OPERATING SYSTEM

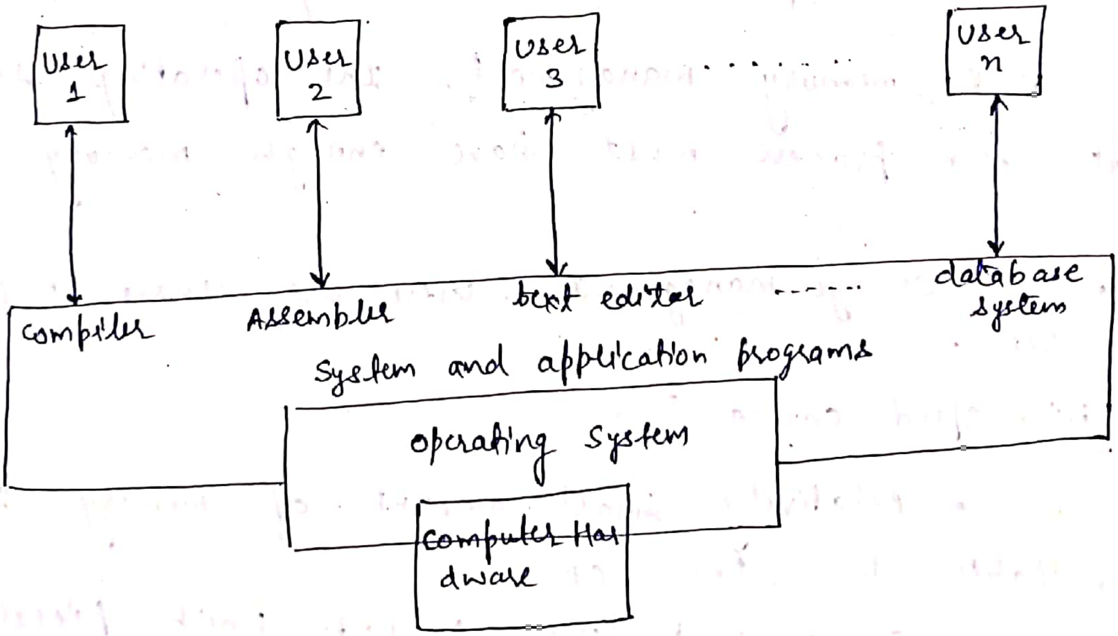
24

UNIT - I

Operating system and functions

- An operating system is a program that manages the computer hardware or resources (memory, I/O devices, processor).

- It provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware or resource.



Functions of operating system :→

(i) User Interface

(II) Processor management

(III) Memory management

(IV) Storage management

(V) Device management

(VI) ~~House Keeping~~ File management

• User Interface is a set of commands or menus through which a user communicates with a program.

• In processor management, the operating system must ensure that each running application (process) is treated fairly in terms of processor time allocated in a multi-Tasping environment.

• The processor is maximally and efficiently utilised

• In memory management, the operating system ensures that each process must have enough memory to execute.

• In storage management, there are three types of memories -

(a) High speed cache :→

• Relatively small amount of memory that are available to the CPU.

• Cache controllers pridict which pieces of data the CPU will need next and pull it from main memory into high speed cache to speed up system performance.

(b) Main Memory :→

(c) secondary Memory :→

- In Device management, operating system takes care of processor, memory, Bus and other I/O devices in terms of interaction with users.

~~Classification of operating Systems~~

(vi) ~~House keeping~~ File management :→

- creating file system
- copying, deleting, moving files
- starting the computer
- Interfacing with the hardware

Classification of Operating systems

(i) Batch system :→

- Batch operating system is the operating system which analyzes your i/p and groups them (similar characteristic) into batches.
- Executing a series of non interactive jobs all at one time.
- Users entered programs on punch cards
- Users give a batch of these programmed cards to the system operator to feed them into the computer.

- Batch operating systems could only execute one program at a time.
- Batch jobs can be stored up during working hours then executed during the evening or whenever the compu is idle.
- Useful for operations that sequire the computer or a peripheral device for an extended period of time.
- Once a batch job begins, it continues until it is done or until an error occurs.
- An example of batch processing is credit card billing.
- The customer does not receive a bill for each separate credit card purchase but one monthly bill for all.
- The bill is created through batch processing, where all of the data are collected and held until the bill is processed as a batch at the end of the billing cycle.
- Other examples of such programs include payroll, forecasting, statistical analysis etc.
- Scheduling in batch system is done either in first come first serve or shortest job first fashion.
- In memory management, it is divided into two parts.
- One is permanently occupied by the resident portion of the operating system and other is used to load transient programs for execution.
- When a transient program terminates, a new program is loaded

into the same area of memory.

- Example of batch operating system is IBM OS/2, IBM' OS360.
- Batch system do not require any time-critical device management because only one program is executed at a time.

(ii) Interactive Operating System :→

- It is an operating system which allows users to interact with programs.
- Provides direct communication between the user and the system.
- User gives instructions to the operating system or to a program directly using input device such as a key-board or a mouse and waits for immediate results on output device.
- It process data immediately and can not wait like a batch operating system.
- Unix is both an interactive operating system and a batch operating system (programs run without user interaction, i.e. programs run in the background).
- Mac & Windows
  (MS word or spreadsheet)

(iii) Multiprogramming Operating System :→

- Multiprogramming increases CPU utilization by organi jobs so that the CPU always has one to execu
- The operating system keeps several jobs in memory simultaneously.
- This set of jobs can be a subset of the jobs kept in the job pool.
- Since the number of jobs that can be kept simultane onely in memory is usually smaller than the number of jobs that can be kept in the job pool.
- The operating system picks and begins to execute one of the jobs in memory.
- Eventually the job may have to wait for some task such as an I/O operation to complete.
- In non multiprogrammed systems, the CPU would sit idle
- In a multiprogrammed system, the operating system simply switches to and execute another job.
- When that job needs to wait the CPU switched to another job and so on.
- Eventually the first job finishes waiting and gets the CPU back.
- It utilizes system resources (CPU, memory, peripheral devices) effectively but does not provide user interaction with computer system.

- For example you may be typing in word, listning to music while in background IE is downloading.

- Windows, Linux, Unix are the multiprogramming operating system.

(iv) Time sharing or multitasking system :→

- It is logical extension of multiprogramming.
- The CPU executes multiple jobs by switching among them, but the switches occurs so frequently that the users can interact with each program while it is running.
- It requires interactive computer system.
- It allows many users to share the computer simultaneously.
- Since each action or command tends to be short, only a little CPU time is needed for each user.
- 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 his use.

- For example Dozens of users access the computer through terminals at once. The computer actually does not run them all simultaneously but it runs a small portion of ones user's job then moves the service to next user.

- Time sharing operating system uses CPU scheduling and multiprogramming to provide each user with small portion of a time shared computer.

- When a process executes, it typically executes only a short time before it either finishes or needs to perform I/O.

- If several jobs are ready to run at the same time CPU scheduling concept breaks the conflict.

- In time sharing system the operating system must ensure reasonable response time which is sometimes accomplished through swapping, where processes are swapped in and swapped out of main memory to the disk.

(V) Real Time System :→

- A real time system has well defined fixed time constraints.

- Processing must be done within the defined constraints or the system will fail.

- Primary objective of real time systems is to provide quick event - response times, and thus meet the scheduling deadlines.

- A hard real time system has the most stringent requirements, guaranteeing that critical real time tasks be completed within their deadlines

- A soft real time system is less restrictive, simply providing that a critical real time task will receive priority over other tasks and will retain that priority until it completes.

- Example of hard real time system is airplane control systems and soft real time system is live video streaming.

- Any general purpose operating system such as windows 2000, IBM's OS/390 can be evaluated for its ~~real~~ real time operating system qualities.

vi) Multiprocessor System :→

- Multiprocessor systems have two or more processors in close communication (tightly coupled system). sharing the computer bus and some times the clock, memory, and peripheral devices.

Advantages :→

Increased Throughput :→

- By increasing the number of processors, we can get more work done in less time.

- When multiple processors cooperate on a task, a certa amount of overhead is incurred in keeping all parts working correctly.

- If there is N processor then speed-up ratio will less than N due to overhead and contention shared resources.

Economy of scale :→

- Multiprocessor systems can cost less than equivalent multiple single-processor system, because they can share peripherals, mass storage and power supplies.

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.

(vii) Multiprocess System :→

- This system supports more than one process (Program) at the same time

- Multiprocessing operating systems, enable several programs to run concurrently.

- Unix and OS/2 is the example of multiprocess operating system.
- It is much more complicated than single process systems because the operating system must allocate resources to competing processes in reasonable manner.
- Also refers to the utilization of multiple CPU in a single computer system and also called parallel processing.
- It is of two types. Asymmetric multiprocessing and symmetric multiprocessing.
- In Asymmetric multiprocessing each processor is assigned a specific task.
- A master processor controls the system.
- Other processors are instructed by master processor or have predefined task.
- In symmetric multiprocessing each processor performs all tasks within the operating system.
- There is no master slave relationship, all processors are peers.
- Example of SMP is solaris and commercial version of UNIX.

- special hardware can differentiate the multiple processors ix) the software can be written to allow only one m and multiple slaves.

- Sun operating system Sun OS version 4 provided asymetric multiprocessing, whereas version 5 (solaris) is symmetric on the same hardware.

(viii) Multiuser System :→

- Computer systems that support two or more simultaneous users.

- All mainframes and minicomputers are multiuser system but personal computers and workstations are not.

- Another term for multiuser is time sharing.

- In other words system that allows concurrent access by multiple users of a computer.

- Example of multiuser operating system include Unix server & thin client.

- In the Unix server multiple remote users have the access to the unix shell at the same time.

- In the case of thin client, multiple window sessions are spread across various terminals and these terminals are powered by a single machine.

(ix) Multithreaded System :→

- Operating system that provide features enabling a process to contain multiple threads of control.

- If a process has multiple threads of control, it can perform more than one task at a time.

- Many software packages that run on modern desktop PCs are multithreaded.

- A web browser might have one thread display images or text while another thread retrieves data from the network.

- For example a word processor may have a thread for displaying graphics, another thread for responding to key strokes from the user and a third thread for performing spelling and grammar checking in the background.

Advantages :-

Responsiveness :→

- Allow a program to continue running even if part of it is blocked.

- For a instance a multithreaded web browser could still allow user interaction in one thread while an image was being loaded in another thread.

Resource Sharing :→

- By default threads share the memory and the resources of the process to which they belong. Lo

- Allows an application to have several different threads of activity within the same address space

Economy :→

- Allocating memory and resources for process creation is costly.

- Because threads share resources of the process to which they belong it is more economical to create and context-switch thread.

Utilization of multiprocessor architectures :→

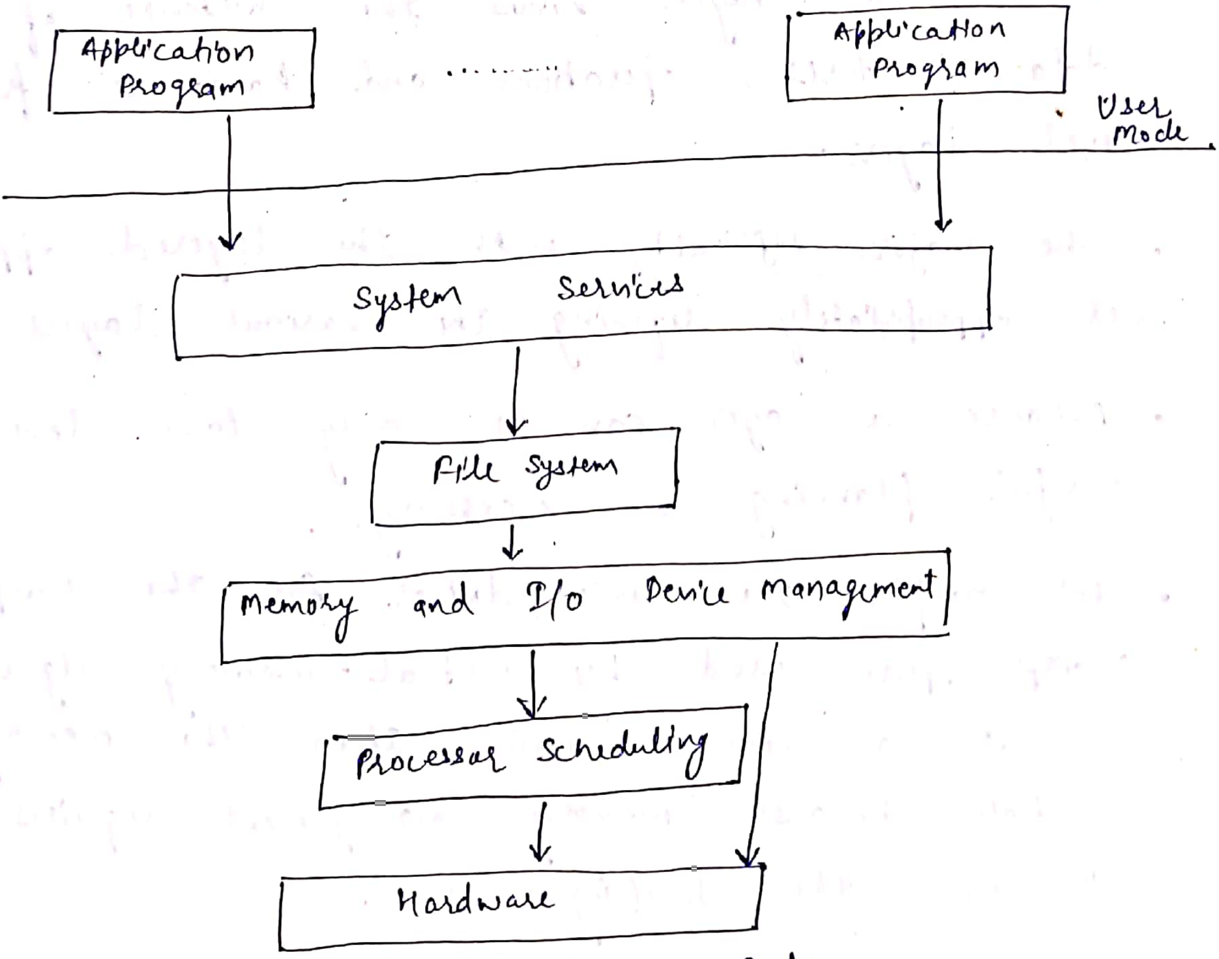
- A single-threaded process can only run on one CPU no matter how many are available.

- Multithreading on a multi-CPU machine increases concurrency.

Operating System Structure

Layered Structure :→

- In layered structure operating system is broken up into a number of layers (levels).
- The bottom layer (layer 0) is the hardware, the highest (layer N) is the user interface.
- A typical operating system layer - say layer M - consists of data structures and a set of routines that can be Invoked by higher-level layers.
- Layer M, in turn can invoke operations on lower level layers.



Layered Operating System

- The main advantage of the layered approach is simplicity of construction and debugging.

- The first layer can be debugged without any con for the rest of the system, because it uses only the basic hardware to implement its functions.

- Once the first layer is debugged its correct functioning can be assumed while the second layer is debugged and so on.

- Each layer is implemented with only those operation provided by lower level layers and does not need to know how these operations are implemented.

- It needs to know only what these operations do and hence each layer hides the existence of certain data structures, operations and hardware from higher level layers.

- The major difficulty with the layered approach involves appropriately defining the various layers.

- Because a layer can use only lower level layers careful planning is necessary.

- For example the dence driver for the backing store (disk space used by virtual-memory algorithms) must be at a lower level than the memory management routine, because memory management requires the ability to use the backing store.

- The backing store driver would normally be above the CPU schedule, because the driver may need to wait for I/o and the CPU can be rescheduled.

- Less effective because 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 mgmt layer, which calls the CPU scheduling and then passed to the hardware.

- At each layer the parameters may be modified data may need to be passed and so on.

- Each layer adds overhead to the system call, the net result is a system call that takes longer than does one on a nonlayered system.

Micro kernels :→

- In microkernel approach structure of operating system is formed by removing all non essential components from the kernel and implementing them as system and user level program.

- The result is smaller kernel and provide minimal process and memory management to a communication facility.

- The main function of microkernel is to provide a com cation facility between the client program and va services that are running in user space.

- For example if the client program wishes to acc a file it must interact with the file serv rather than interacting directly it communicat indirectly by exchanging messages with the micro kernel. This is called message passing.

- Advantage is ease of extending the operating system.

- All new services are added to user space and consequently do not require modification of the kernel.

- If there is need of modification then it will be very few or small due to small kernel.

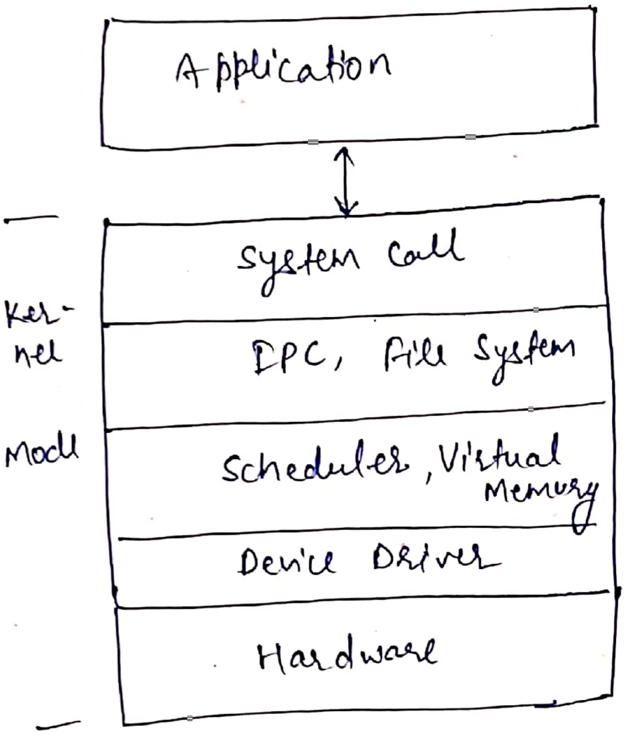
- Also provides more security and reliability since services are running as user - rather than kernel processes.

- If service fails the rest of the operating system remains untouched.

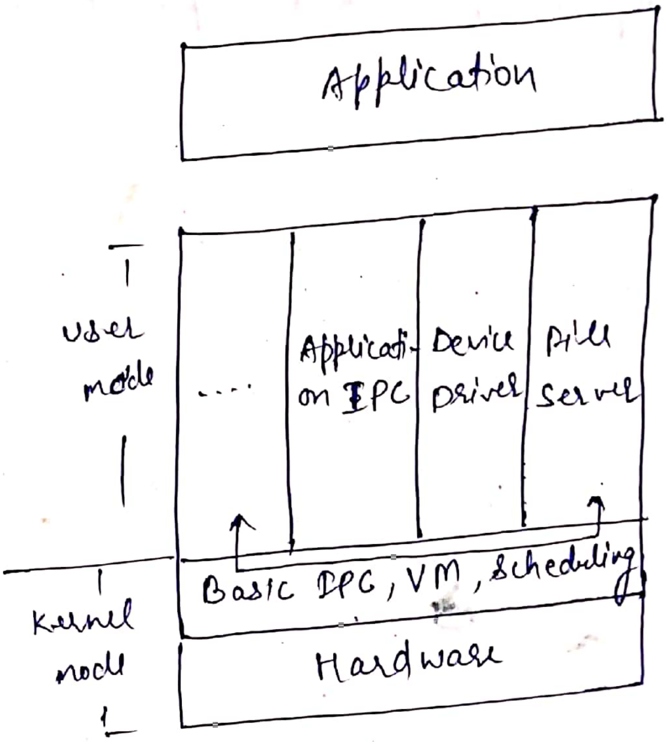
- Examples are Tru64UNIX (Digital UNIX), windows NT.

- Disadvantage is that it can suffer from performance decrease due to increased system function overhead.

• Monolithic Kernel based operating system



microkernel based operating system



• Routine and sub routine are general and nearly synonymous terms for any sequence of code that is intended to be called and used repeatedly during the executable of a program.

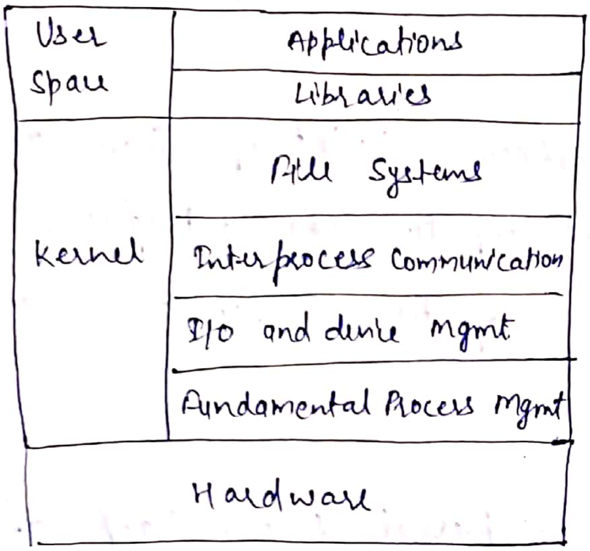
- For example in assembler language a routine that requires some variable input can be encoded into a macro definition with a specified interface called a macro - instruction.

Monolithic Kernel OS - OS-9, open vms, Linux, BSD etc.

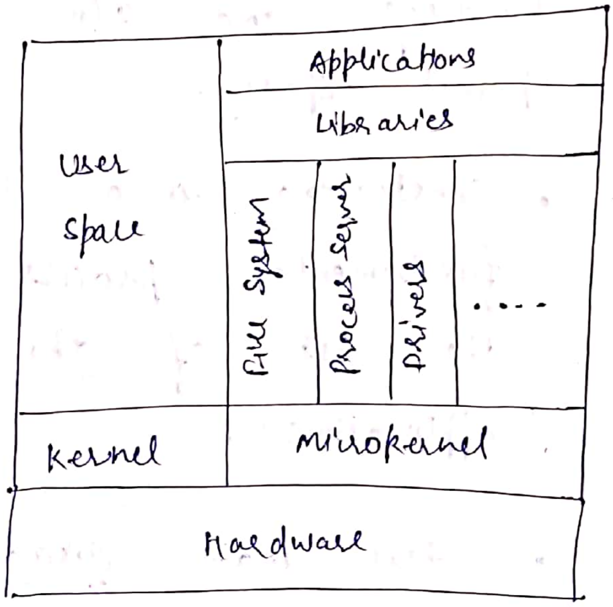
microkernel - Mach, Qnx, minix, Amiga

System call is a request for service that a progra
makes of the kernel.

Monolithic Kernel :→



Monolithic Kernel OS



Microkernel OS

Kernel :—

- The kernel manages communication between hardware and software.
- Responsible for managing memory, and I/o to memory cache, the hard drive and other devices.
- Also handles device signals, task scheduling and other essential duties.
- One of the first components loaded into memory during the boot process and remains active as long as the computer is operational.

→ Operating system itself consists of two parts
(a) The kernel space (privileged mode)
(b) User space (unprivileged mode)

- monolithic kernel based OS runs every basic sys service like process and memory management, Inter handling and I/O communication, file system etc. I. kernel space.

- Constructed in a layered fashion, built up from the fundamental process management up to the interfaces to the rest of the operating system (libraries and applications).

- Three main <u>disadvantages</u> = kernel size, lack of extensibility and the bad maintainability.

- Bug fixing and addition of new features means a recompilation of the whole kernel.

- compillation of new kernel can take several hours and a lot of memory.

- In <u>microkernel</u>, basic process communication and I/O control reside on kernel space and let the other, system services reside in user space in form of normal processes (servers).

- There is a server managing memory issues, one server does process management, other drivers and so on.

Operating System Services

(i) Program Execution :→

- The operating system provides an environment where the user can conveniently run programs.
- The user does not have to worry about the memory allocation or multitasking etc. (CPU scheduling)

(ii) I/O operations :→

- The user need only specify I/O device and the operation on it.
- The operating system converts that request into device - or controller specific commands.

(iii) File system Manipulation :→

- There are many details in file creation, deletion, allocation and naming that users should not have to perform.
- Blocks of disk space are used by files and must be tracked.
- Deleting a file requires removing the name file information and freeing the allocated blocks.
- Protections must also be checked to assure proper file access.

(IV) communication :→

- Message passing between two processes in the sa machine or between systems.

- Requires messages to be turned into packets of in mation and sent to the network controller.

- It is operating system which perform these task user does not worry about that.

(V) Error detection :→

- Error detection occurs at both the hardware and software levels.

- At hardware level all data transfers must be inspe ted to ensure that data have not been corrupted in transit.

- At the software level media must be checked for data consistency for instance whether the number of allocated and unallocated blocks of storage match the total no. on the device.

- Operating system takes care of all these operations.

- Also detects errors in the CPU and memory hardware, in I/O devices, or in user programs.

Common System Components

(i) Process Management :→

- The operating system is responsible for the following activities in connection with process management.

(a) Process creation and deletion

(b) Process suspension and resumption

(c) Process synchronization & process communication

(ii) Main Memory Management :→

- Operating system keeps track of which parts of memory are currently being used and by whom.
- O S decide which processes to load when memory space becomes available.
- Allocate and deallocate memory space as needed

(iii) File management :→

- OS is responsible for file creation and deletion.
- Directory creation and deletion.
- support of primitives for manipulating files and directories.
- mapping files onto secondary storage.
- File backup on stable (non volatile) storage media.

(iv) I/O system management :→

- The I/O system consists of

(a) A buffer - caching system.

(b) A general device - driver interface

(c) Drivers for specific hardware devices.

(v) Secondary storage Management :→

- OS is responsible for free space management, storage allocation and disk scheduling.

(vi) Networking (Distributed Systems) :→

- A distributed system is a collection of processors that do not share memory or a clock. Each processor has its own local memory.
- The processors in the system are connected through a communication network.
- Communication takes place using a protocol.
- A distributed system provides user access to various system resources and hence allows computation speedup, increased data availability, enhanced reliability.

(vii) Protection system :→

- Protection refers to a mechanism for controlling access by programs, processes or users to both system and user resources.

iii) Command - Interpreter System :→

Many commands are given to the operating system by control statements which deal with :

→ Process creation and management

- I/O handling
- Secondary storage management
- Main memory management
- File system access
- Protection
- Networking
- The program that reads and interprets control statements is called command line interpreter or shell (in Unix)

Reentrant Kernels

- A reentrant program is one that does not modify itself and any global data.
- Multiple processes or threads can execute reentrant programs concurrently without interfering one another.
- A reentrant kernel is one where many processes/threads can execute the same kernel programs concurrently without affecting one another.

- In non reentrant kernels, a process does not modify kernel programs, but can modify global kernel data;

- A reentrant kernel enables processes to give away the CPU while in kernel mode, not hindering other processes from also entering kernel mode.

- A typical use is I/O wait.

- The processes wants to read a file, it calls a kernel function for this.

- Inside the kernel function, the disk controller is asked for the data.

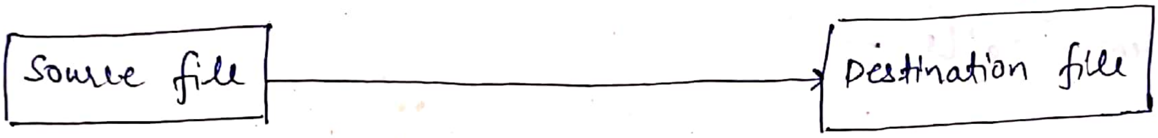
- Getting the data will take some time & the function is blocked during that time.

- With an reentrant Kernel, the scheduler will assign CPU to another process until an interrupt from disk controller indicates that the data is available and our thread can be resumed.

- This process can still access I/O, like user input.

System Calls

- System calls provide an interface to the services made available by an operating system.
- These calls are available as routines written in C and C++.
- For example writing a simple program to read data from one file and copy them to another file.



Example of how system calls are used

Example system call sequence

```
Acquire input file Name
  Write prompt to screen
  Accept Input
Acquire output file name
  Write prompt to screen
  Accept input
Open the input file
  If file does not exist, abort
Create output file
  If file exists, abort
Loop
   Read from input file
    Write to output file
Until read fails
Close output file
Write completion message to screen
 Terminate normally
```

- Behind the scenes, the functions that make up an API typically invoke the actual system calls on the behalf of application programmer.

- For example the win32 function CreateProcess() actually calls the NTCreateProcess() system call in the windows kernel.

Why would an application programmer prefer programming according to an API rather than invoking actual system calls –

(a) Portability :→

- An application programmer designing a program using an API can expect her program to compile and run on any system that supports the same API.

(b) Actual system calls can often be more detailed and difficult to work with than the API.

System call interface that serves as the link to system calls made available by the operating system.

- The system call interface intercepts function calls in the API and invokes the necessary system call with in the operating system.

A number is associated with each system call and the system call interface maintains a table indexed according to these numbers.

- The system call interface then invokes the intended system call in the operating system kernel.
- And returns the status of the system call and any return values.

Types of System Calls

Routine & subroutine → It is a sequence of program instructions that perform a specific task, packaged as a unit. This unit can be used in programs wherever that particular task should be performed. Subprograms may be defined within a program or separately in libraries.

- 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

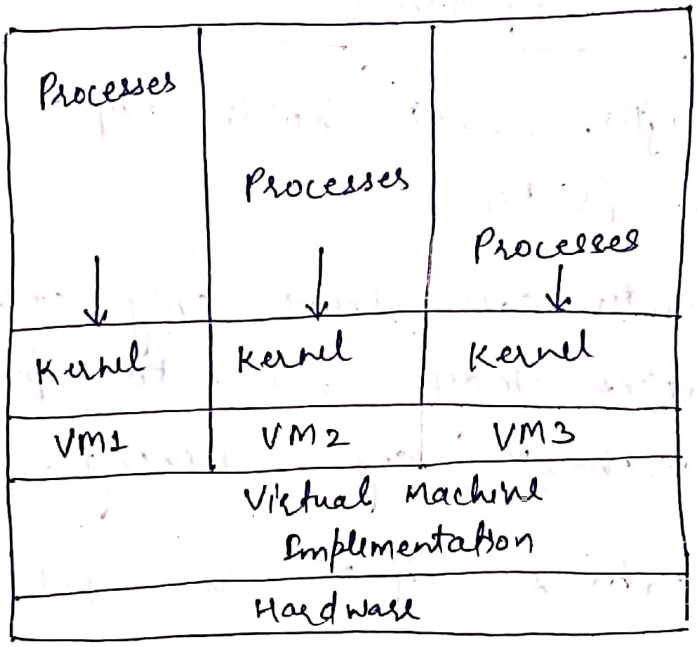
- Device Management
  - Request device, release device
  - Read, write, reposition
  - Get device attributes, set device attributes
  - logically attach or detach devices

- 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

- Communications
  - create, delete communication connection
  - send, receive messages
  - transfer status Information
  - Attach or detach remote devices

Virtual Machines

- The fundamental idea behind a virtual machine is to abstract the hardware of a single computer (CPU, memory , disk , NIC, etc) into several different execution environments .
- Thereby creating the illusion that each separate execution environment is running its own private computer.



Virtual Machine

- There are several reasons for creating a virtual machine all of which are fundamentally related to being able to share the same hardware yet run several different execution environments (i.e. different operating system) .
- Difficulty with the virtual machine approach is disk systems .

- suppose that the physical machine has three disk dri-
but wants to support seven virtual machines.

- It can not allocate a disk drive to each virtual machine because the virtual machine software itself will need subs-tantial disk space to provide virtual memory and spooling.

Spooling

- Spool is a buffer that holds output for a device, such as a printer.

- Although printer can serve only one job at a time several applications may wish to print their output concurrently without any mixing.

- Each application's output is spooled to a separa-te disk file by an operating system.

- When an application finishes printing, the spooling system queues the corresponding spool file to the printer.

- The spooling system copies the queued spool file ~~for~~ ~~[illegible]~~ to the printer one at a time.