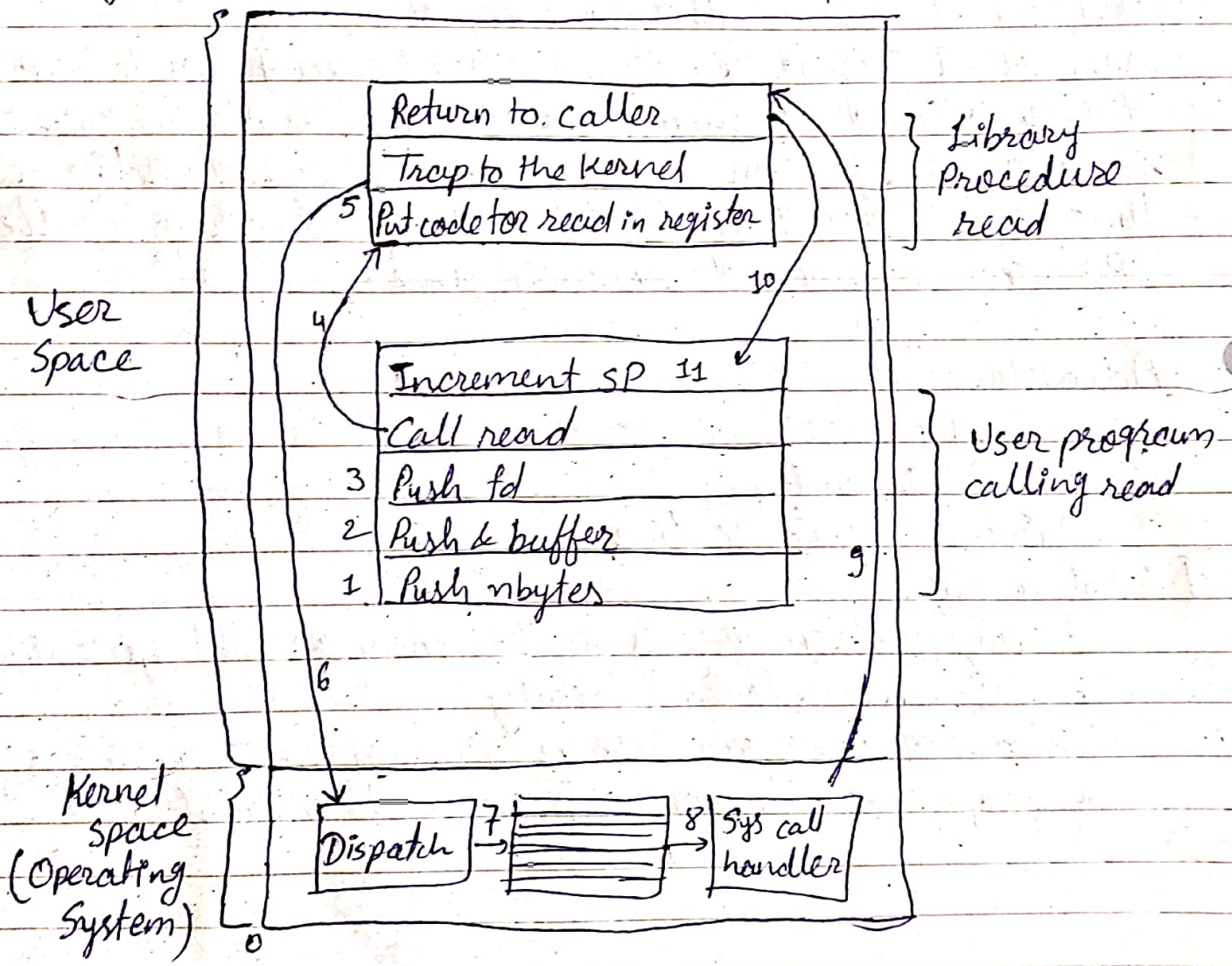
→ Some operating systems includes the command interpreter in the kernel.
→ Others such as Windows and UNIX, thread the command interpreter as a special program that is running when a job is initiated or when a user first log on (on interactive system).
→ On system with multiple command interpreters to choices from, the interpreters are known as shells.
→ For Eg. UNIX and Linux systems, a user may choose among several different shells, including the Bourne shell, C shell, Bourne Again shell, Korn shell and others.
→ Third party shells and free user written shells are also available.
→ Most shells provide similar functionality, and a user's choice of which shell to use is generally based on personal preference.
→ User can check what type of shell he/she is using by using command : echo $SHELL.

⇒ System Calls.

→ The interface between a process and an OS is provided by system calls.
→ In general, system calls are available as assembly language instructions.
→ They are also included in the manuals used by the assembly level programmers.
→ System calls are usually made when a process in user mode requires access to a resource.
→ Then it requests the kernel to provide the resource via system call.



⟹ System calls. (Example)



Steps in making the System call read (fd, buffer, nbytes).

→ As can be seen from this diagram, the processes execute normally in the user mode until a system call interrupts this.
→ Then the system call is executed on a priority basis in the kernel mode.
→ After the execution of the system call, the control returns to the user mode and execution of user processes can be resumed.
→ In general, system calls are required in the following situations:
- If a file system requires the creation or deletion of files. Reading and Writing from files also requires a system call.
- Creation and management of new processes.
- Network connections also require system calls. This includes sending and receiving packets.
- Access to a hardware devices such as a printer, scanner etc. requires a system call.

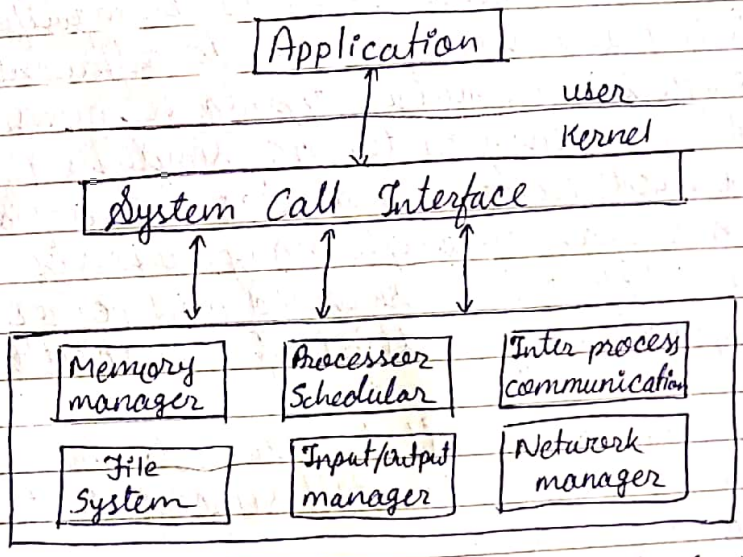
→ Types of System Calls.
- Process control
  - → end, abort → load, execute
  - → create process, terminate process
  - → get process attributes, set process attributes
  - → wait for time
  - → wait event, signal event
  - → allocate and free memory.
- File management
  - → create file, delete file → open, close
  - → read, write, reposition
  - → get file attributes, set file attributes
- Device management
  - → request device, release device
  - → read, write, reposition
  - → get drive attributes, set drive attributes
  - → logically attach or detach devices
- 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

- Communication
  → create, delete communication connection
  → send, receive messages
  → transfer status information
  → attach or detach remote devices.
- Protection
  → set, get permissions
  → allow, deny users.

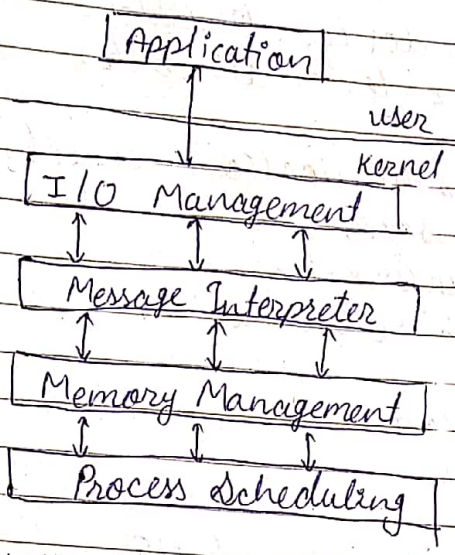
⇒ Architecture ~~and Structure~~ of OS.

① Monolithic



→ The core software of an OS are collectively known as the Kernel. The kernel has unrestricted access to all of the resources on the system.
→ In early monolithic system, each component of the OS was contained within the kernel, could communicate directly with any other component and had unrestricted system access.
→ While this made the OS very efficient, it also meant that errors were more difficult to isolate and there was a high risk of damage due to erroneous or malicious code.

(2) Layered



→ The modularity of layered operating system allows the implementation of each layer to be modified without requiring any modification to adjacent layers. Although this modular approach imposes structure and consistency on the OS, simplifying debugging and modification, a service request from a user process may pass through many layers of system software before it is serviced and performance compares unfavourably to that of a monolithic kernel.

→ Also, because all layes still have unrestricted access to the system, the kernel is still susceptible to errant or malicious code.

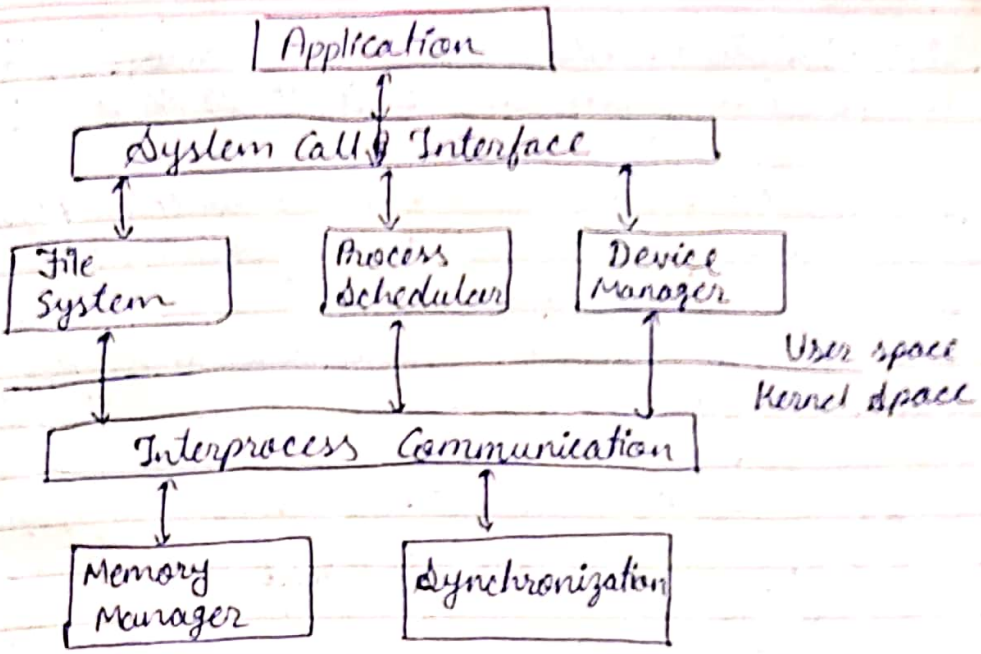
→ Many OS including Windows & Linux implement some level of layering.

(3) Micro-Kernel

→ Micro-kernel architecture includes only a very small number of services within the kernel is an attempt to keep it small and scalable.

→ The services typically include low-level memory management, inter-process communication and basic process synchonisation to enable processes to cooperate.

→ In microkernel designs, most OS components, such as process management & device management execute outside the kernel with a lower level of system access.



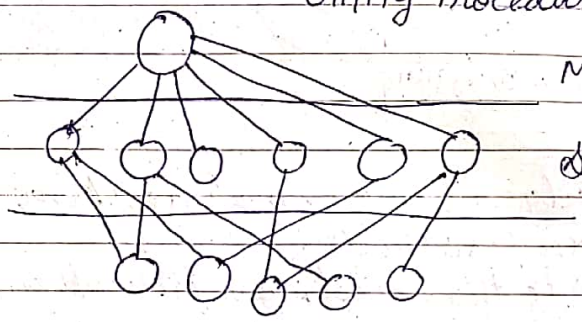
→ Microkernels are highly modular, making them extensible, portable and scalable.
→ OS components outside the kernel can fail without causing the OS to fall over.
→ Once again, the downside is an increased level of inter-module communication which can degrade system performance.

⇒ Structure of OS

① Monolithic System :
- → The OS is written as a collection of procedures each of which can call any of the other ones whenever it needs to.
- → When this technique is used then each procedure in the system has a well-defined interface in terms of parameter & results & each one is free to call any of the other ones. If the latter provides some useful computation that the former needs.
- → Monolithic system provides a basic structure for the OS.
  - Main program that invokes the requested service procedure
  - Set of service procedures that carry out the system calls
  - Set of utility procedures that help the service procedure.

→ In Monolithic System model, there is one service procedure for each system call, that takes care of it.
→ The utility procedures do jobs that are needed by several service procedures, such as fetching the data from the user programs.
→ This vision of the procedures into the following three layers. → Main Procedure,
Service Procedure,
Utility Procedure.

→

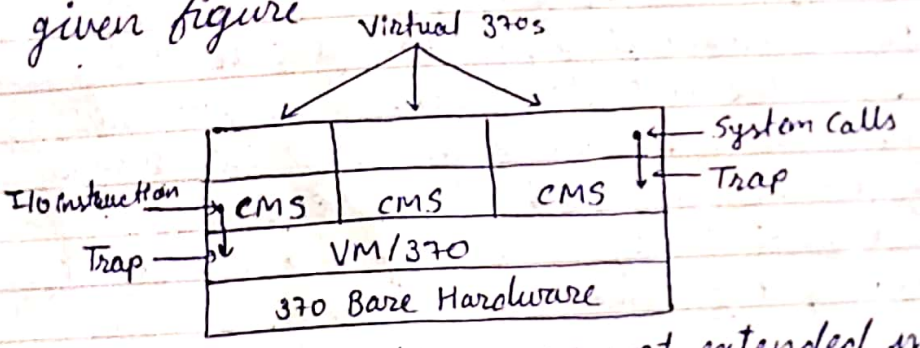


② Layered
→ This system has 6 layers

Layer	Function	Discription
0	Processor allocation and multiprogramming	This layer dealt with allocation of the processor, switching bet" processes when interrupts occured or timers expired
1	Memory & drum management	It did the memory management
2	Operator - process communication	It handled the communication between each process and operator console
3	I/O management	This layer took care of managing the input/output devices and buffering the information streams to and ~~form~~ from them.
4	User programs	On this layer, user programs were found.
5	The operator	On this layer, the system operator process was located.

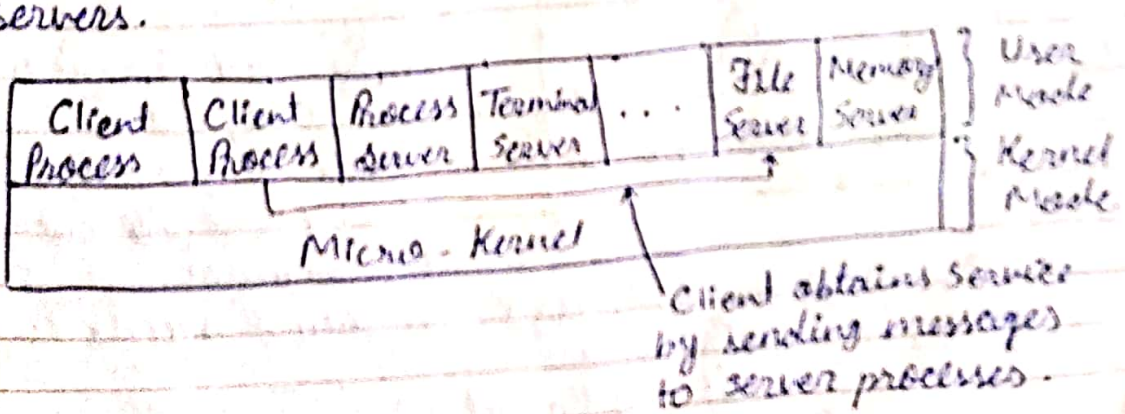
Virtual machine is a software computer that, like a physical computer, runs an OS and applications.

③ Virtual Machines
- The system originally called CP/CMS, later renamed VM/370, was based on an astute observation.
- That was a time sharing system, provides multi-programming and an extended machine with a more convenient interface than the bare hardware.
- The heart of the system known as virtual machine monitor that runs on the bare hardware and ~~those~~ does the multiprogramming, providing several virtual machines to next layer up as shown in the given figure



- These virtual machines are not extended machines, with files and other nice features.
- They are the exact copies of the bare hardware, including the kernel/user mode, Input/Output, interrupts and everything else the real machine has.

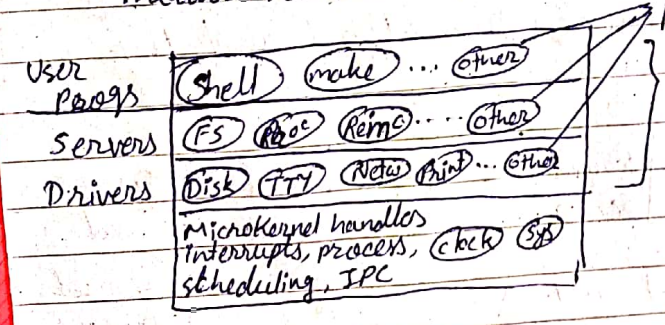
④ Client-Server Model
- In the client-server model, as shown in the figure given below, all the kernel does is handle the communication between the client and the servers.



→ By splitting the OS up into parts, each of which only handled one fact of the system, such as file service, terminal service, process service or memory service, each part becomes small and manageable.
→ The adaptability of the client-server model, to use in distributed system is the advantage of this model.

⑤ Exo-Kernel
→ Exokernel is a program present at the bottom layer running in the kernel mode.
→ The work of exokernel is just to allocate the resources to the virtual machines and check attempts to use them to make sure no machine is trying to use some other's resources.
→ Exokernels saves a layers of mapping which is the advantage of the exokernel scheme.

⑥ Micro-Kernel
→ In layered approach, the designer have choice where to draw the kernel and user mode boundary.
→ It is better to put as little as possible in kernel mode because bugs in the kernel can bring down the system instantly.
→ The microkernel design provides high relatibility by splitting OS up into small well defined modules, only one module run in kernel and rest of all run in user mode.



→ As each device driver runs as a user process, a bug in audio driver will cause the sound to be stop, but not crash the computer
→ Eg. Integrity, K42, QNX, Symbian & MINIX3.
→ Kernel consists only
• Sys (Kernel call handler)
• Clock (because scheduler interact with it)