Chap 1: Introduction

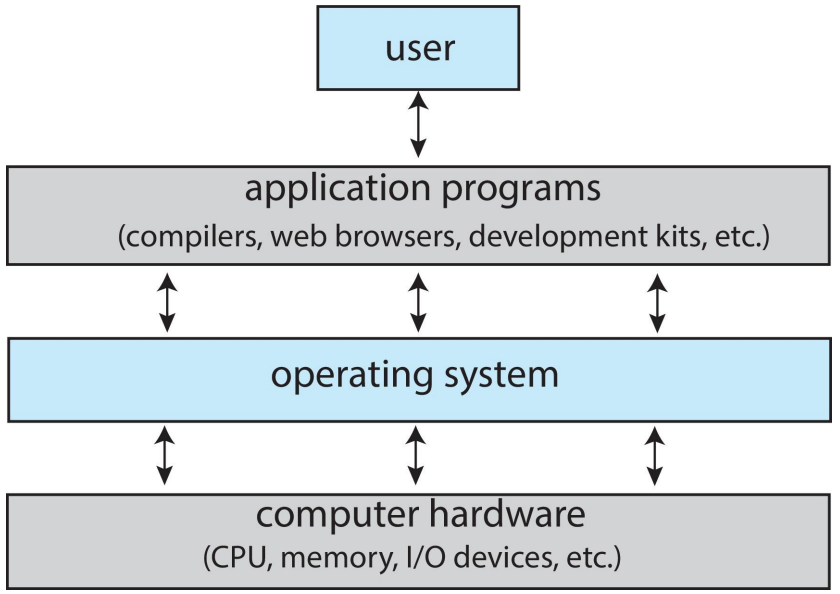
BY PRATYUSA MUKHERJEE, ASSISTANT PROFESSOR (I)

KIIT DEEMED TO BE UNIVERSITY

Computer System Structure

Computer system can be divided into four components:

- **Hardware** – provides basic computing resources such as CPU, memory, I/O devices
- **Operating system** – Controls and coordinates use of hardware among various applications and users
- **Application programs** – define the ways in which the system resources are used to solve the computing problems of the users. For example Word processors, compilers, web browsers, database systems, video games
- **Users** – People, machines, other computers



What is an Operating System ?

- A program that acts as an **intermediary** between a user of a computer and the computer hardware
- An operating system is software that manages the computer hardware and controls the execution of application programs. (software abstracting hardware)
- A more common definition is that the operating system is the **one program running at all times on the computer** (usually called the kernel),
- Operating system goals:
  - **Effective**:Execute user programs and make solving user problems easier
  - **Convinient**: Make the computer system convenient to use
  - **Efficient**: Use the computer hardware and other resources in an efficient manner
  - **Evolvable**: An OS should be constructed in such a way as to permit the effective development, testing and introduction of new system functions without at the same time interfering with service.

Types of Programs

- **Kernel** - The one program running at all times on the computer.
  It is the central component of the OS.
  It is the frst program loaded on start up.

- **System Program** - Programs associated with the OS but are not the part of Kernel.

- **Application Program** - All programs not associated with the operating system.

- Today's OSes for general purpose and mobile computing also include **middleware** – a set of software frameworks that provide addition services to application developers such as databases, multimedia, graphics

What does the OS do?

- **Intermediary** : It manages the computer HW and provides a basis for application programs
- **Control Program (kernel)**
- **Resource Allocation**:
  - Users want convenience, ease of use and good performance. They do not care about resource utilization.
  - But shared computer such as mainframe or minicomputer must keep all users happy.

Thus OS is a resource allocator and control program making efficient use of HW and managing execution of user programs.

- **Memory Management**:
  - Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory.
  - An OS keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
  - In multiprogramming, the OS decides which process will get memory when and how much. It allocates the memory when a process requests it to do so and also de-allocates the memory when a process no longer needs it or has been terminated.

- **Processor Management**:
  - In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called **process scheduling**.
  - An OS keeps tracks of processor and status of process. The program responsible for this task is known as **traffic controller**.
  - It does the allocation and de-aalocation of processors to processes.
  - It can create and delete both user and system processes.
  - It takes care of suspending and resuming processes.
  - It providing mechanisms for process synchronization, communication and deadlock handling.

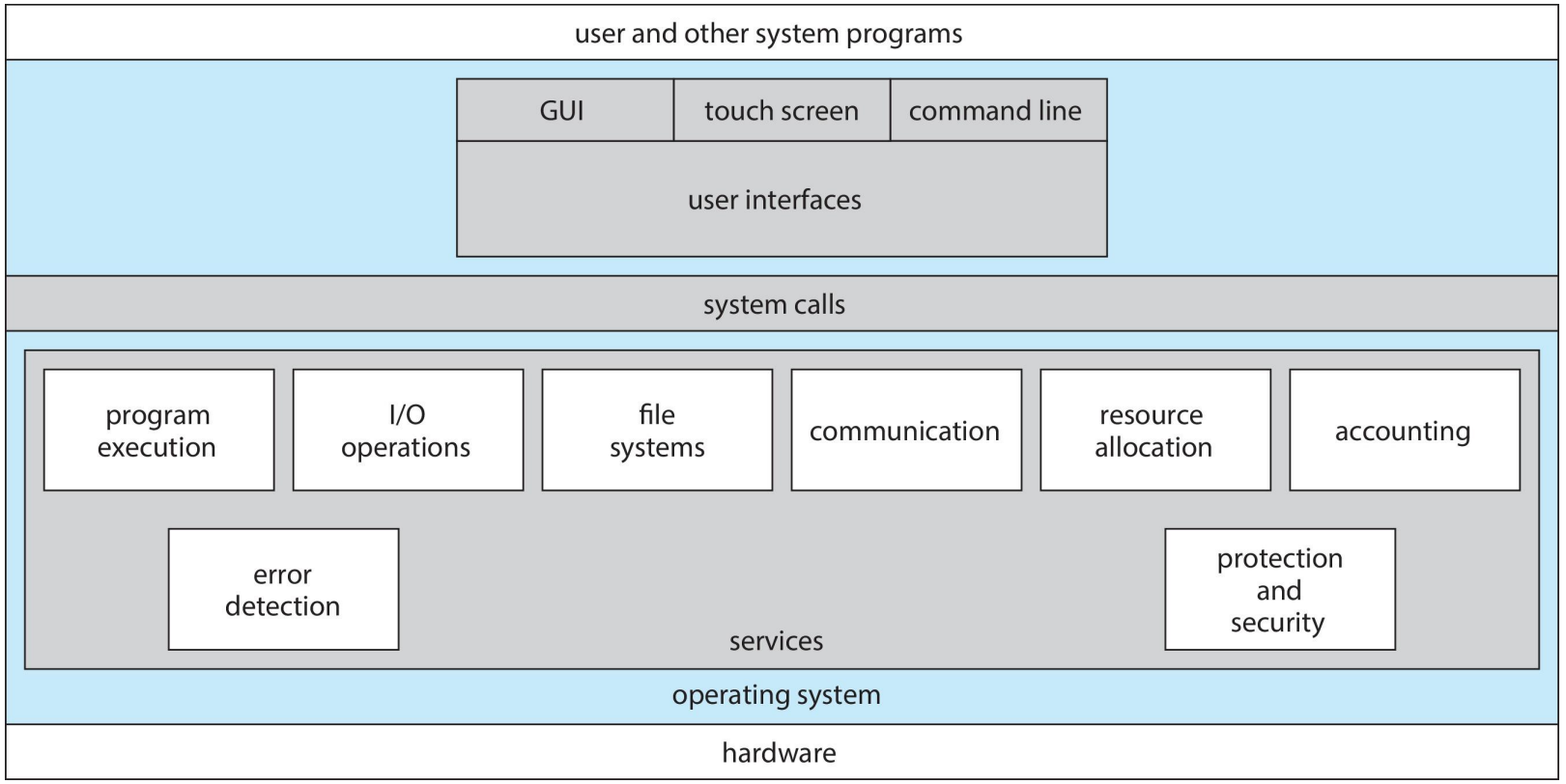
- **Device Management:**
  - An Operating System manages device communication via their respective drivers.
  - It tracks all devices. Program responsible for this task is known as the **I/O controller.**
  - It decides which process gets the device when and for how much time. It allocates and de-allocates devices in the efficient way.

- **File Management**:
  - A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.
  - An OS keeps track of information, location, uses, status etc. The collective facilities are often known as **file system**.
  - It provides access control on most systems to determine who can access what.
  - It decides who gets the resources and allocates as well as de-allocates the resources.
  - It enables creating and deleting files and directories and provides mapping files onto secondary storage as well as backup onto stable (non-volatile) storage media

Following are some of the important activities that an Operating System performs −

- **Security** − By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- **Control over system performance** − Recording delays between request for a service and response from the system.
- **Job accounting** − Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other softwares and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

A View of Operating System Services



Two Views of Operating System

Operating System: User View

The user view of the computer refers to the interface being used. Such systems are designed for one user to monopolize its resources, to maximize the work that the user is performing. In these cases, the operating system is designed mostly for ease of use, with some attention paid to performance, and none paid to resource utilization.

Operating System: System View

Operating system can be viewed as a resource allocator also. A computer system consists of many resources like - hardware and software - that must be managed efficiently. The operating system acts as the manager of the resources, decides between conflicting requests, controls execution of programs etc.

Types of Operating System

What is the difference between CPU Operation and I/O Operation ??

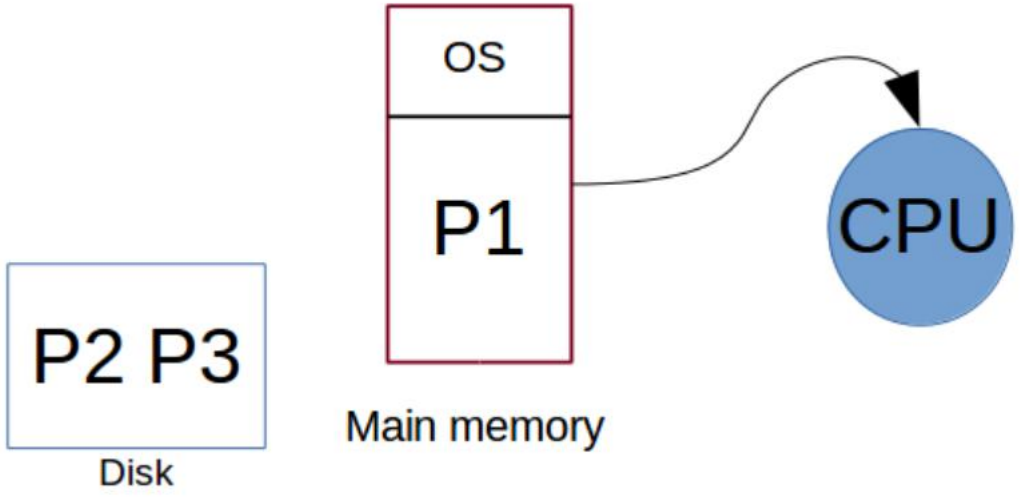
- **Serial Processing:**
  - Instructions are executed in FIFO.
  - A program counter is used for executing all instructions
  - All jobs are firstly prepared and stored in offline devices like Punch Card and after one particular card is entered in the system, all instructions are executed one by one.
  - There is involvement of an operator and usally the user does not interact with system while it operates.

- **Batch Processing:** Similar types of jobs are sorted and stored on a punch card called a batch

Disadvantages??

Job vs Task vs Process vs Program

- **Uni-Programming OS:**
  - Here only one program sits in main memory at a time (apart from the OS).
  - It was used in old computers and mobiles. When the computer starts then operating system and application programs are loaded into main memory. We only count user programs running in RAM (RAM is also called main memory)
  - Thus for example either of the browser, calculator or word processor runs at a time.



Disadvantages??

- **Multi Programming OS:**

  - Here multiple programs reside in main memory (RAM) at a time (apart from the OS).
  - It allows new jobs to take over the CPU whenever the currently running job waits (e.g. for user I/O).
  - OS uses **context switching** in main memory for running multiple programs.
  - CPU can execute only one job at a time thus Operating System may interrupt the process and give the control one of another ready process. In this way, idle time of CPU can be reduced.
  - A computer running excel and a browser simultaneously is an example of multiprogramming.

Can you represent it pictorially?

Preemptive Multiprogramming vs Non Premptive Multiprogramming OS

A program can force another program to leave the CPU	Program runs on CPU as long as it wants. It leaves the CPU either for I/O operation or it is terminated.

- **Interactive Operating System:**
  - It allows direct interaction with Operating System to supply commands and data as application program.
  - Typically, only user that has the full control over the system.
  - **Example??**

- **Real time Operating System:**
  - Real time Operating System give response within a predefined set period of time.
  - 2 Types: Hard and Soft (**Homework: Find the difference between them)**

- **Network Operating System**
- **Distributed Operating System**
- **Clustered Operating System** (types)
- **Parallel Operating System**
- **Embedded Operating System**
- **Hand Held Device Operating System**

Homework: Describe these

- **Time-Sharing Operating Systems**
  - Each task has given some time to execute for smooth execution
  - The task can be from single user or from different users also.
  - The time that each task gets to execute is called **quantum**. After this time interval is over OS switches over to next task.

- **Multi-Tasking Operating System**
  - Multitasking is the ability of an operating system to execute more than one task simultaneouslyon a single processor machine.
  - But in reality no two tasks can be executed on a single processor machine simultaneoulsly.
  - Actually CPU switches from one task to the next task so quickly that appears as if all the tasks are executing at the same time. **(Time quantum is pretty low)**
  - More than one task/program/job/process can reside into the same CPU at one point of time.
  - Thus, it provides the interface for executing the multiple program tasks by single user at a same time on the one computer system

Can you represent it pictorially?
Preemptive and Non Preemptive Multitasking OS??
Disadvantages?

- **Multiprocessing**
  - Multiprocessing is the ability of an operating system to execute more than one process simultaneously on a multi processor machine.
  - In this, a computer uses more than one CPU at a time.

Can you represent it pictorially?

Asymmetric and Symmetric Multitasking OS??

- **Multithreading**
  - Multithreading is the ability of an operating system to execute the different parts of a program called threads at the same time.
  - In multithreading system, more than one threads are executed parallely on a single CPU.

Define a Multi User Operating System ?

Operating System Operations

OS are **interrupt driven**

- If no work to do, OS sits quirtly and waits for something to happen.
- Thus events are signaled by occurrence of an **Interrupt / Trap**
- **Trap** is a s/w generated interrupt caused by an error or specific request from user program so that an OS service is performed.
- For each type of interrupt, separate segment of code in the OS determines the action to be taken
- Interupt Service Routine deals with the interrupt.

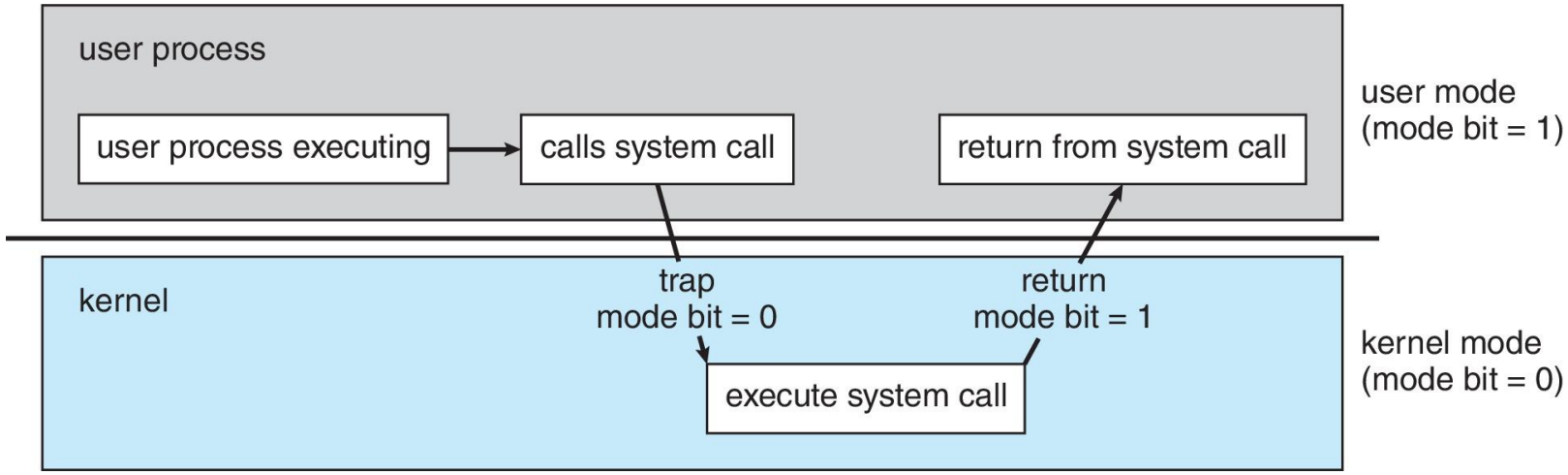
The 2 modes are:

- Dual Mode Operation
- Timer

An important criteria of OS

- The computer must execute only one process at a time to prevents erroneous executions.
- A well designed OS must ensure than an incorrect program cannot cause other programs to execute incorrectly
- Thus it is important to distinguish between OS codes and user defined codes.
- Most computers provide a hardware support that allows us to differentiate among various modes of execution

Dual Mode Operation



- We must separate the two modes of operation: **User mode** and **kernel mode** (superviser/system/priviledged mode)
- A bit - **Mode bit** - provided by hardware to indicate the current mode
  - If bit = 0 means kernel mode
  - If bit = 1 means user mode
- Provides ability to distinguish when system is running user code or kernel code.
  - When a user is running ➔ mode bit is “user”
  - When kernel code is executing ➔ mode bit is “kernel”
- But, when a user requests a service from OS (via a system call), the system must transition from user to kernel mode to fulfill the request.
- Thus when OS gains control of computer, it is in kernel mode. It will pass to user mode before passing the control to user program.
- Some instructions designated as privileged, only executable in kernel mode

Timer Operation

- It is important to ensure that OS maintains control over CPU
- A user program can not be allowed to get stuck in an infinite loop and never return the control to the OS
- Thus, we can use a timer - it interrupts the computer after a specific period.
- Before turning over the control to the user, the operating system ensures that the timer is set to interrupt. If it interrupts, control transfers to OS automatically.
- Instructions that can modify the content of a timer are privileged.

Operating System Components

Kernel

- It mainly interacts with the hardware.
- Acts as the core component of an OS
- It assists lot of activities at hardware level:
  - I/O management
  - Memory Management
  - Process Management
  - Device Management
  - System Calls Control etc.

Shell

- It mainly interacts with the user.
- Acts as the command interpreter and is the interface to the OS
- It prompts the user for input and finally interprets it for the user . - instruction handler
- Also handles the output/response from the OS

further explained on whiteboard

Design and Implementation of OS

- Design and Implementation of OS is not “solvable”, but some approaches have proven successful.
- Internal structure of different Operating Systems can vary widely.
- Start the design by defining goals and specifications.
- Affected by choice of hardware, type of system.
- **User goals** – operating system should be convenient to use, easy to learn, reliable, safe, and fast.
- **System goals** – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient.
- Specifying and designing an OS is highly creative task of software engineering.
- Partitioning task into small modules/components instead of monolithic systems is a good approach.

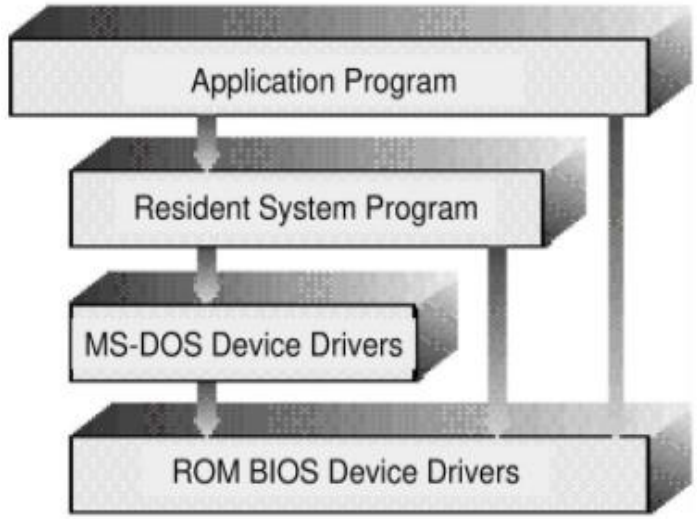
Operating System Structure

Various ways to structure the OS

- Simple structure – MS-DOS
- More complex – UNIX
- Layered – an abstraction
- Microkernel – Mach
- Modular
- Hybrid

Simple Structure OS : MS-DOS

- Intially OSs did not have well defined structures - MS- DOS
- MS-DOS was written to provide the most finctionality in least space, thus it was not carefully divided into modules
- Hence here, the interfaces and levels of functionality are not well separated.
- ***In MS-DOS application programs are able to access the basic I/O routines. These types of operating system cause the entire system to crash if one of the user programs fails.***



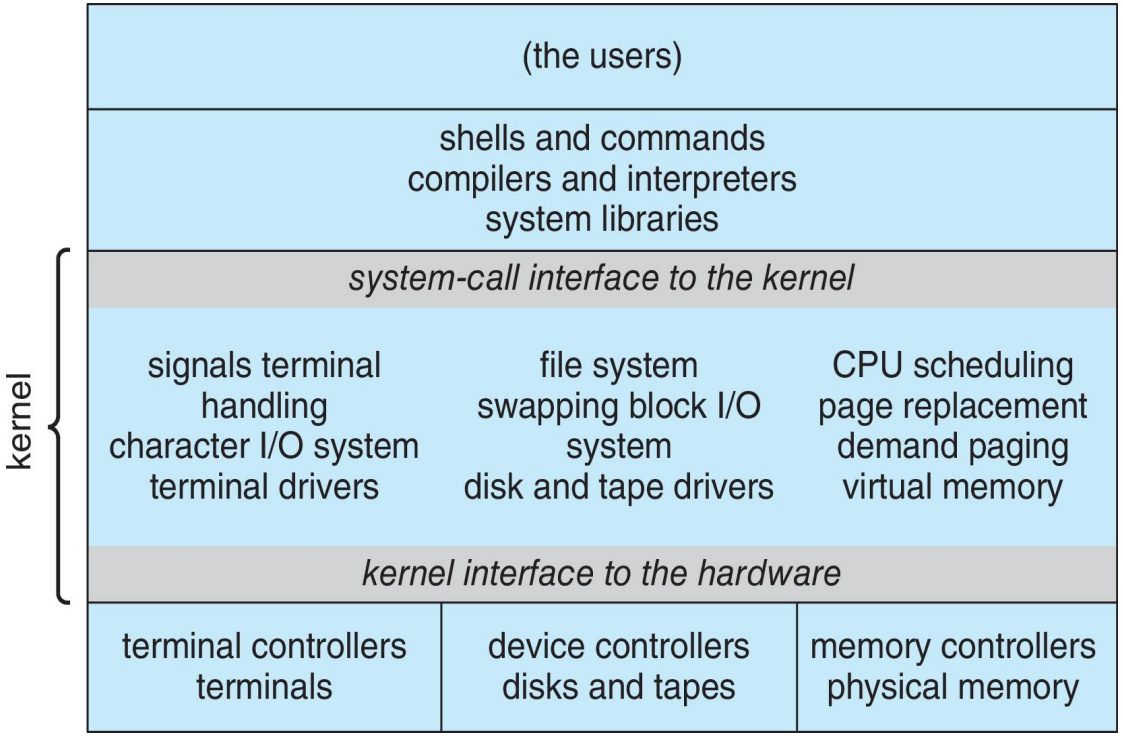
Advantages??
Disadvantages??

More complex OS – Original UNIX

- Original UNIX was limited by hardware functionality and had limited structuring.
- The UNIX OS consists of two separable parts: Systems programs and The kernel
- The Kernel consisted of everything below the system-call interface and above the physical hardware
- The kernel caters to the file system, CPU scheduling, memory management, and other operating-system functions through system calls. ***Thus a large number of functions for one level***

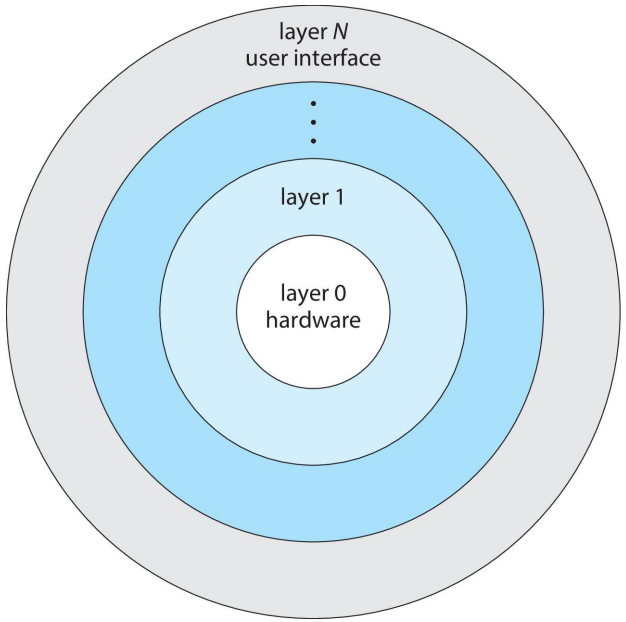
Advantages??

Disadvantages??



Layered Approach

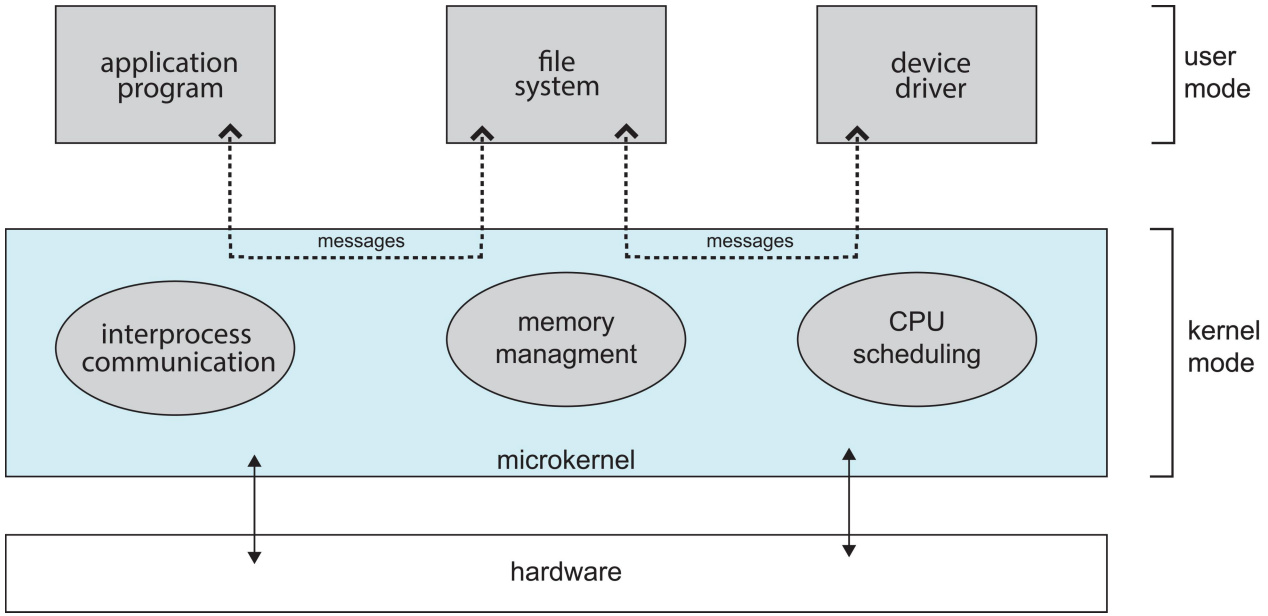
- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers



Advantages??
Disadvantages??

Microkernel

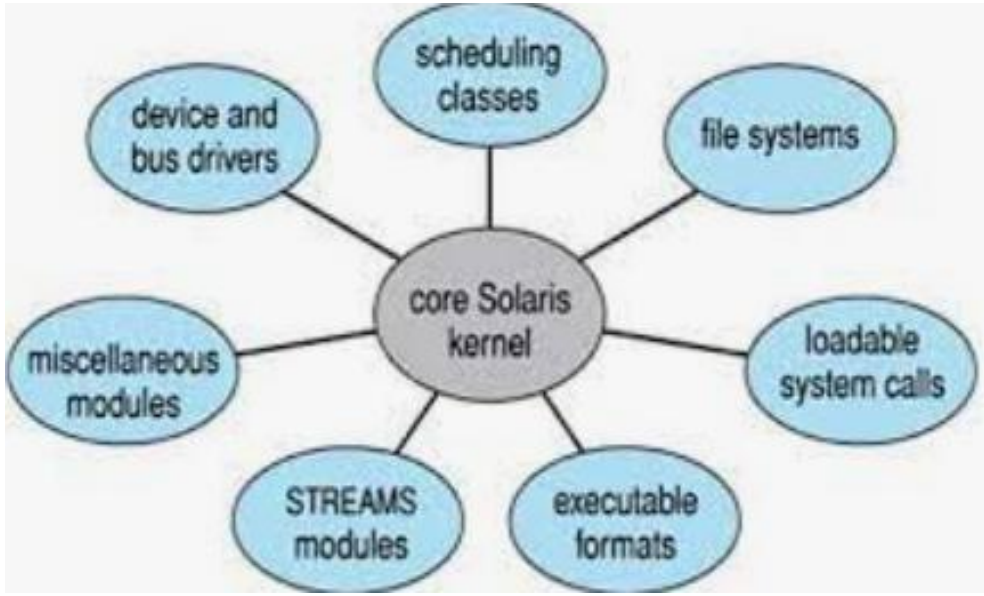
- Since kernel is the core part of the operating system, so it is meant for handling the most important services only.
- Thus in this architecture only the most important services are inside kernel.
- The rest of the OS services are present inside system application program.
- Thus users are able to interact with those not-so important services within the system application through **message passing**
- The microkernel is solely responsible for the most important services of operating system: Inter process-Communication, Memory Management and CPU Scheduling



Advantages??
Disadvantages??

Modular Approach:

- It involves designing of a modular kernel.
- The kernel has only set of core components and other services are added as dynamically loadable modules to the kernel either during run time or boot time.
- It resembles layered structure due to the fact that each kernel has defined and protected interfaces but it is more flexible than the layered structure as a module can call any other module.
- It is also similar to microkernel approach **(Justify ?)**



Can you guess what will be a Hybrid Approach??