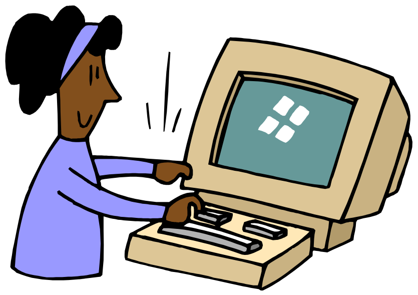
Basic elements of computer

Processor

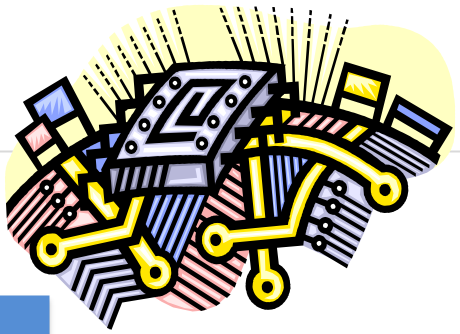
I/O Modules



Main Memory

System Bus

Processor



Referred to as the ***Central Processing Unit*** **(CPU)**

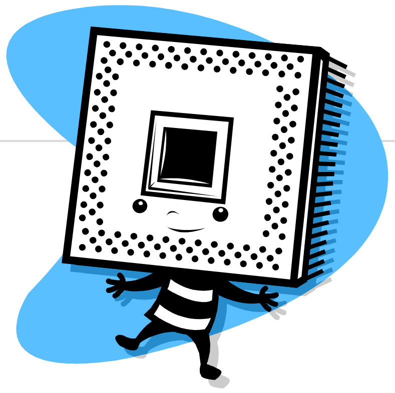
Arithmetic & Logic Unit

Performs the **data processing** functions

Control Unit

Controls the operation of the computer

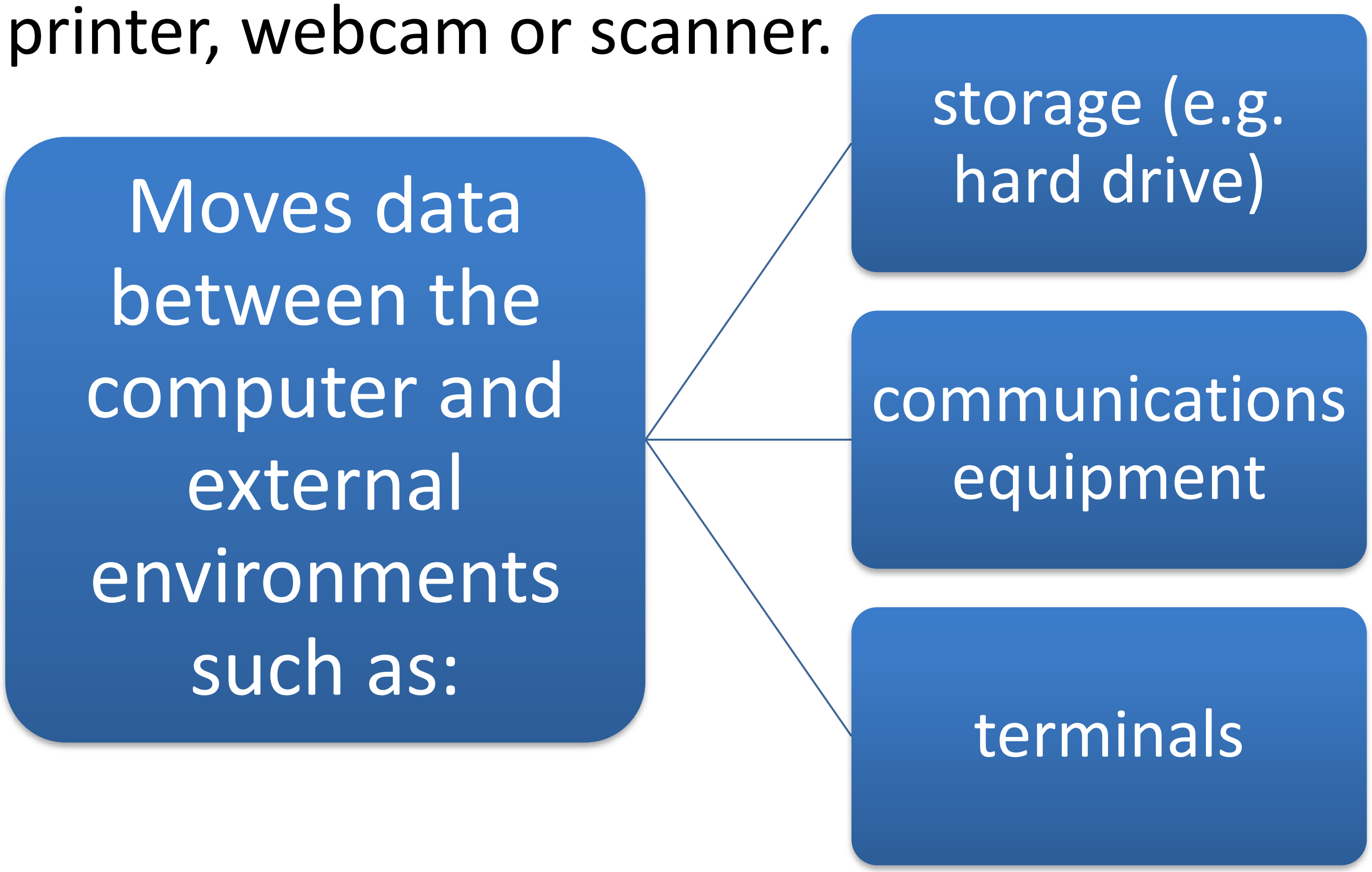
Main memory



- **Volatile**
  - **Contents of the memory is lost when the computer is shut down**
- Referred to as primary memory or **RAM**.

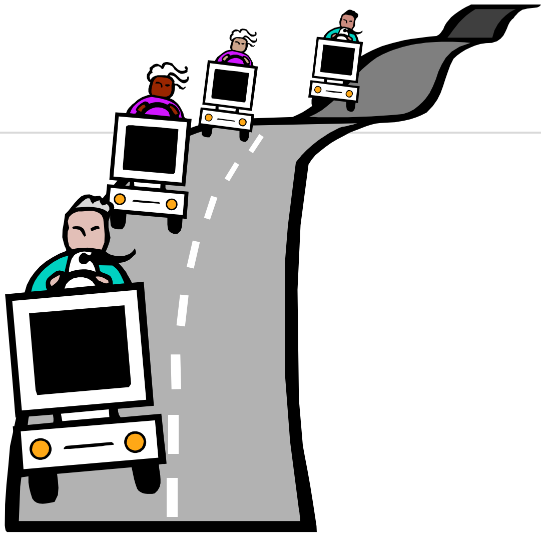
I/O modules

- Input/output module is a **device that acts as the connective bridge between a computer system** at one end and an **I/O or peripheral device** at the other, such as a printer, webcam or scanner.

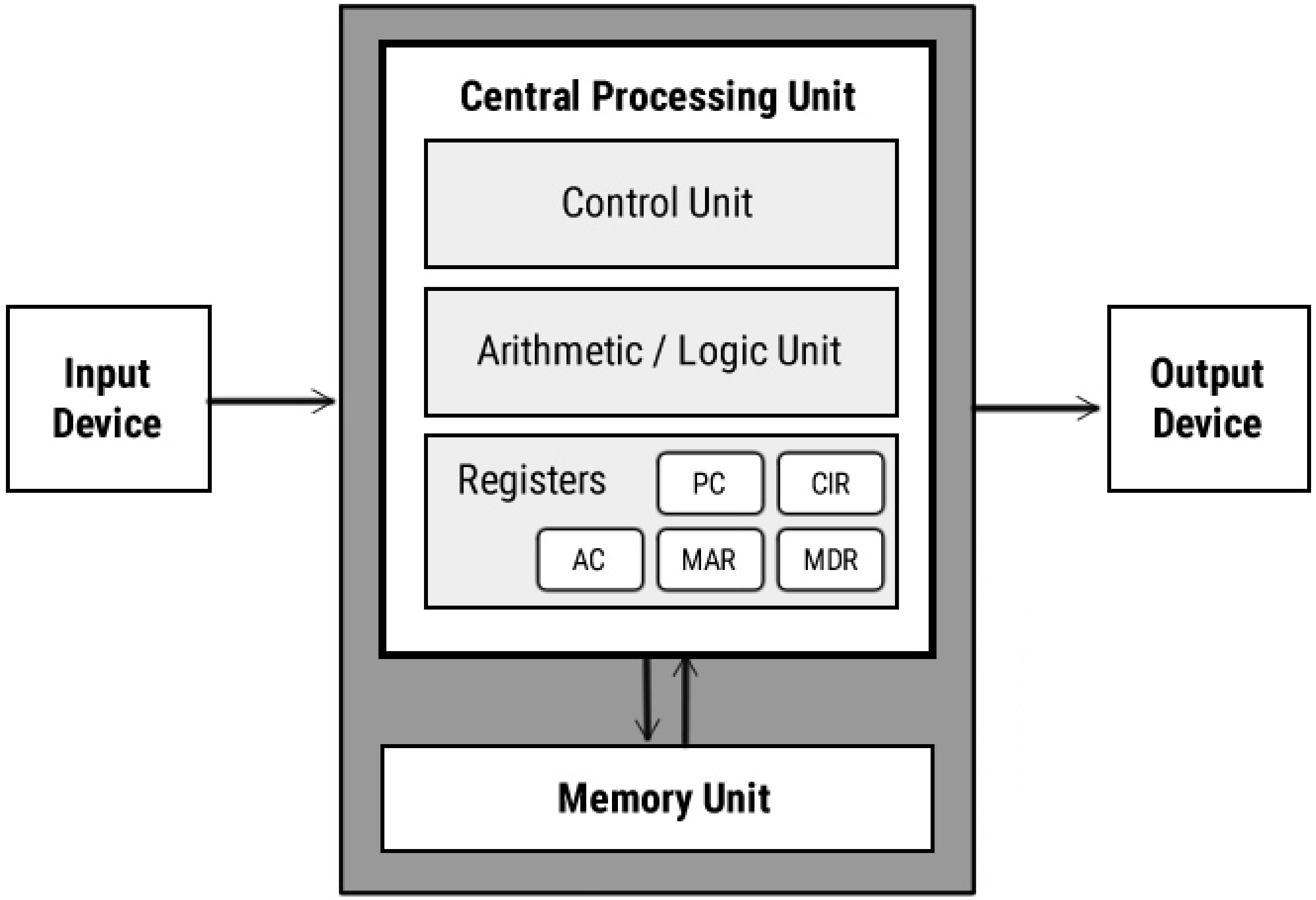

Moves data between the computer and external environments such as:
storage (e.g. hard drive)
communications equipment
terminals

System bus

- **Provides communication among processors, main memory, and I/O modules.**

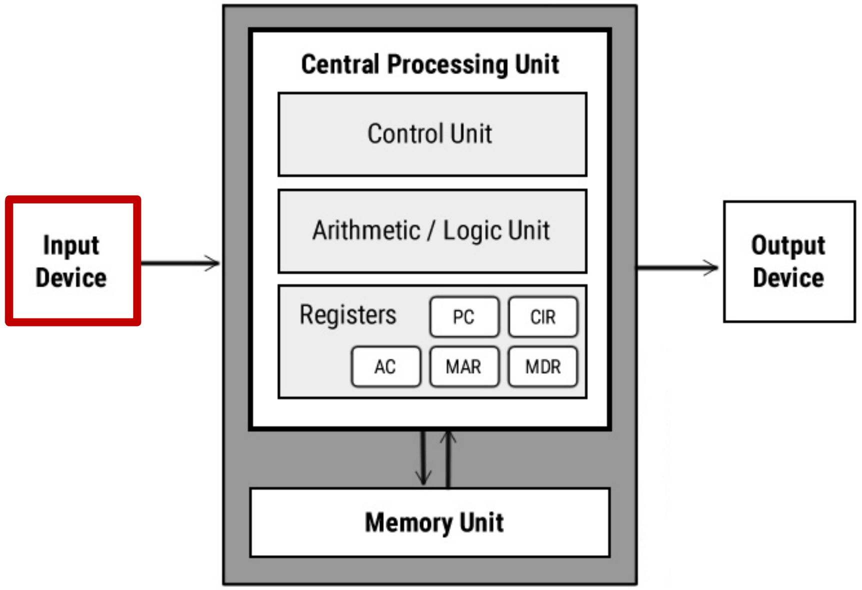


Computer system architecture



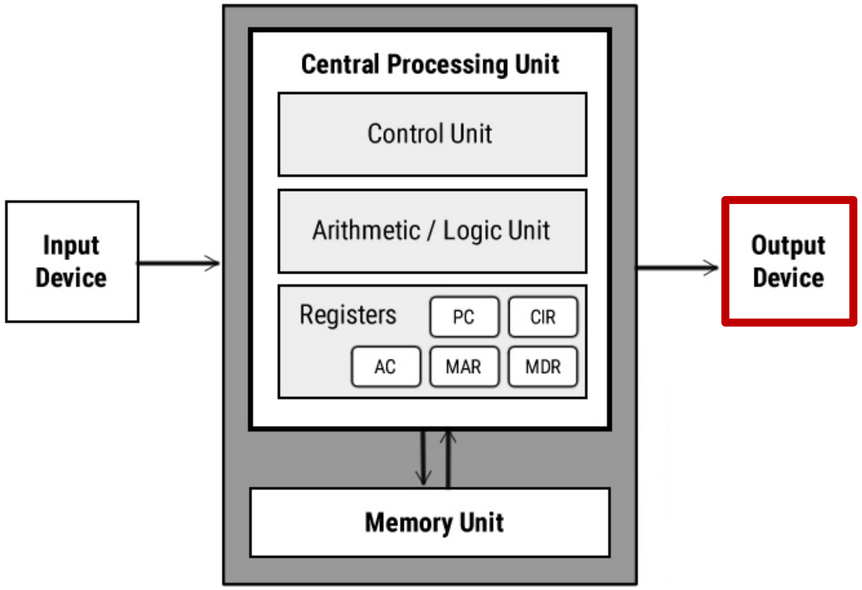
Input unit

- It **provides data and instructions** to the computer system.
- Commonly used input devices are **keyboard, mouse, magnetic tape** etc.
- Input unit performs following tasks:
  - **Accept the data and instructions** from the outside environment.
  - **Convert it into machine language**.
  - **Supply the converted data** to computer system.



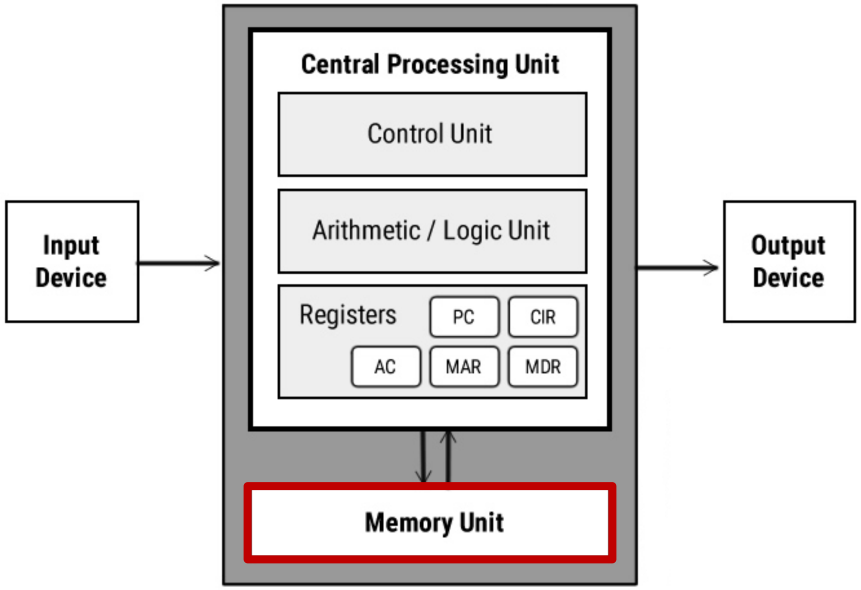
Output unit

- **It connects the internal system of a computer to the external environment**.
- It provides the results of any computation, or instructions to the outside world.
- Some output devices are **printers, monitor** etc.



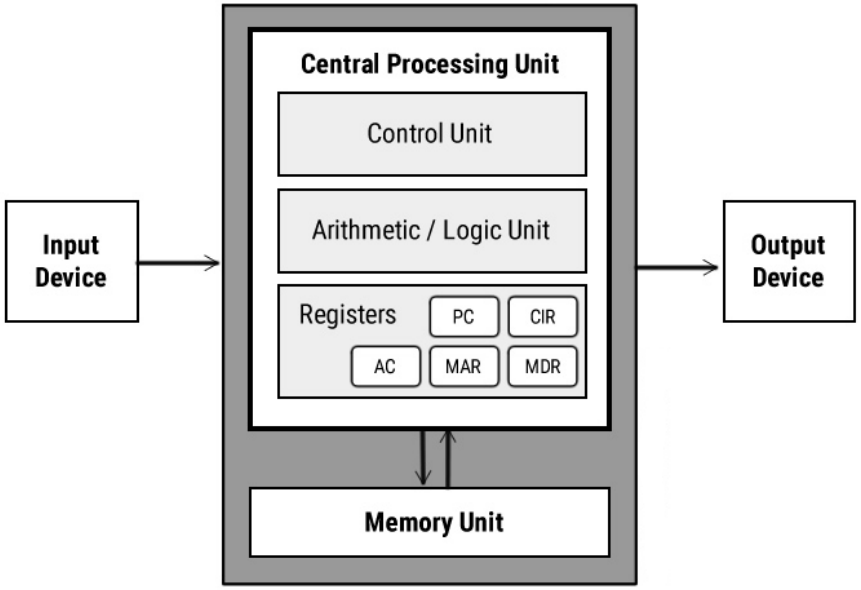
Storage unit (Memory unit)

- This unit **holds the data and instructions**.
- It also **stores the intermediate results** before these are sent to the output devices.
- It also **stores the data for later use**.
- The storage unit of a computer system can be divided into two categories:
  1. Primary Storage
  2. Secondary Storage



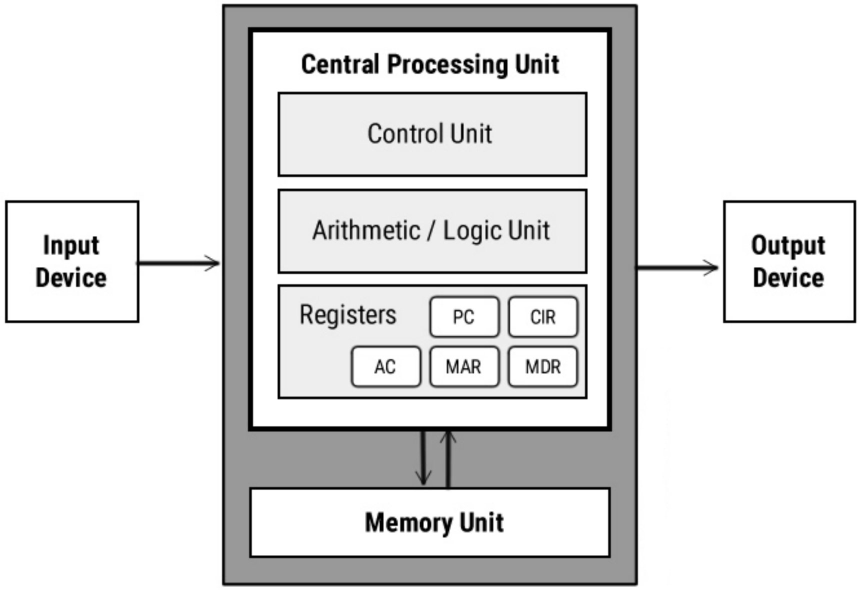
Primary Storage

- This memory is **used to store the data which is being currently executed**.
- It is **used for temporary storage** of data.
- The **data is lost, when the computer is switched off**.
- **RAM** is used as primary storage memory.



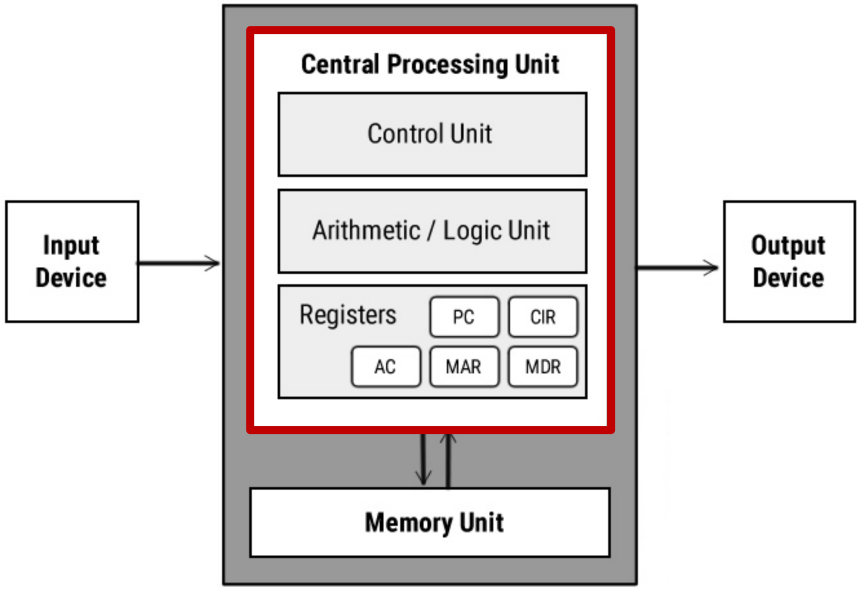
Secondary Storage

- The secondary memory is **slower and cheaper than primary memory**.
- It is **used for permanent storage of data**.
- Commonly used secondary memory devices are **hard disk, CD** etc.



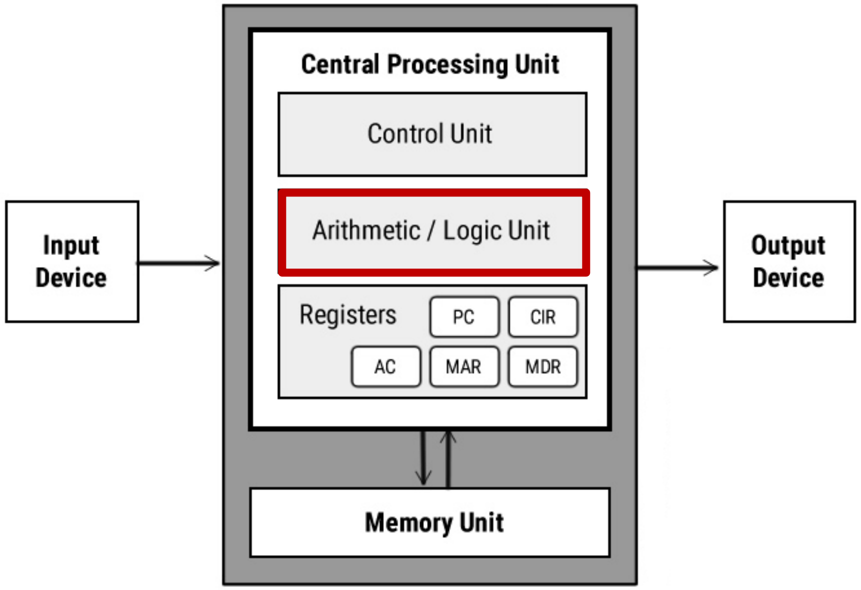
CPU (Central Processing Unit)

- The **Arithmetic Logical Unit** and **Control Unit** are together known as CPU.
- CPU is the **brain of computer system**. It performs following tasks:
  - performs all operations.
  - takes all decisions.
  - controls all the units of computer.



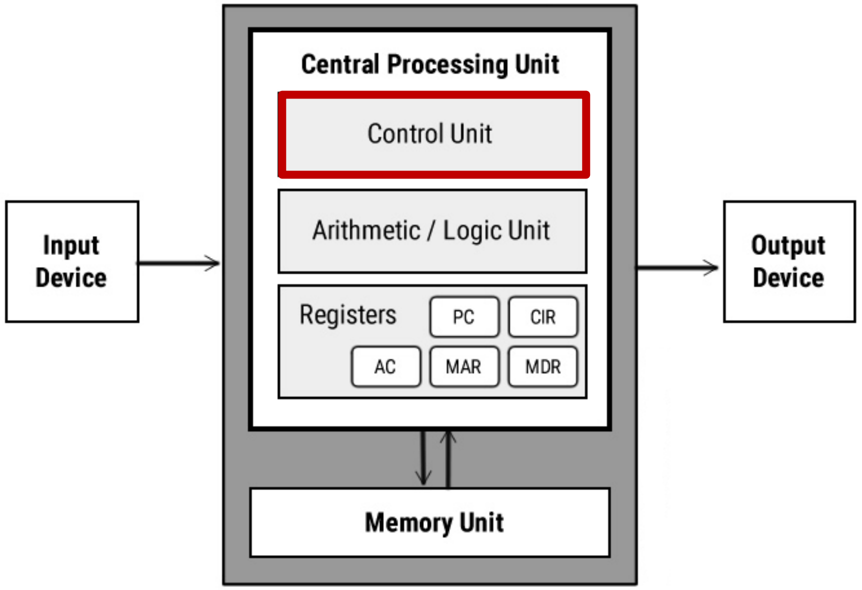
ALU (Arithmetic Logical Unit)

- All the **calculations are performed in ALU** of the computer system.
- The ALU can **perform basic operations such as addition, subtraction, division, multiplication** etc.
- Whenever calculations are required, the **control unit transfers the data from storage unit to ALU**.
- When the operations are done, the result is transferred back to the storage unit.



CU (Control Unit)

- It **controls all other units** of the computer.
- It **controls the flow of data and instructions** to and from the storage unit to ALU.
- Thus it is also known as **central nervous system of the computer**.



What is Operating System (OS)?

- A Computer System consists of various hardwares such as

Processor

RAM

Keyboard & Mouse

Hard Disk

Monitor

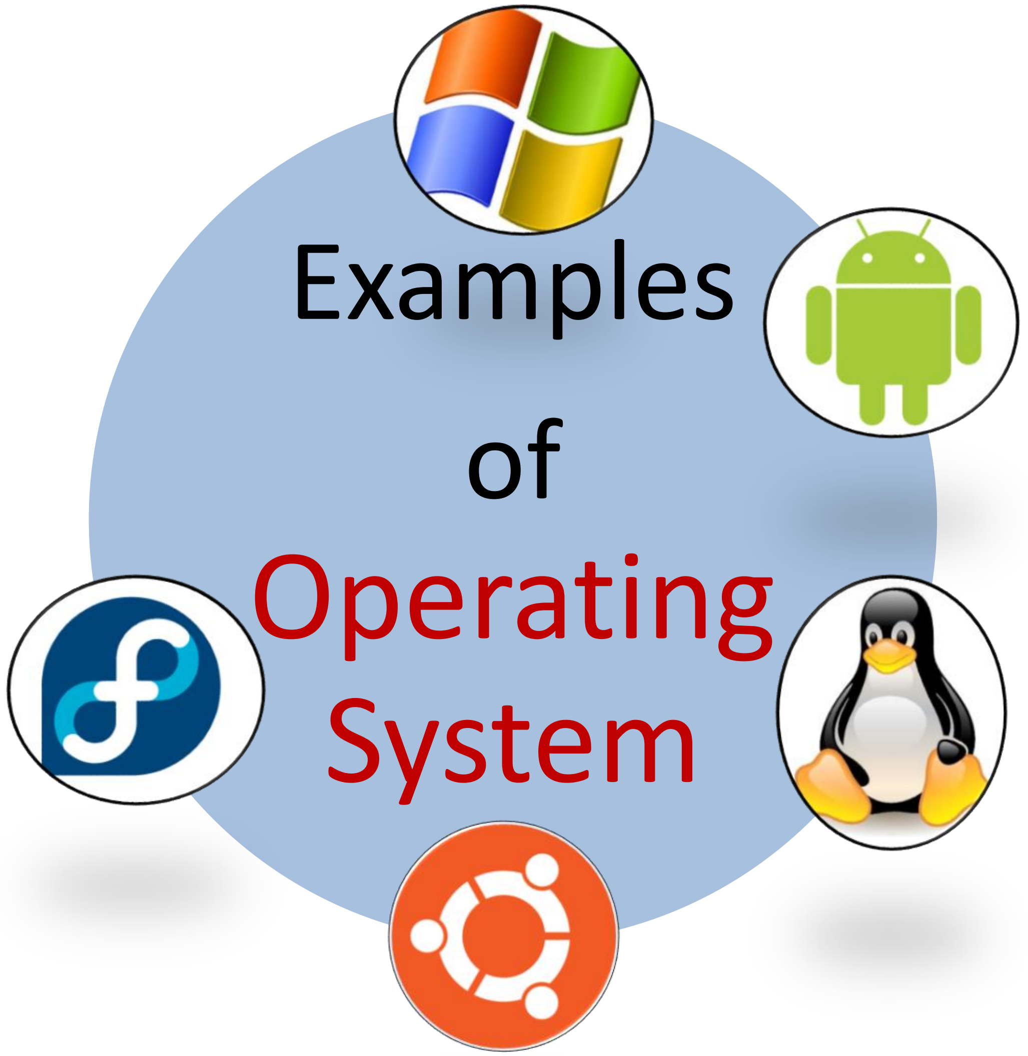
Printer

Who manages (controls) these hardwares??? **Operating System**

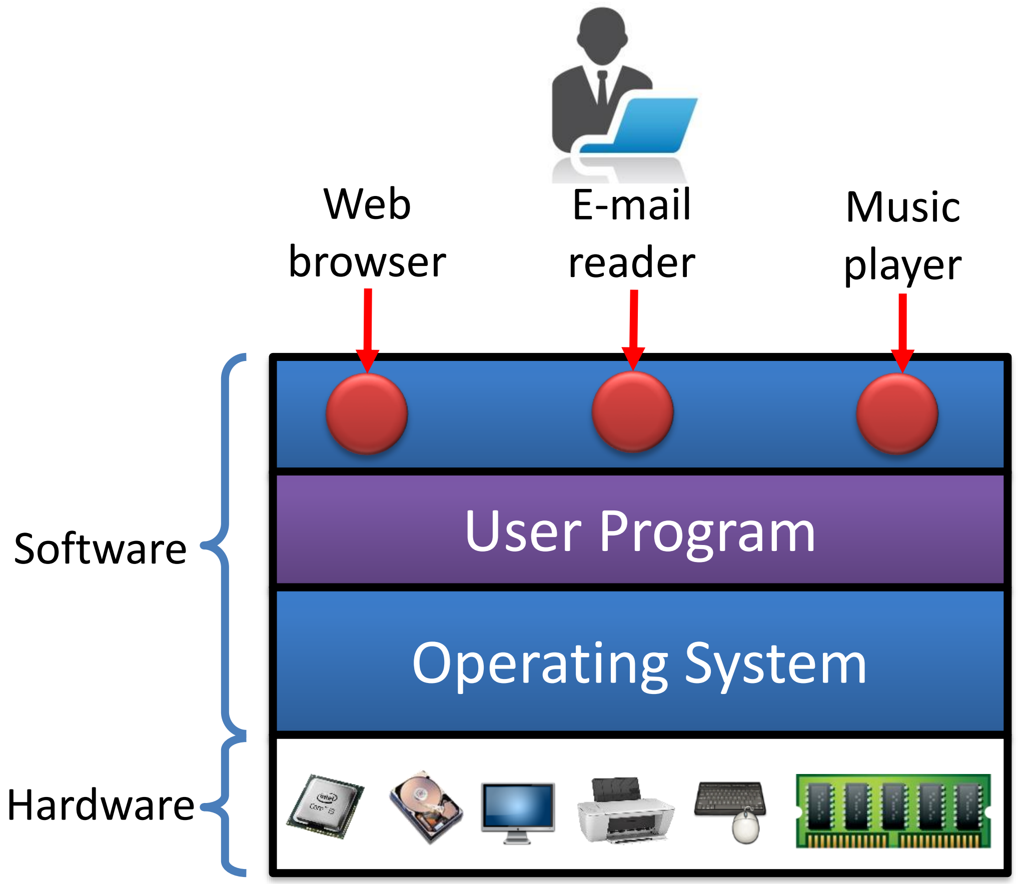
Definition of Operating System

- An Operating System (OS) is a collection of software that
  - manages hardware resources
  - provides various service to the users

Examples of Operating System



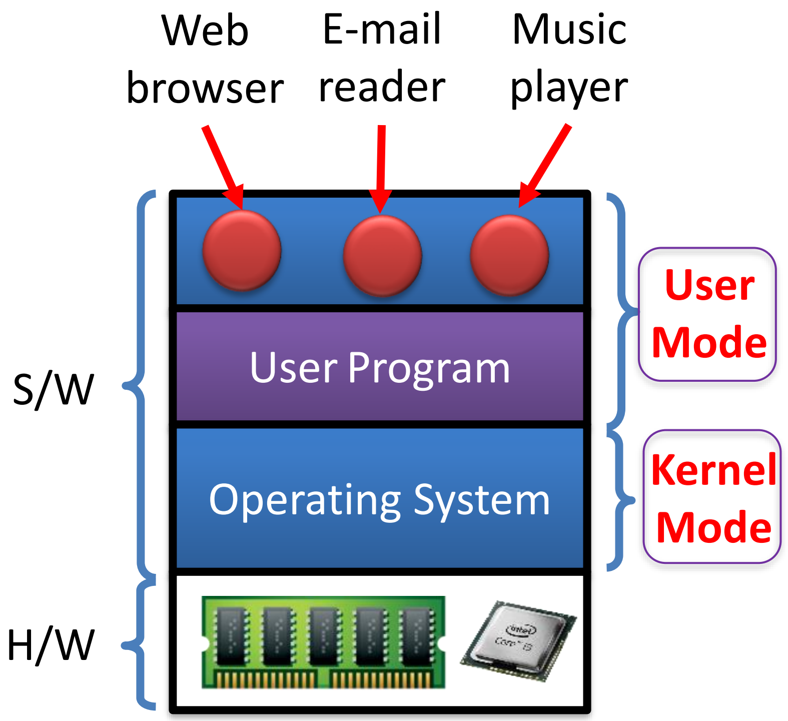
Where OS lies? (Interaction of OS & Hardware)



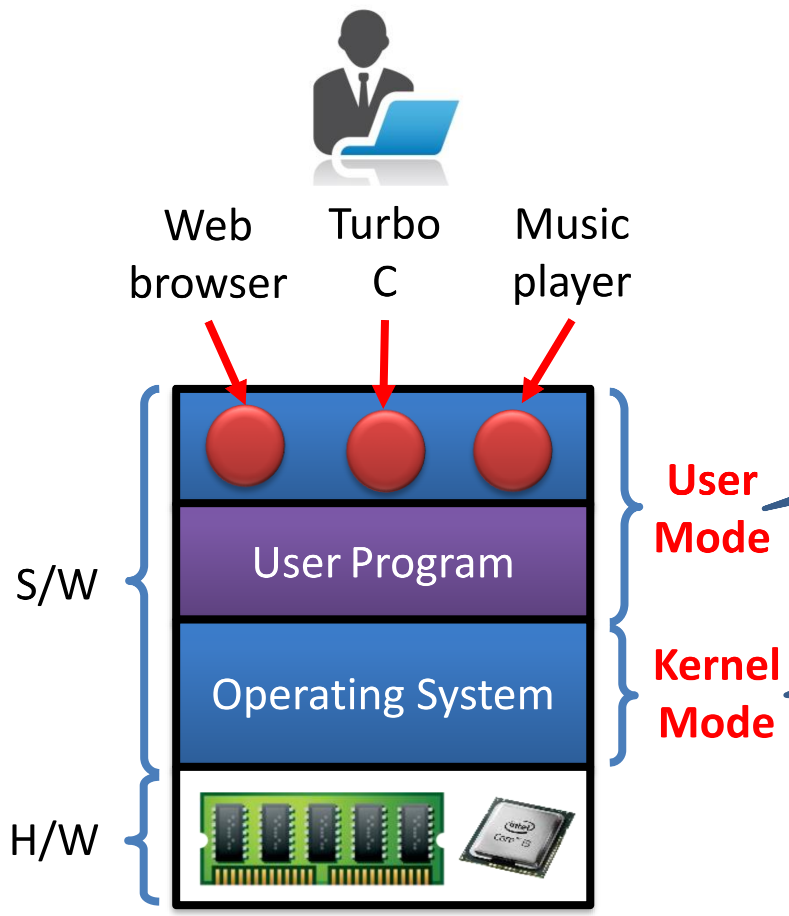
- OS **lies between hardware and user program**.
- It **acts as an intermediary** between the user and the hardware.

Modes of operation of computer

1. **Kernel Mode**
   - has complete access to all the hardware
   - can execute any instruction that a machine is capable of executing
   - has high privileged (rights)
2. **User Mode**
   - can execute only subset (few) of the machine instructions
   - has less privileged (rights)



Why and How switch occur?



- User is writing program in C.
- Once finish writing will execute it.

user-mode program performs **trap** instruction

- To execute this program
  1. Load program into RAM
  2. Program is executed by processor

Roles of OS (OS can be viewed as)

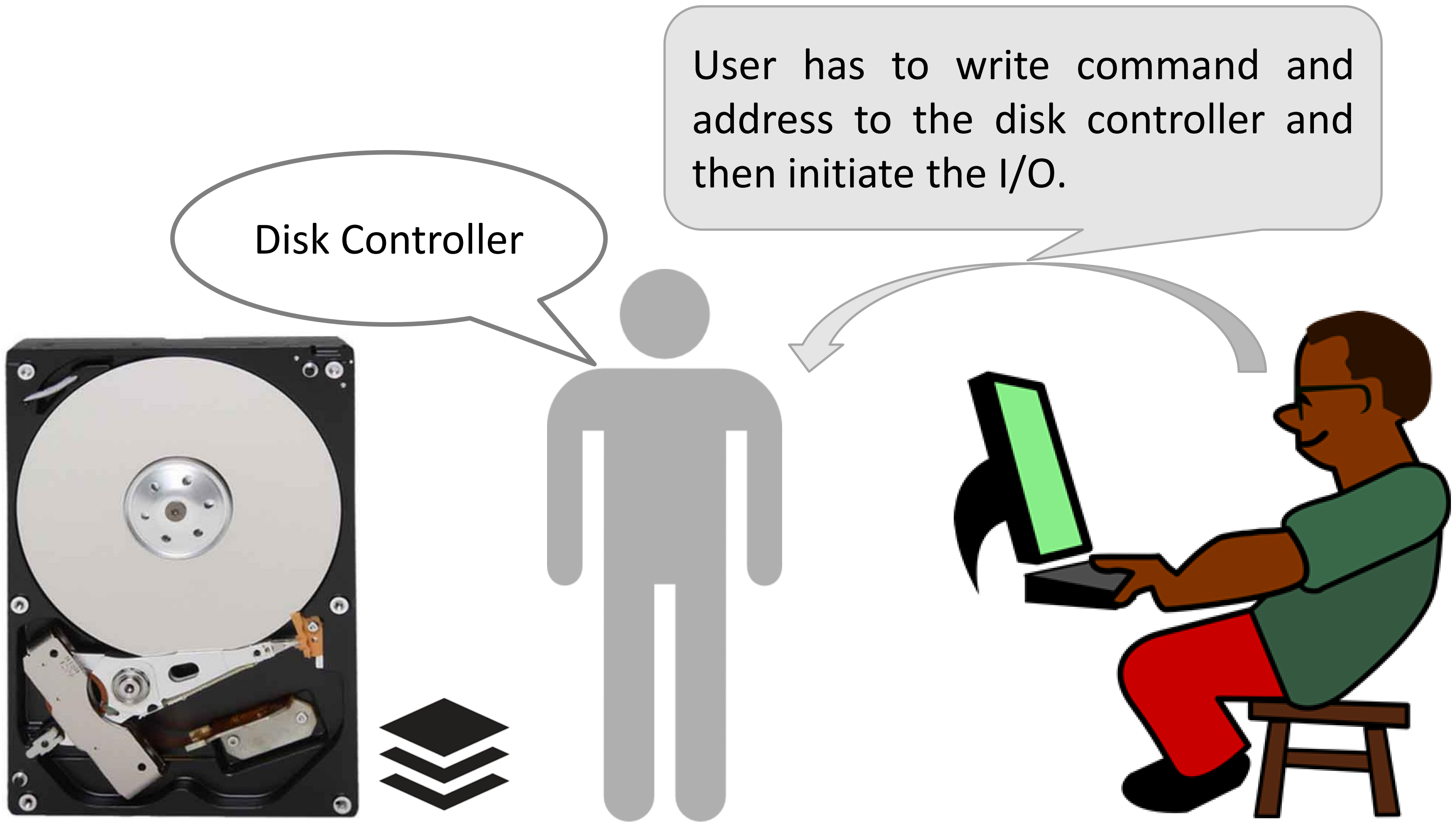
1. The OS as an Extended/Virtual Machine
2. The OS as a Resource Manager

OS as Extended Machine

- The architecture of a computer is difficult to program
  - Architecture (instruction set, memory organization, I/O, bus structure) of most of computer at the machine level language is primitive and awkward to program.
  - Example: If user want to read from floppy or hard disk:

OS as Extended Machine

- Example: If **user want to read from floppy or hard disk**:


User has to write command and address to the disk controller and then initiate the I/O.
Disk Controller

OS as Extended Machine

- Example: If user want to read from floppy or hard disk:

The disk controller will find the requested data in the disk and fetch it from disk to disk controller buffer.

User has to check the status of disk controller operation where it has finished or not.

OS as Extended Machine

- Example: If user want to read from floppy or hard disk:

If success, the data from disk controller buffer should be moved to main memory (to the application buffer).

OS as Extended Machine (Cont…)

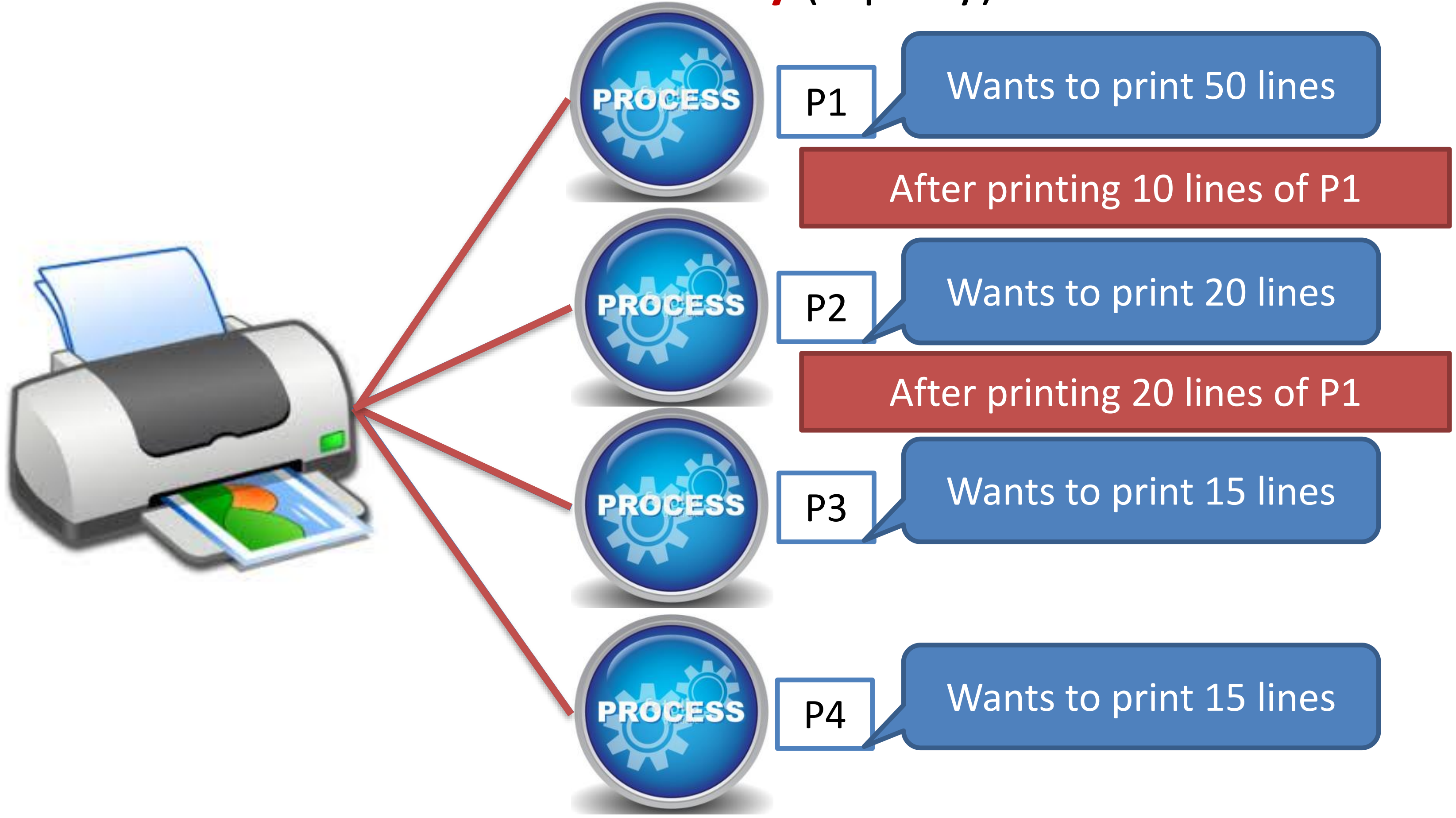
- If all the users will have to do these messy details:
  - The **program will be very difficult to write and quite long**.
  - The **program will be hardware dependent**.
- User **don’t want to be involved in programming** of storage devices.
- Therefore, an **OS provides a set of basic commands** or instructions to perform various operations such as read, write, modify, save or close.
- Dealing with these command is easier than directly dealing with hardware.
- Operating system **hides the complexity of hardware** and **present a beautiful interface** to the users.

OS as Resource Manager

- There are lots of resources in computer system
  - CPU (Processor)
  - Memory
  - I/O devices such as hard disk, mouse, keyboard, printer, scanner etc.
- If a computer system is used by **multiple applications (or users)**, then they will **compete for these resources**.

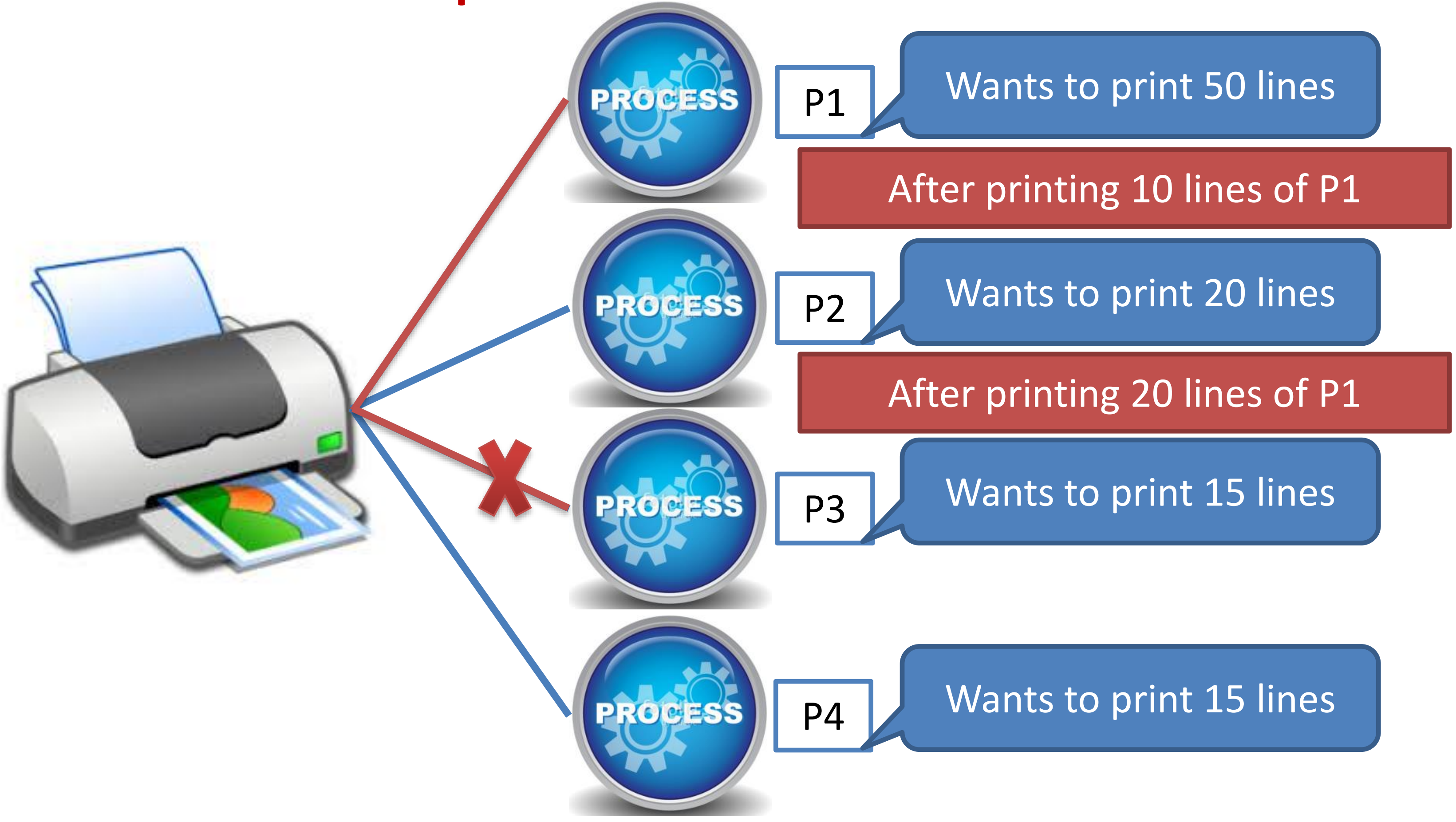
OS as Resource Manager

- It is the job of OS to allocate these resources to the various applications so that:
  - The **resources are allocated fairly** (equally)



OS as Resource Manager

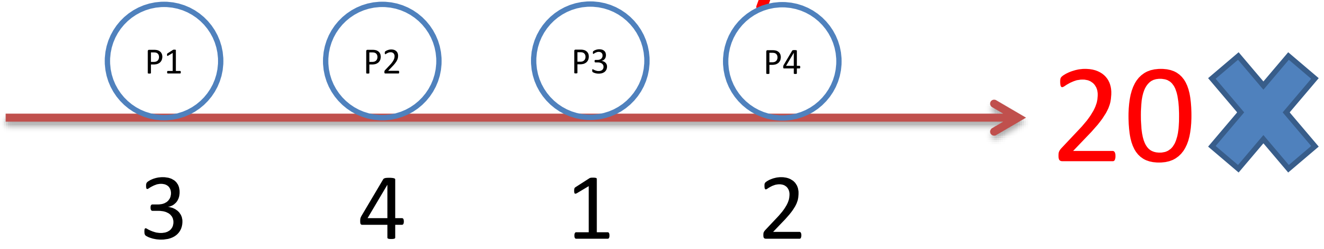
- It is the job of OS to allocate these resources to the various applications so that:
  - The **resources are protected from cross-access**



OS as Resource Manager

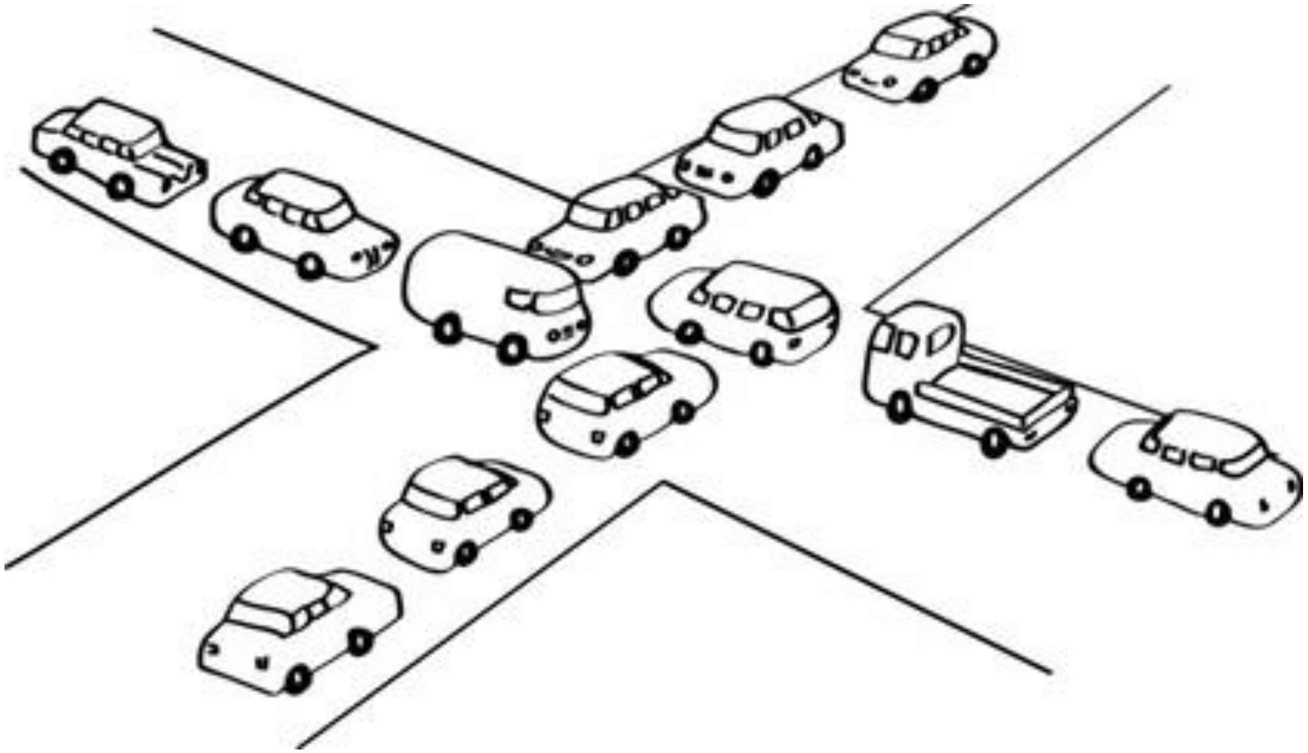
- It is the job of OS to allocate these resources to the various applications so that:
  - Access to the **resources is synchronized** so that operations are correct and consistent
  - Example: If we write a program to calculate below in C language

$$7 + 9 - 6 * 4 / 2 = 4$$



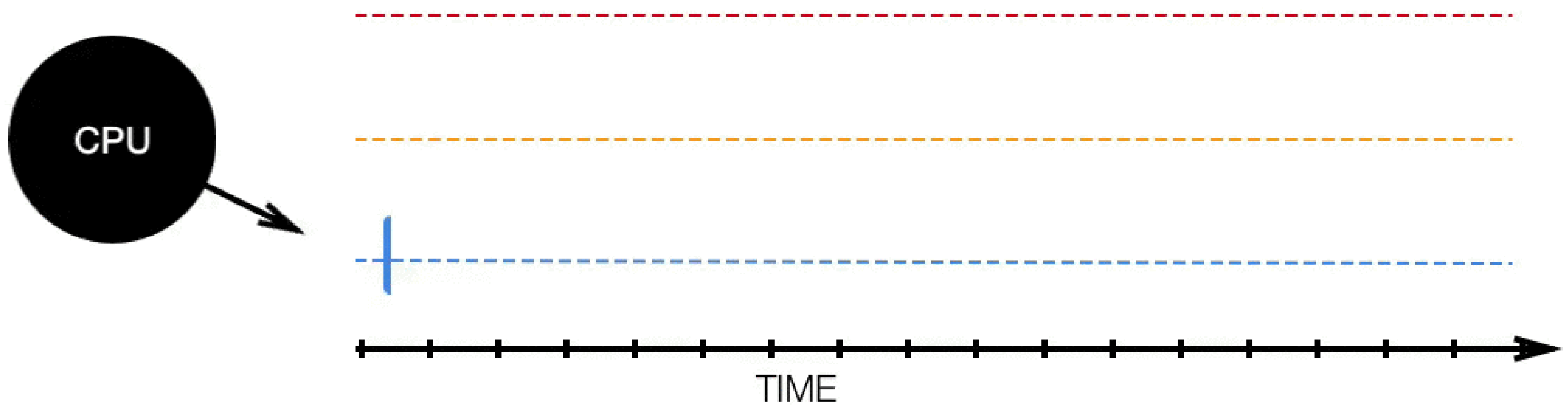
OS as Resource Manager

- It is the job of OS to allocate these resources to the various applications so that:
  - Deadlock are detected, resolved and avoided.



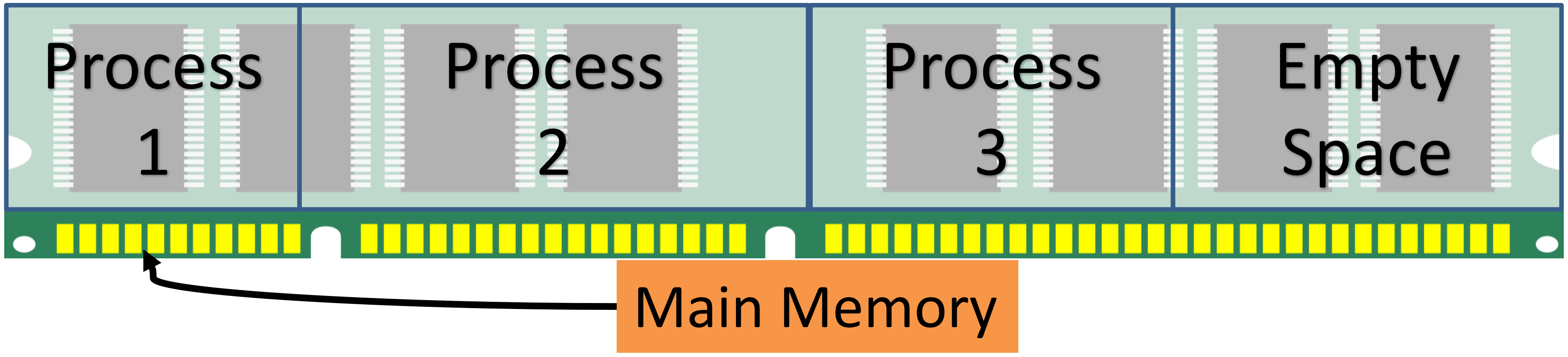
OS as Resource Manager (Cont...)

- Resource manager – sharing resources in two different ways:
  1. In time sharing/multiplexing **(i.e CPU)**



OS as Resource Manager (Cont…)

- Resource manager – sharing resources in two different ways
  2. In space sharing/multiplexing. **(i.e Memory)**



Objectives / Goals of Operating System

- Make the computer system **convenient to use in an efficient manner**.
- **Hide the details of the hardware** resources from the users.
- **Provide** users a **convenient interface** to use the computer system.
- **Act as an intermediary** between the **hardware and its users**, making it easier for the users to access and use other resources.
- **Manage the resources** of a computer system.
- **Keep track** of who is using which resource, granting resource requests, and mediating conflicting requests from different programs and users.
- **Provide efficient** and **fair sharing of resources** among users and programs.

History of OS (First generation)

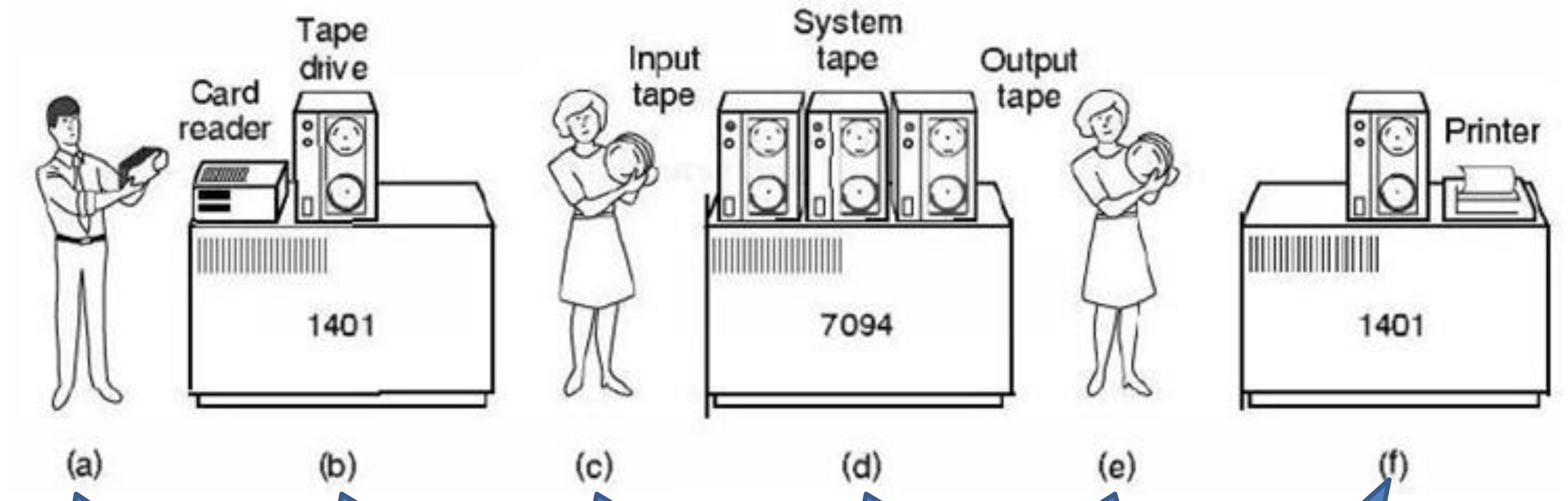
- First generation (1945-1955)
  - **Vacuum tubes and plug-boards** are used in these systems.

Vacuum tubes

Plug board

History of OS (Second generation)

- Second generation (1955-1965)
  - **Transistors** are used in these systems
  - The machine that are produced are called mainframes.
  - **Batch systems** was used for processing.



Programmer 1401 reads ba Operator carri 7094 Operator carries ou 1401 prints output

History of OS (Third generation)

- Third generation (1965-1980)
  - Integrated circuits (**IC's**) are used in place of transistors in these computers.
  - It provides **multiprogramming** (the ability to have several programs in memory at once, each in its own memory partition).

History of OS (Fourth generation)

- Fourth generation (1980-present)
  - Personal Computers
  - **LSI** (**Large Scale Integration**) circuits, chips containing thousands of transistors are used in these systems.

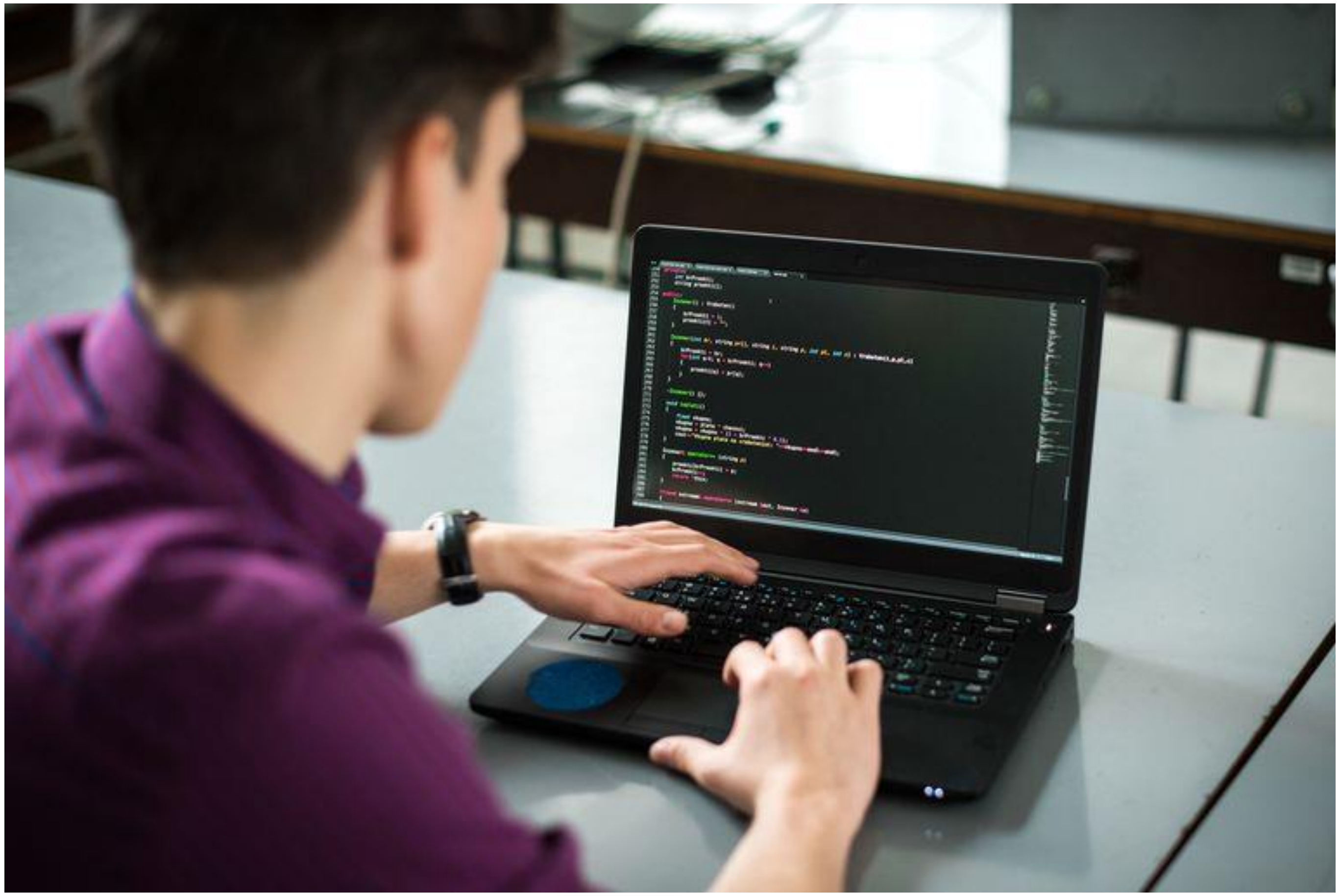
Services / Functions / Tasks of OS

Definition of Operating System

- An Operating System (OS) is a collection of software that
  - manages hardware resources
  - provides various service to the users

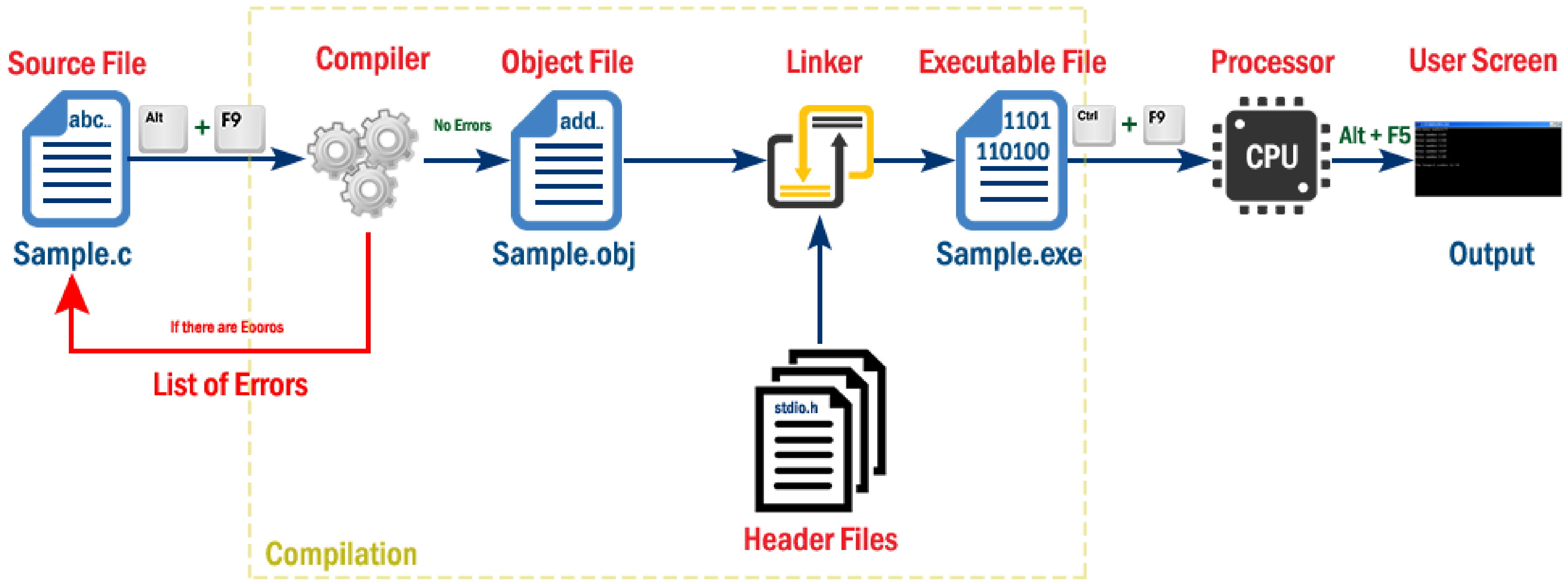
Services / Functions / Tasks of OS

1. Program development
    - It **provides editors** and debuggers to assist (help) the programmer in creating programs.



Services / Functions / Tasks of OS

2. Program execution
   - Following tasks need to be perform to execute a program:
     - ✓ Instructions and data must be **loaded into main memory**.
     - ✓ I/O **devices and files must be initialized**.
   - The OS **handles all these duties** for the user.

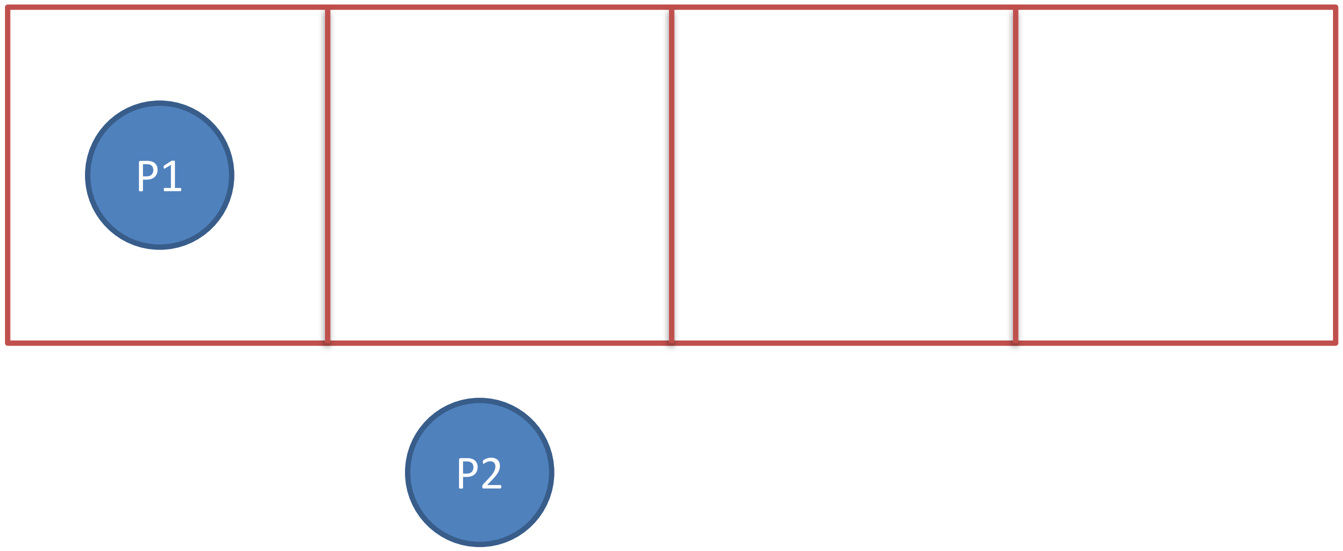


Services / Functions / Tasks of OS

3. Access to I/O devices (Resource allocation)
   - A running program may require I/O, which may involve file or an I/O device.
   - For efficiency and protection, users cannot control I/O devices directly.
   - Therefore, the OS **controls these I/O devices** and **provides to program as per requirement**.

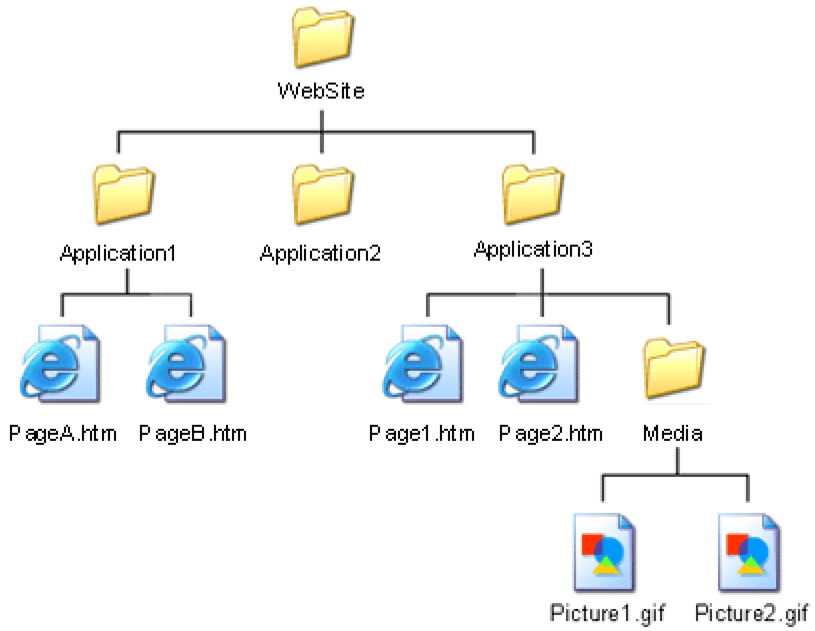
Services / Functions / Tasks of OS

4. Memory management
   - OS **manages memory hierarchy**.
   - OS **keeps the track** of which part of memory area in use in use and free memory.
   - It **allocates memory** to program when they need it.
   - It **de-allocate the memory** when the program finish execution.



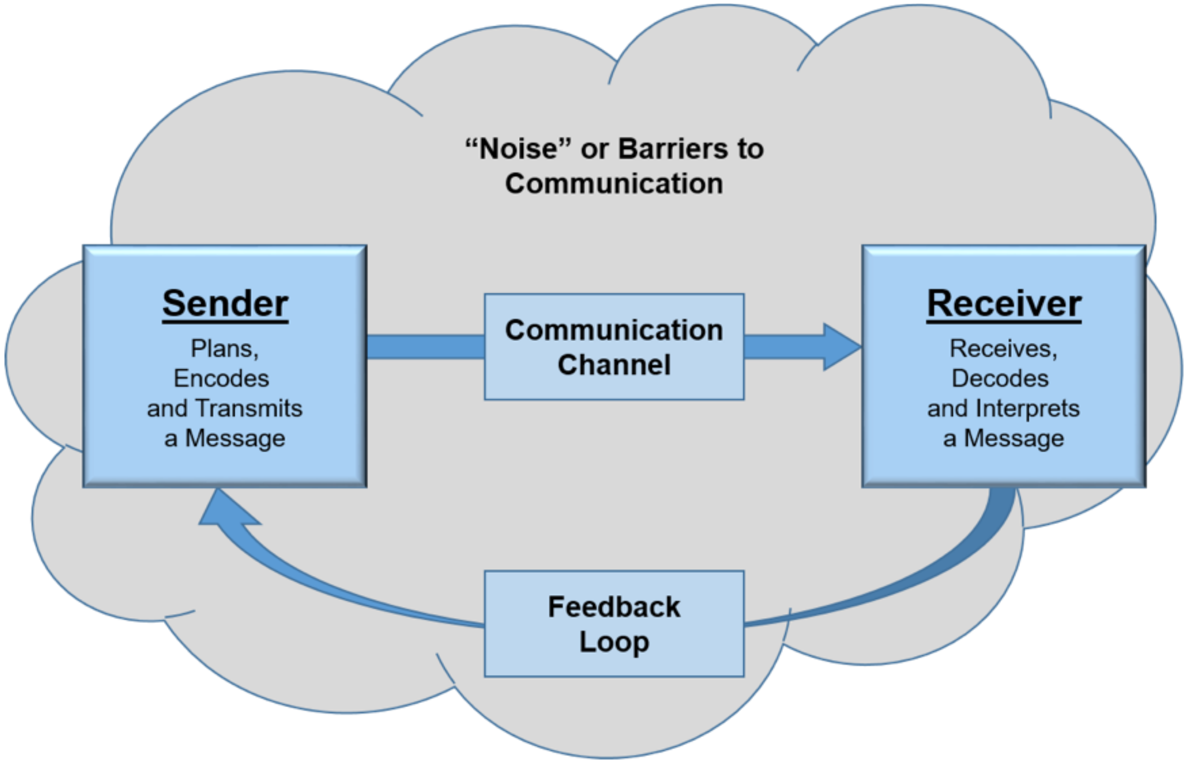
Services / Functions / Tasks of OS

5. Controlled access to file
    - In case of file access, OS **provides a directory hierarchy** for easy access and management of file.
    - OS **provides various file handling commands** using which user can easily read, write and modify file.



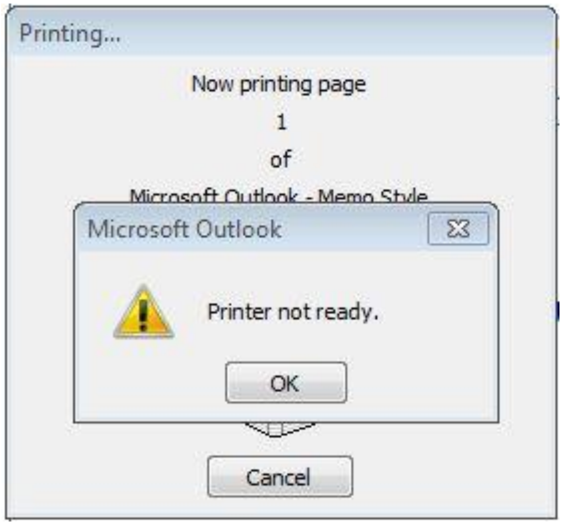
Services / Functions / Tasks of OS

6. Communication
   - In multitasking environment, the processes need to communicate with each other and to exchange their information.
   - Operating system **performs the communication among various types of processes** in the form of shared memory.


"Noise" or Barriers to Communication
Sender
Plans, Encodes and Transmits a Message
Communication Channel
Receiver
Receives, Decodes and Interprets a Message
Feedback Loop

Services / Functions / Tasks of OS

7. Error detection and response
   - An error may occur in CPU, in I/O devices or in the memory hardware.
   - Following are the major activities of an operating system with respect to error handling –
     - ✓ The OS **constantly checks for possible errors**.
     - ✓ The OS **takes an appropriate action** to ensure **correct and consistent** computing.


Printing...
Now printing page
1
of
Microsoft Outlook - Memo Style
Microsoft Outlook
Printer not ready.
OK
Cancel

Services / Functions / Tasks of OS

8. Accounting
   - **Keeping a track of which users are using how much and what kinds of computer resources** can be used for accounting or simply for accumulating usage statistics.
   - Usage statistics is used to reconfigure the system to improve computing services.

Services / Functions / Tasks of OS

9. Protection & Security
    - Protection involves **ensuring that all accesses to system resources is controlled**.
    - To make a system secure, the user needs to authenticate himself or herself to the system.

Services / Functions / Tasks of OS (Revision)

1. Program development
2. Program execution
3. Access to I/O devices
4. Memory management
5. Controlled access to file
6. Communication
7. Error detection and response
8. Accounting
9. Protection & Security

TYPES OF OS [1]

Distinguished by the response time and how data is entered into the system

- Single user
- Multi user
- Multitasking
- Multi processing
- Embedded
- Real time

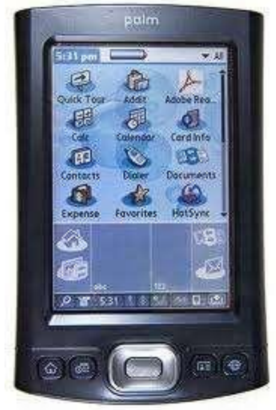
[1] SINGLE USER [1]

TWO TYPES:

- Single user, single task
- Single user, multi tasking

Single user, single task

- Designed to manage the computer so that one user can effectively do one thing at a time.
- Example: The Palm OS for Palm handheld computers



Single user, multi tasking

- Designed with a single user in mind but can deal with many applications running at the same time
- Type of operating system most people use on their desktop and laptop computers today

CONT…..

- Examples: Microsoft's Windows and Apple's Mac OS platforms

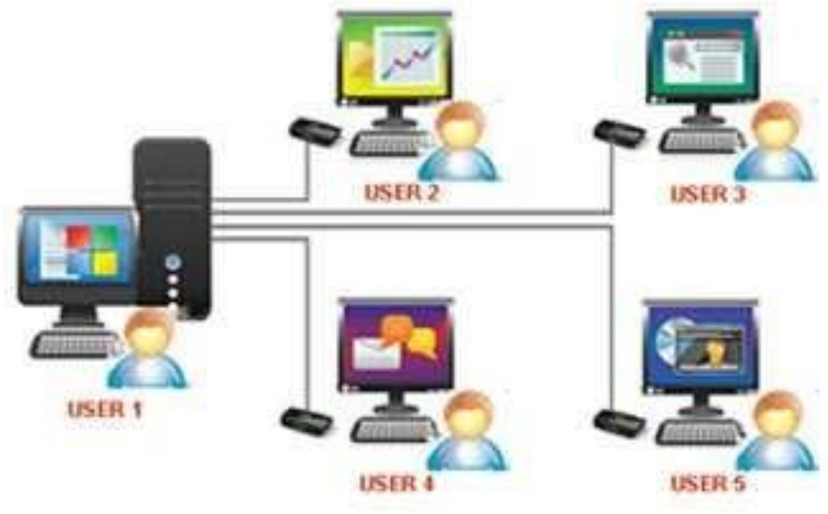
- For Example: It's entirely possible for a Windows user to be writing a note in a word processor while downloading a file from the Internet while printing the text of an e-mail message.

[2] MULTI USER [2]

- Allows many different users to take advantage of the computer's resources simultaneously
- Allows multiple users to access the computer system at the same time
- Time Sharing system and Internet servers as the multi user systems

CONT…..

Examples: UNIX, VMS and Mainframe Operating systems



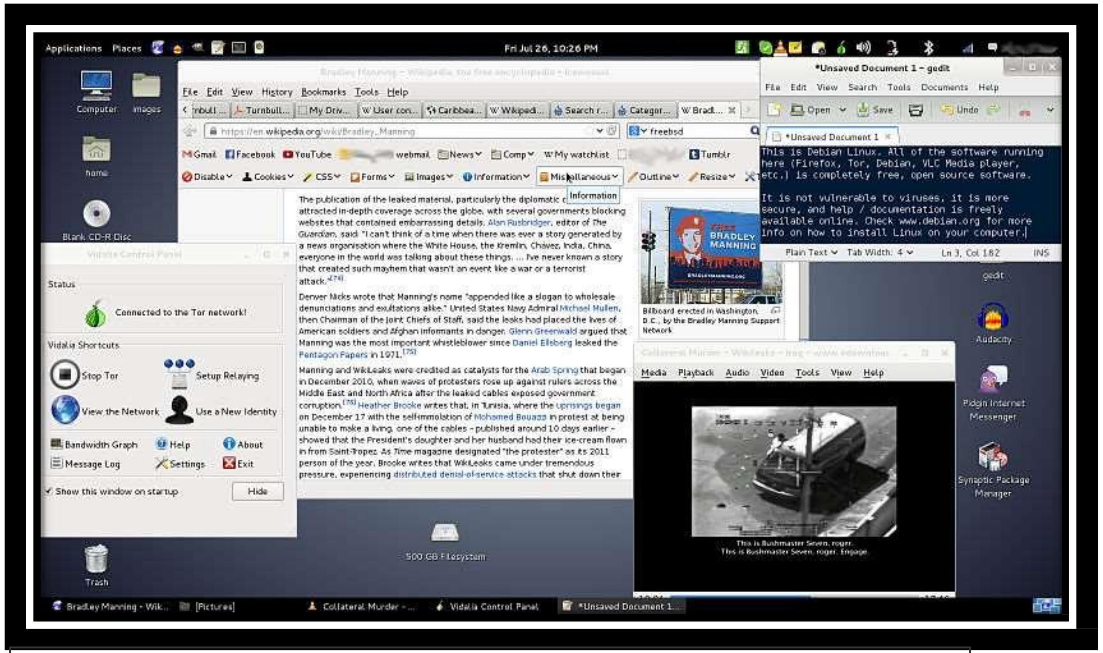
[3] MULTI TASKING [3]

Allows more than one program to run concurrently.

The tasks share common processing resources, such as a CPU and main memory

In the process, only one CPU is involved, but it switches from one program to another so quickly that it gives the appearance of executing all the programs at the same time.

CONT…..



The publication of the leaked material, particularly the diplomatic c... attracted in-depth coverage across the globe, with several governments blocking websites that contained embarrassing details. Alan Rusbridger, editor of *The Guardian*, said: "I can't think of a time when there was ever a story generated by a news organisation where the White House, the Kremlin, Chávez, India, China, everyone in the world was talking about these things. ... I've never known a story that created such mayhem that wasn't an event like a war or a terrorist attack."[74]

Denver Nicks wrote that Manning's name "appended like a slogan to wholesale denunciations and exultations alike." United States Navy Admiral Michael Mullen, then Chairman of the Joint Chiefs of Staff, said the leaks had placed the lives of American soldiers and Afghan informants in danger. Glenn Greenwald argued that Manning was the most important whistleblower since Daniel Ellsberg leaked the Pentagon Papers in 1971.[75]

Manning and WikiLeaks were credited as catalysts for the Arab Spring that began in December 2010, when waves of protesters rose up against rulers across the Middle East and North Africa after the leaked cables exposed government corruption.[76] Heather Brooke writes that, in Tunisia, where the uprisings began on December 17 with the self-immolation of Mohamed Bouazizi in protest at being unable to make a living, one of the cables – published around 10 days earlier – showed that the President's daughter and her husband had their ice-cream flown in from Saint-Tropez. As *Time* magazine designated "the protester" as its 2011 person of the year, Brooke writes that WikiLeaks came under tremendous pressure, experiencing distributed denial-of-service attacks that shut down their

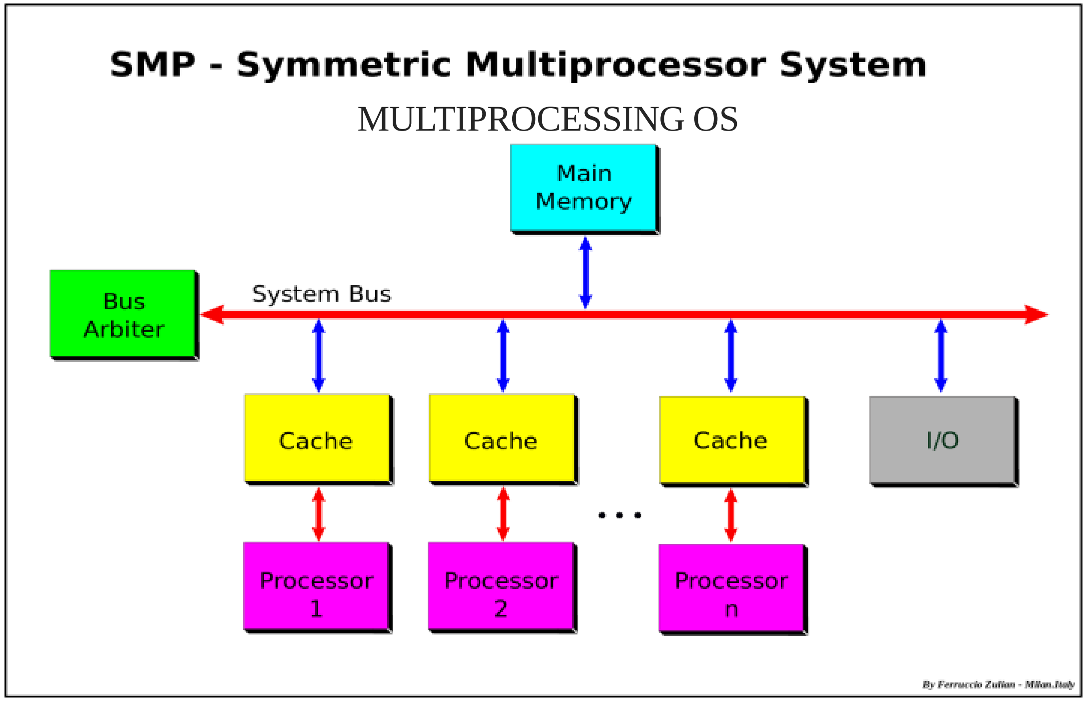
```
This is Debian Linux. All of the software running
here (Firefox, Tor, Debian, VLC Media player,
etc.) is completely free, open source software.

It is not vulnerable to viruses, it is more
secure, and help / documentation is freely
available online. Check www.debian.org for more
info on how to install Linux on your computer.
```

screenshot of Debian Linux (version 7.1, "Wheezy") running the GNOME desktop environment, Firefox, Tor, and VLC media player, all at the same time.

[4] MULTI PROCESSING [3]

- Multiprocessing, in general, refers to the utilization of multiple CPUs in a single computer system
- Enables several programs to run concurrently
- The term also refers to the ability of a system to support more than one processor and/or the ability to allocate tasks between them

CONT.....


SMP - Symmetric Multiprocessor System
MULTIPROCESSING OS
Main Memory
Bus Arbiter
System Bus
Cache
Cache
Cache
I/O
...
Processor 1
Processor 2
Processor n
By Ferruccio Zulian - Milan.Italy

[5] EMBEDDED OS [4][5]

- Designed to be used in embedded computer systems
- Are able to operate with a limited number of resources on small machines like PDAs
- Are very compact and extremely efficient by design
- is a computer that is part of a different kind of machine
- Examples include computers in cars, digital televisions, ATMs, airplane controls, digital cameras, GPS navigation systems, elevators, and among many other possibilities.

CONT……

Embedded OS in a car

Android OS in digital camera

[6] REAL TIME OPERATING SYSTEM [6]

- is a multitasking operating system that aims at executing real-time applications
- The main objective of real-time operating systems is their quick and predictable response to events
- In it, the time interval required to process and respond to inputs is so small that it controls the environment

CONT….

- Examples: QNX, RTLINUX
- Are used to control machinery, scientific instruments and industrial systems

CONT…..

virtualization and real-time systems
AEROSPACE & DEFENSE
AVIONICS
MEDICAL
SECURE CLIENT
NETWORK SECURITY
EMBEDDED DEVICES

System calls

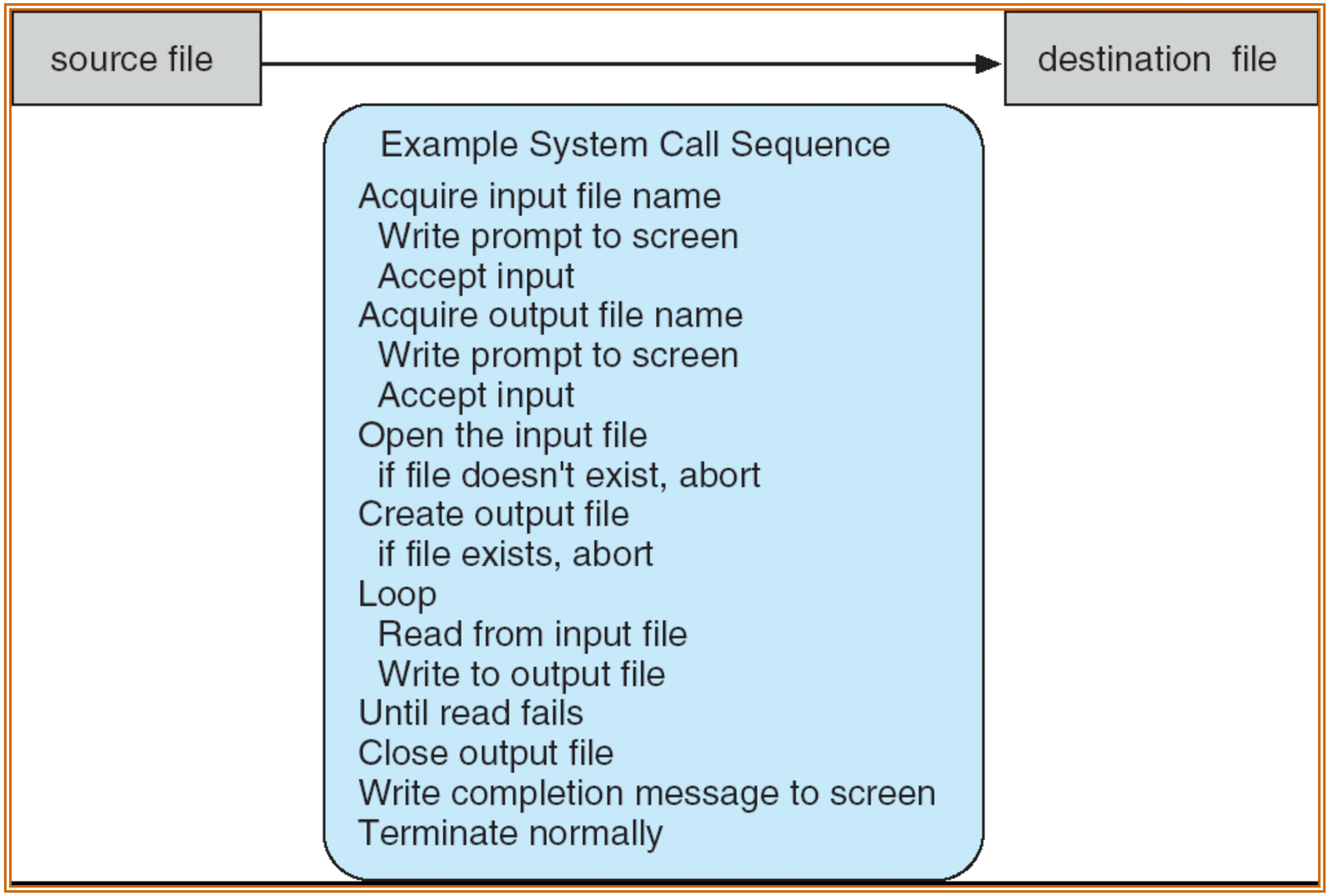
- The **interface between OS and user program is defined by the set of system calls** that OS provides.
- System calls vary from OS to OS. (Note that the system-call names used throughout this text are generic)
- **Example**: In Unix Read system call is
  - count=read(fd, buffer, nbytes)
    - fd is a file descriptor.
      - When a file is opened, permissions are checked.
      - If access is allowed, a number (fd) is returned.
      - Then file can be read/written.
    - nbytes is number of bytes to read
    - buffer is where read deposits (stores) the data
- Typically written in a high-level language (C or C++)
-

System calls

- Mostly accessed by programs via a high-level **Application Program Interface (API)** rather than direct system call use
- Three most common APIs are
  - Win32 API for Windows,
  - POSIX (The Portable Operating System Interface) API for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X),
  - and Java API for the Java virtual machine (JVM)
- **Why use APIs rather than system calls?**
- Application developers often do not have direct access to the system calls, but can access them through an application programming interface (API). The functions that are included in the API invoke the actual system calls. By using the API, certain benefits can be gained:
  - Portability: as long a system supports an API, any program using that API can compile and run.
  - Ease of Use: using the API can be significantly easier than using the actual system call.

Example of System Calls

- System call sequence to copy the contents of one file to another file

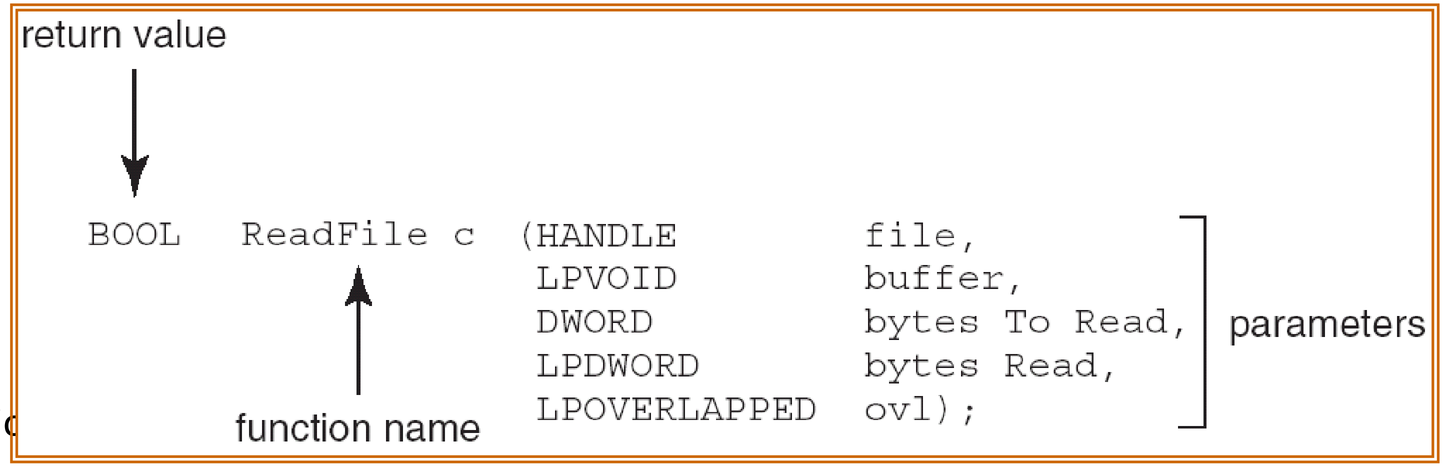


Example System Call Sequence

```
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 doesn't 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
```

Example of Standard API

- Consider the ReadFile() function in the
- Win32 API—a function for reading from a file



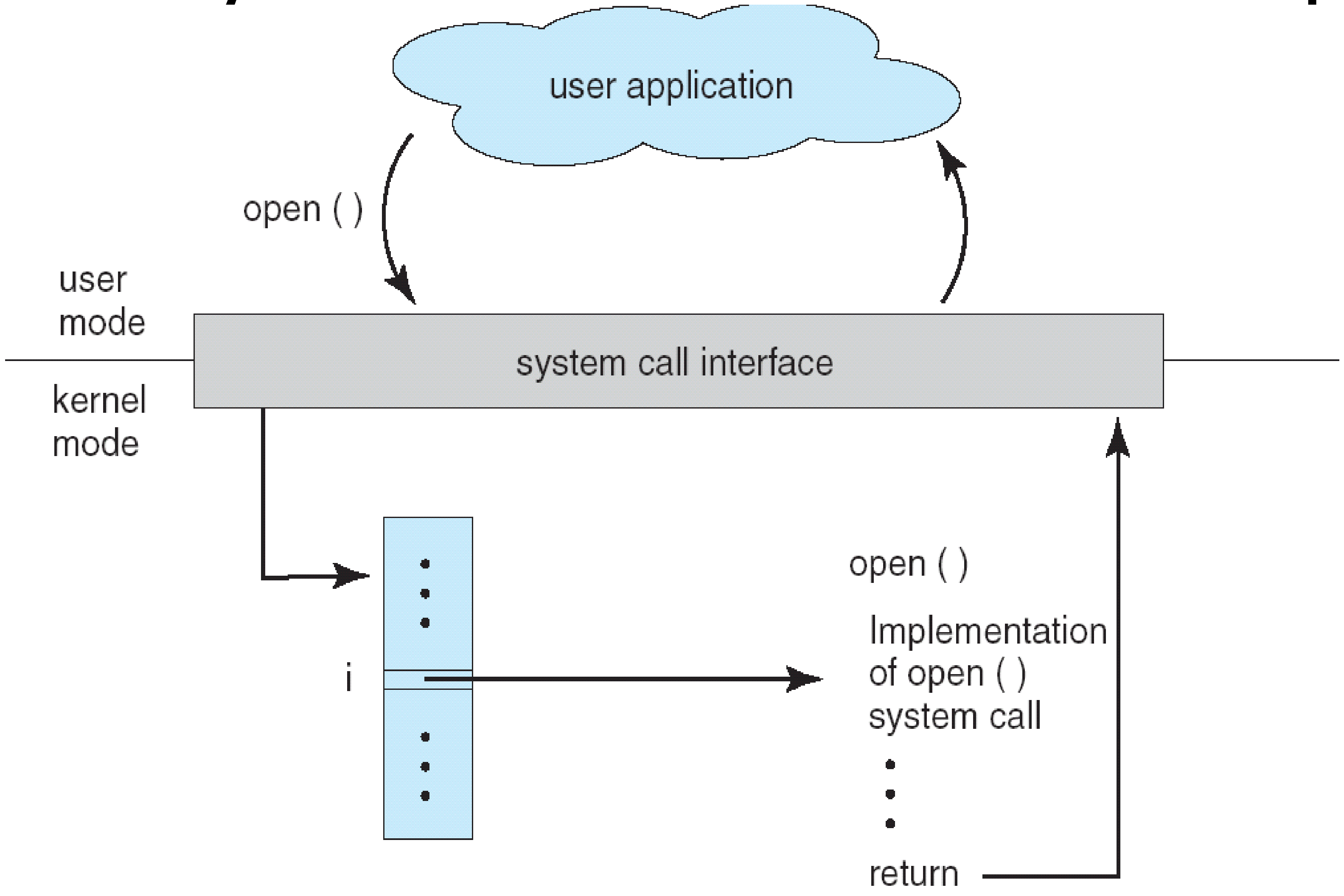
```
BOOL   ReadFile c  (HANDLE        file,
                    LPVOID        buffer,
                    DWORD         bytes To Read,
                    LPDWORD       bytes Read,
                    LPOVERLAPPED  ovl);
```

- A descripti
  - HANDLE file—the file to be read
  - LPVOID buffer—a buffer where the data will be read into and written from
  - DWORD bytesToRead—the number of bytes to be read into the buffer
  - LPDWORD bytesRead—the number of bytes read during the last read
  - LPOVERLAPPED ovl—indicates if overlapped I/O is being used

System Call Implementation

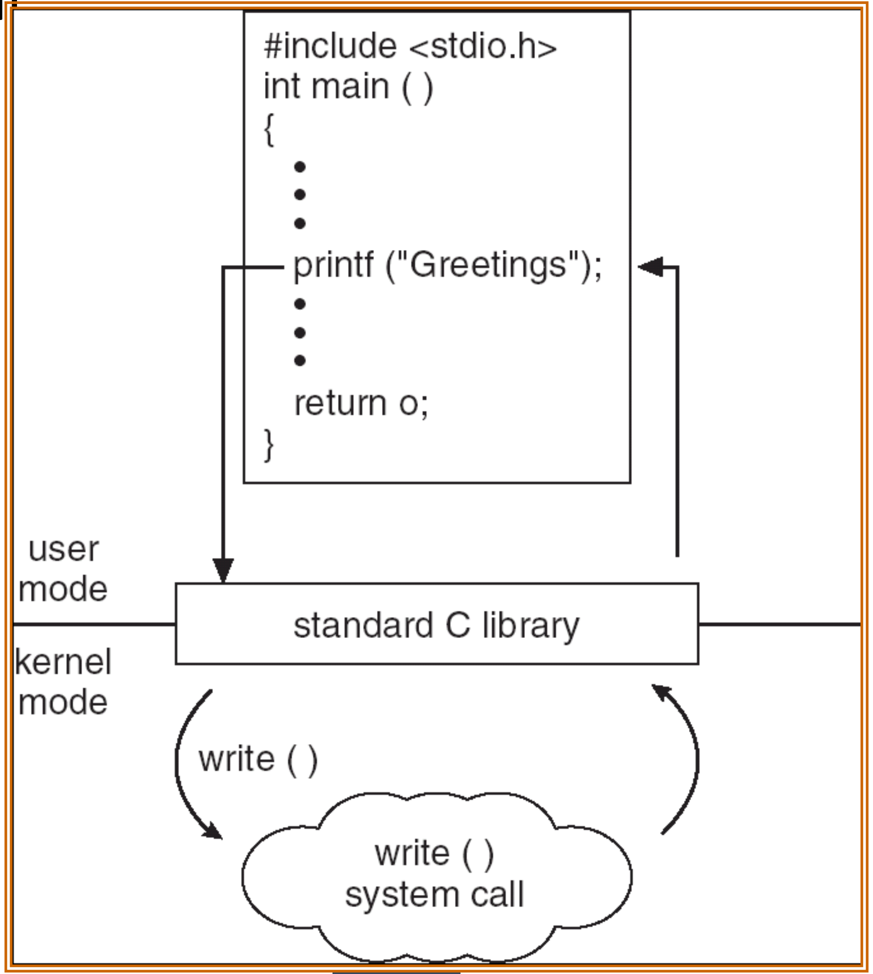
- Typically, a number associated with each system call
  - System-call interface maintains a table indexed according to these numbers
- The system call interface invokes intended system call in OS kernel and returns status of the system call and any return values
- The caller need know nothing about how the system call is implemented
  - Just needs to obey API and understand what OS will do as a result call
  - Most details of OS interface hidden from programmer by API
    - Managed by run-time support library (set of functions built into libraries included with compiler)

API – System Call – OS Relationship



Standard C Library Example

- C program invoking printf() library call, which calls write() system call



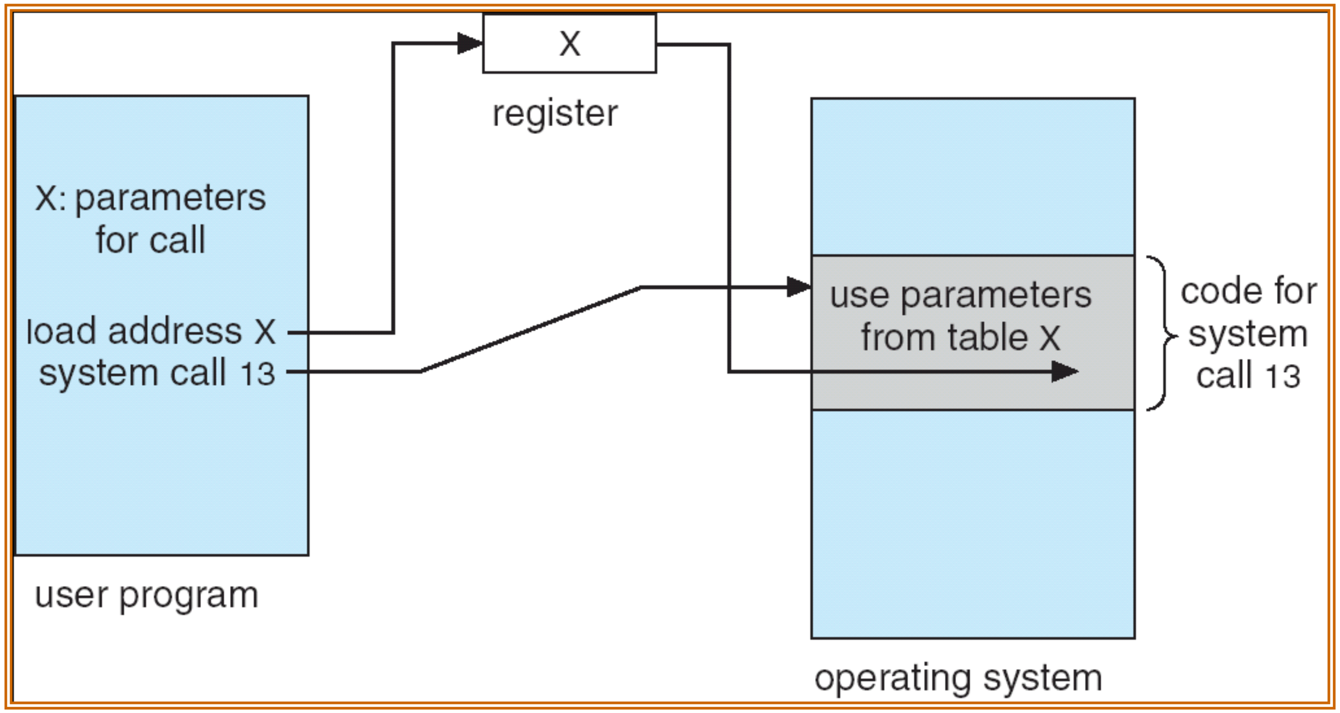
```
#include <stdio.h>
int main ( )
{
   •
   •
   •
   printf ("Greetings");
   •
   •
   •
   return o;
}
```

System Call Parameter Passing

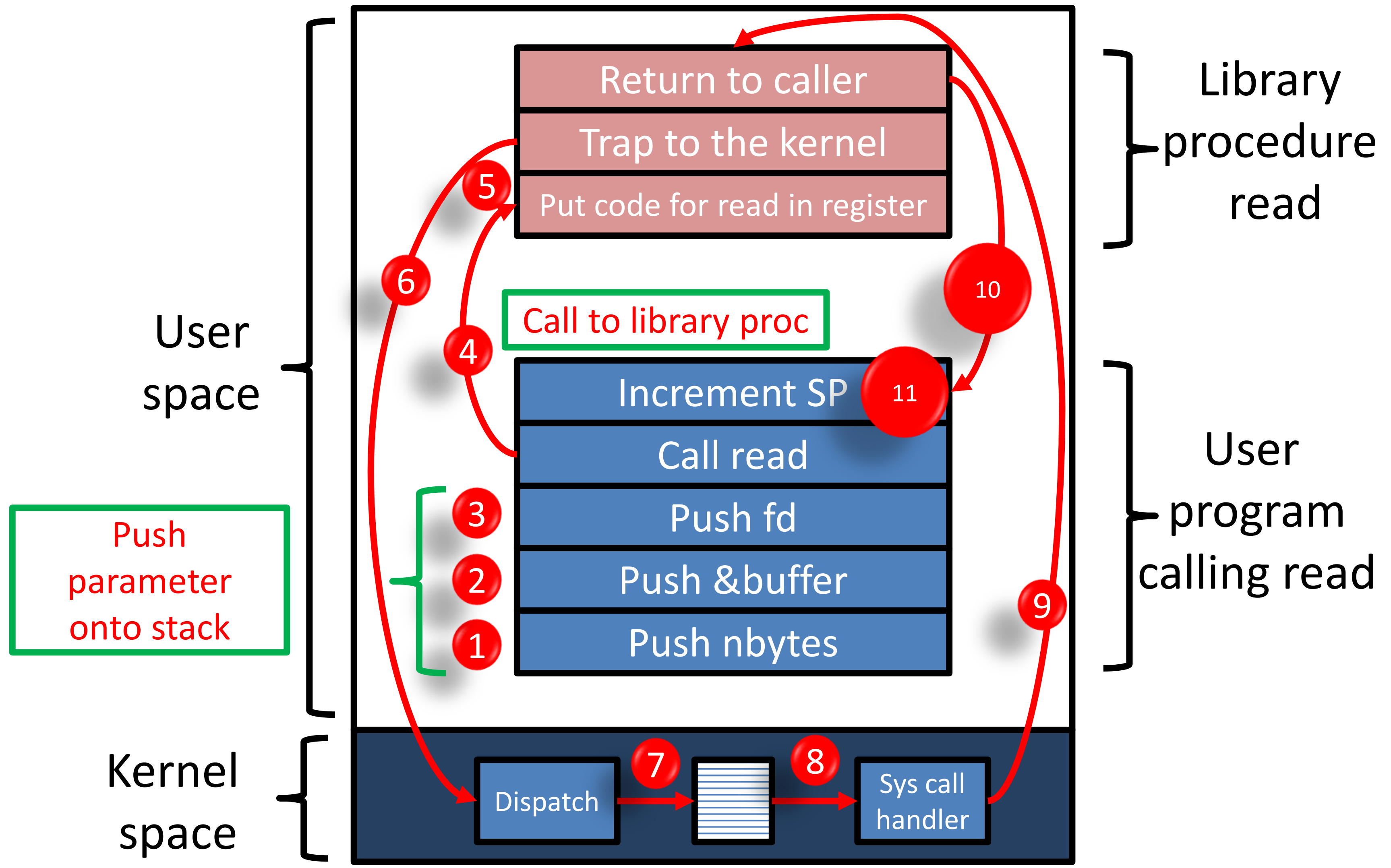
- Often, more information is required than simply identity of desired system call
  - Exact type and amount of information vary according to OS and call
- Three general methods used to pass parameters to the OS
  - **Simplest: pass the parameters in *registers***
    - In some cases, may be more parameters than registers
  - **Parameters stored in a *block,* or table, in memory, and address of block passed as a parameter in a register**
    - This approach taken by Linux and Solaris
  - **Parameters placed, or *pushed,* onto the *stack* by the program and *popped* off the stack by the operating system**

  Block and stack methods do not limit the number or length of parameters being passed

Parameter Passing via Table



System calls (Cont…)



Types of System Calls

- Process control
- File management
- Device management
- Information maintenance
- Communications

System Calls (Cont…)

Director and file system management	
Call	Description
s = mkdir(name,mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system
Miscellaneous	
Call	Description
s = chdir(dir name)	Change the working directory
s = chmod(name,mode)	Change a file’s protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

System Calls (Cont...)

Process management	
Call	**Description**
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

File management	
Call	**Description**
fd = open(file, how, ...)	Open a file for reading, writing, or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

System Programs

- System programs provide a convenient environment for program development and execution.
- Some of them are simply user interfaces to system calls; others are considerably more complex
- Can be divided into:
  - File manipulation
  - File modification
  - Programming language support
  - Program loading and execution
  - Communications
  - Application programs

System Programs

- Status information
  - Some ask the system for info - date, time, amount of available memory, disk space, number of users
  - Others provide detailed performance, logging, and debugging information
  - Typically, these programs format and print the output to the terminal or other output devices
  - Some systems implement a registry - used to store and retrieve configuration information

- Most users' view of the operation system is defined by system programs, not the actual system calls

System Programs (cont’d)

- File modification
  - Text editors to create and modify files
  - Special commands to search contents of files or perform transformations of the text
- Programming-language support - Compilers, assemblers, debuggers and interpreters sometimes provided
- Program loading and execution- Absolute loaders, relocatable loaders, linkage editors, and overlay-loaders, debugging systems for higher-level and machine language
- Communications - Provide the mechanism for creating virtual connections among processes, users, and computer systems
  - Allow users to send messages to one another’s screens, browse web pages, send electronic-mail messages, log in remotely, transfer files from one machine to another

Operating System Design and Implementation

- Design and Implementation of OS not “solvable”, but some approaches have proven successful
- Internal structure of different Operating Systems can vary widely
- Start by defining goals and specifications
- Affected by choice of hardware, type of system
- *User* goals and *System* goals
  - 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

Operating System Design and Implementation (Cont.)

- Important principle to separate

  Policy: What will be done?
  Mechanism: How to do it?

- Mechanisms determine how to do something, policies decide what will be done
  - The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later

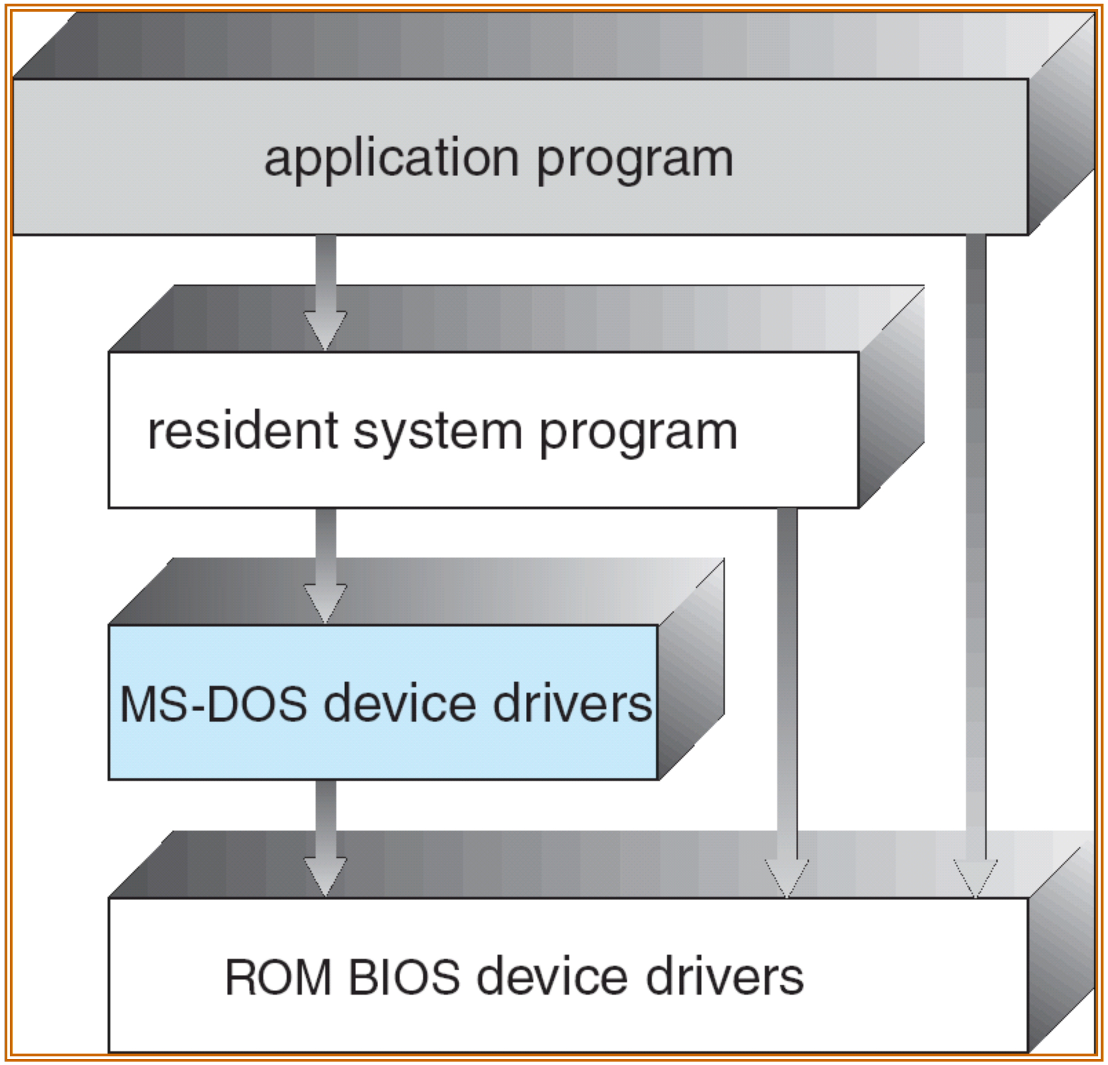
Operating system structures

1. Monolithic systems(Simple Structure)
2. Layered systems
3. Microkernel
4. Client-server model
5. Virtual machines
   - VM/370
   - Virtual machines rediscovered
   - The java virtual machine
6. Exokernels

Simple Structure

- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

MS-DOS Layer Structure



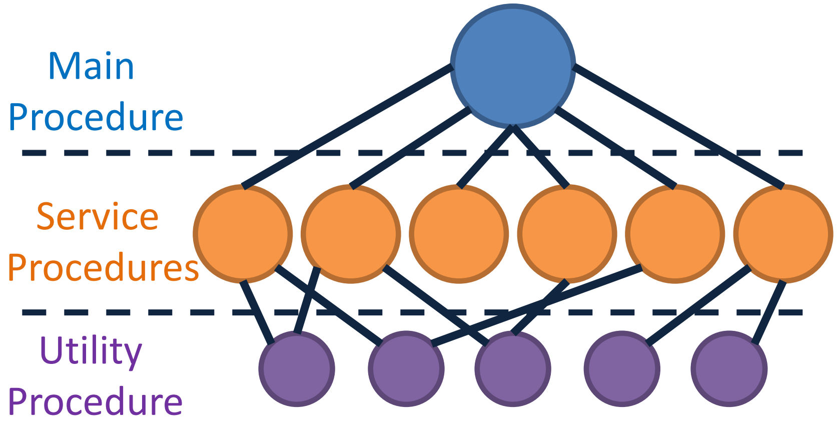
Monolithic systems

- The entire OS **runs as a single program in kernel mode**.
- OS is written as a **collection of procedures, linked together into a single large executable binary program**.
- Each procedure has well defined **interface in terms of parameter and results**, and **each one is free to call any other one**.

✓ **A main program that invoke the requested service procedure.**

✓ **A set of service procedures that carry out the system calls.**

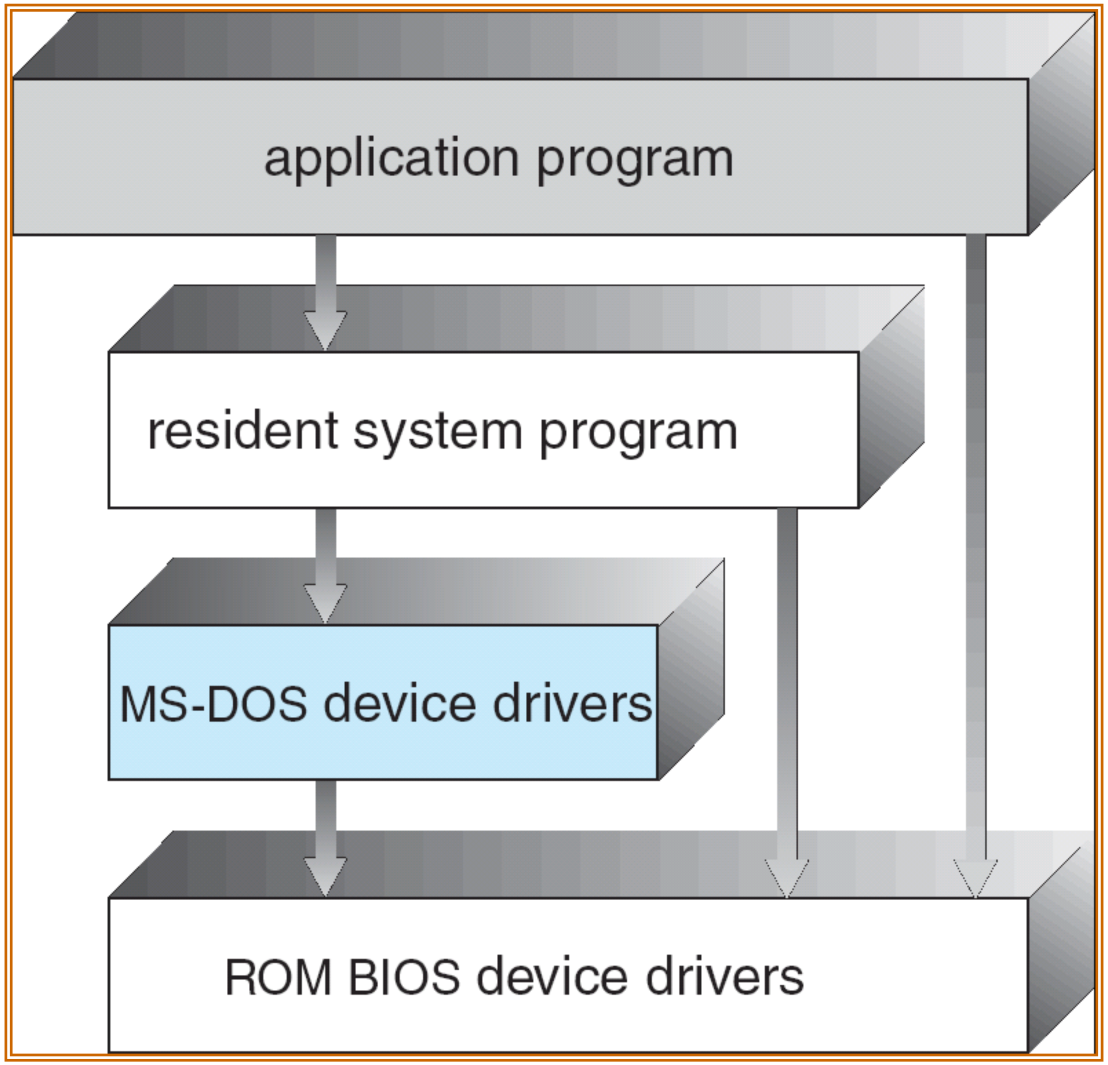
✓ **A set of utility procedures that help the service procedure.**



Layered systems

- In this system, the **OS is organized as a hierarchy of layers**.
- The first system constructed in this way was **THE system**.

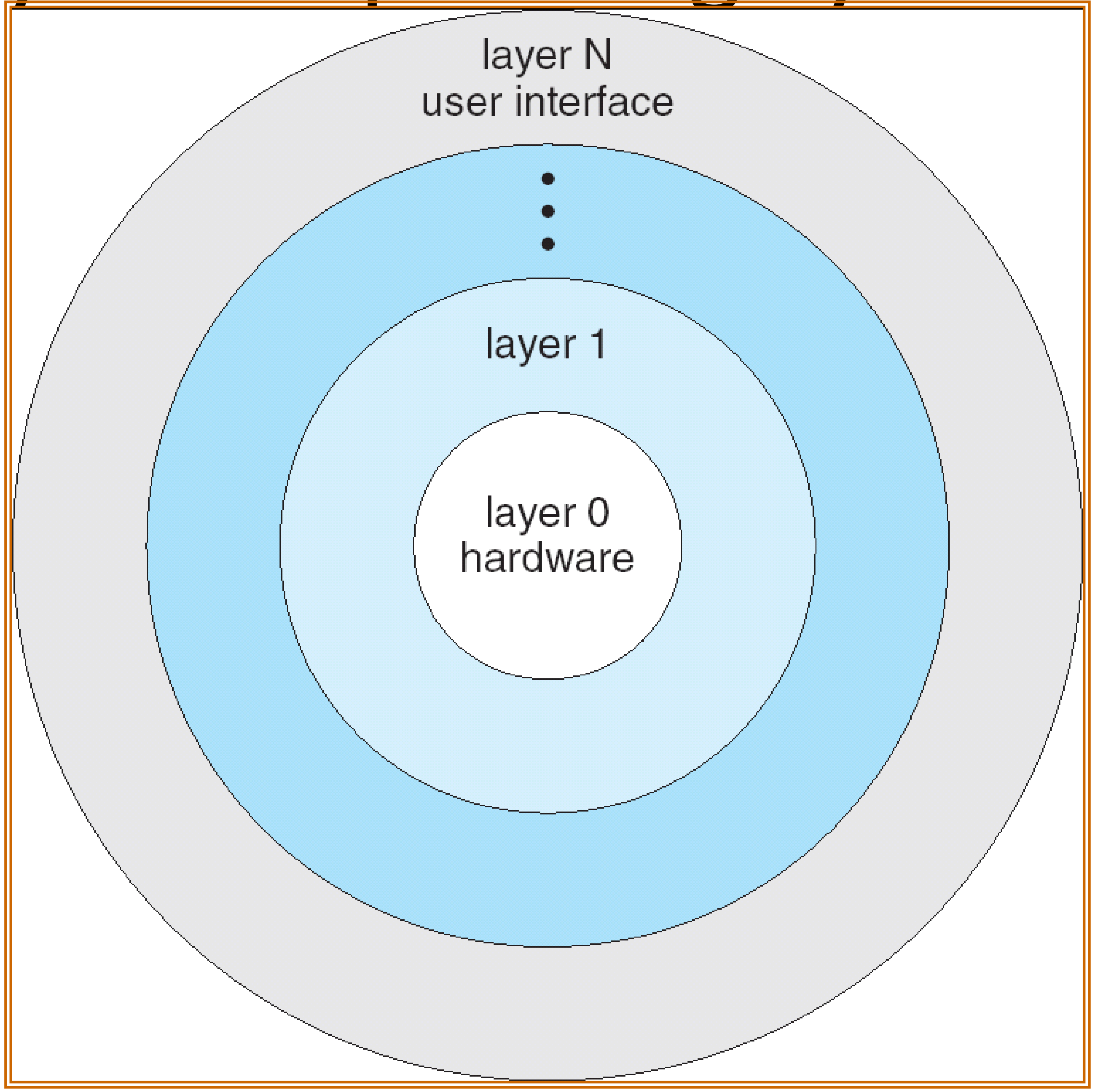
MS-DOS Layer Structure



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

Layered Operating System



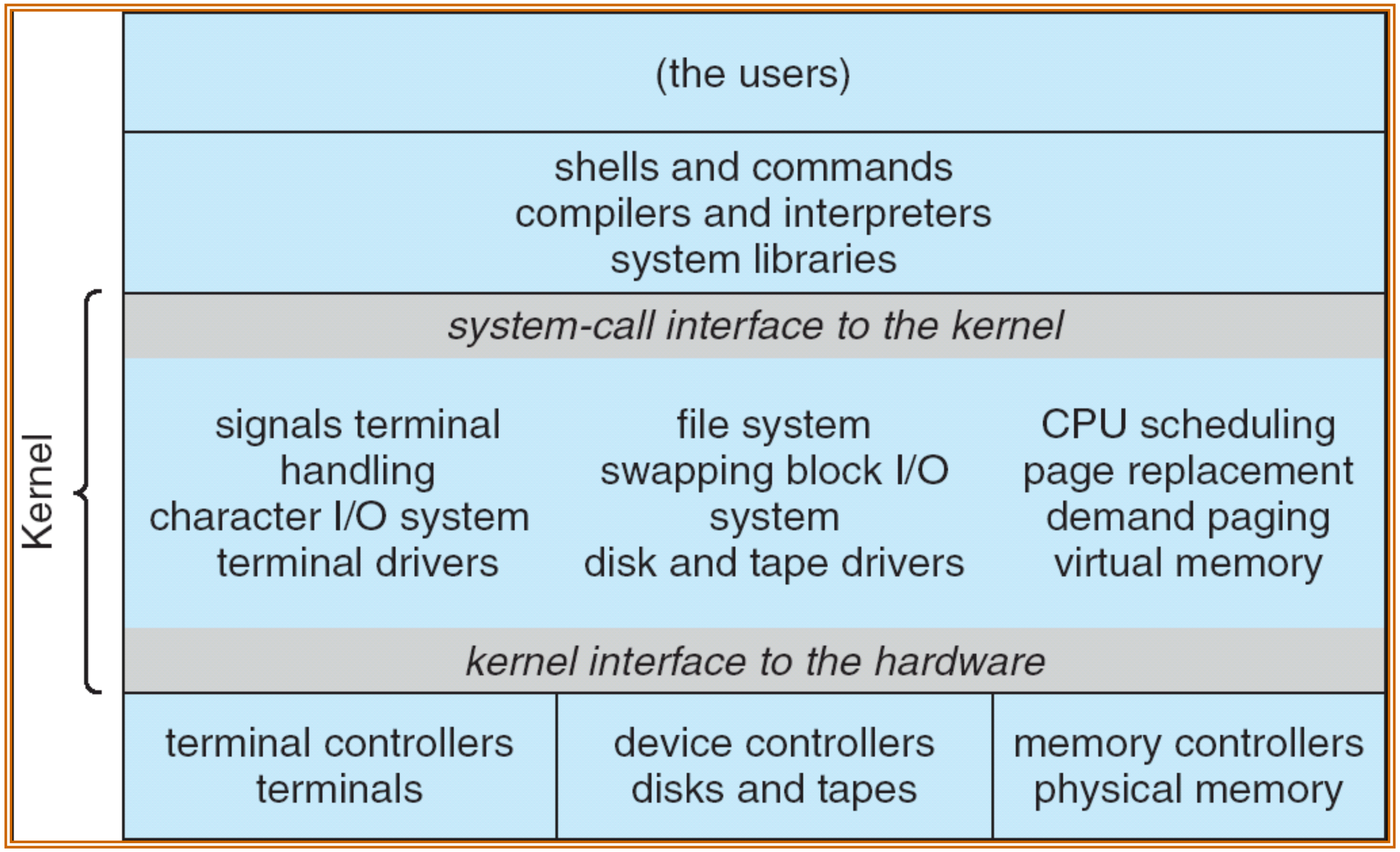
Architecture of THE system

Layer	Function	Description
5	The operator	Operator was located.
4	User Programs	User programs were found.
3	Input / Output Management	Takes care of managing the I/O devices. Buffering the information.
2	Operator-process communication	Handles communication between each process and the operator console (i.e. user).
1	Memory and drum management	Did the memory management. Allocated space for process in main memory and on a 512K word drum used for holding parts of processes for which there was no room in memory.
0	Processor allocation and multi-programming	Provided the basic multiprogramming of the CPU. Dealt with allocation of the processor, switching between processes when interrupts occurred or timers expired.

UNIX

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts
  - Systems programs
  - The kernel
    - Consists of everything below the system-call interface and above the physical hardware
    - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level

UNIX System Structure



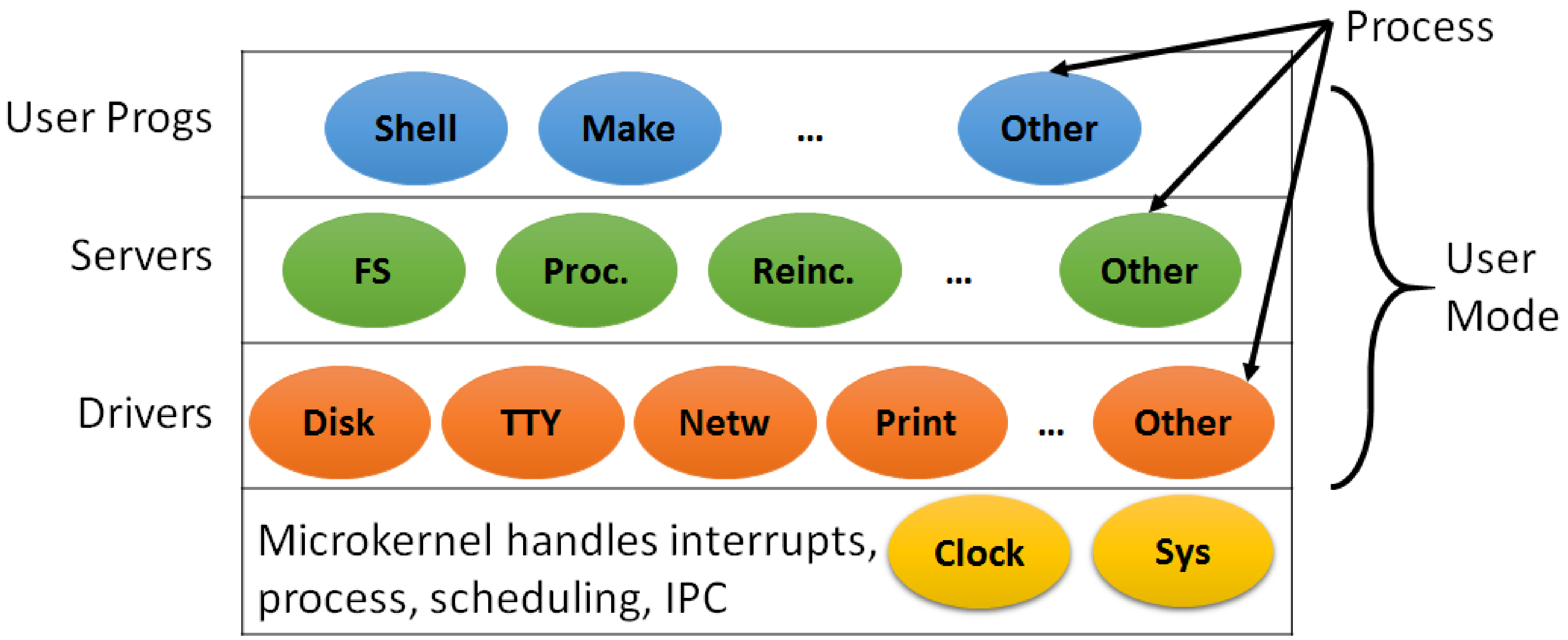
Microkernel

- 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 reliability 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.
- **Examples**: Integrity, K42, QNX, Symbian and MINIX 3.

Microkernel System Structure

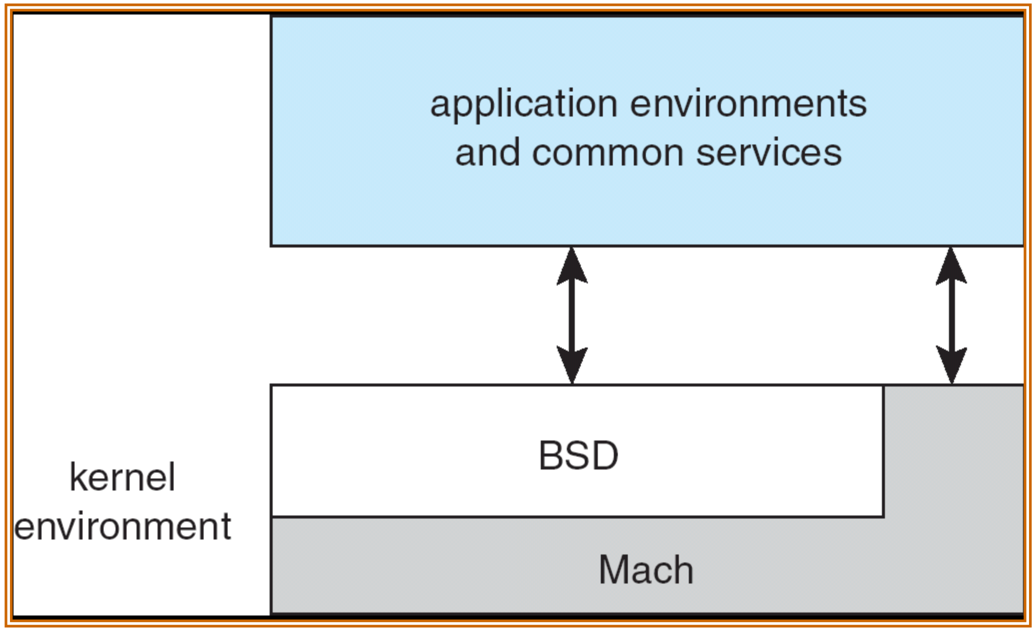
- Moves as much from the kernel into "*user*" space
- Communication takes place between user modules using message passing
- Benefits:
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- Detriments:
  - Performance overhead of user space to kernel space communication

Microkernel (Cont…)



- Kernel contains only
  - Sys (Kernel call handler)
  - Clock (because scheduler interact with it)

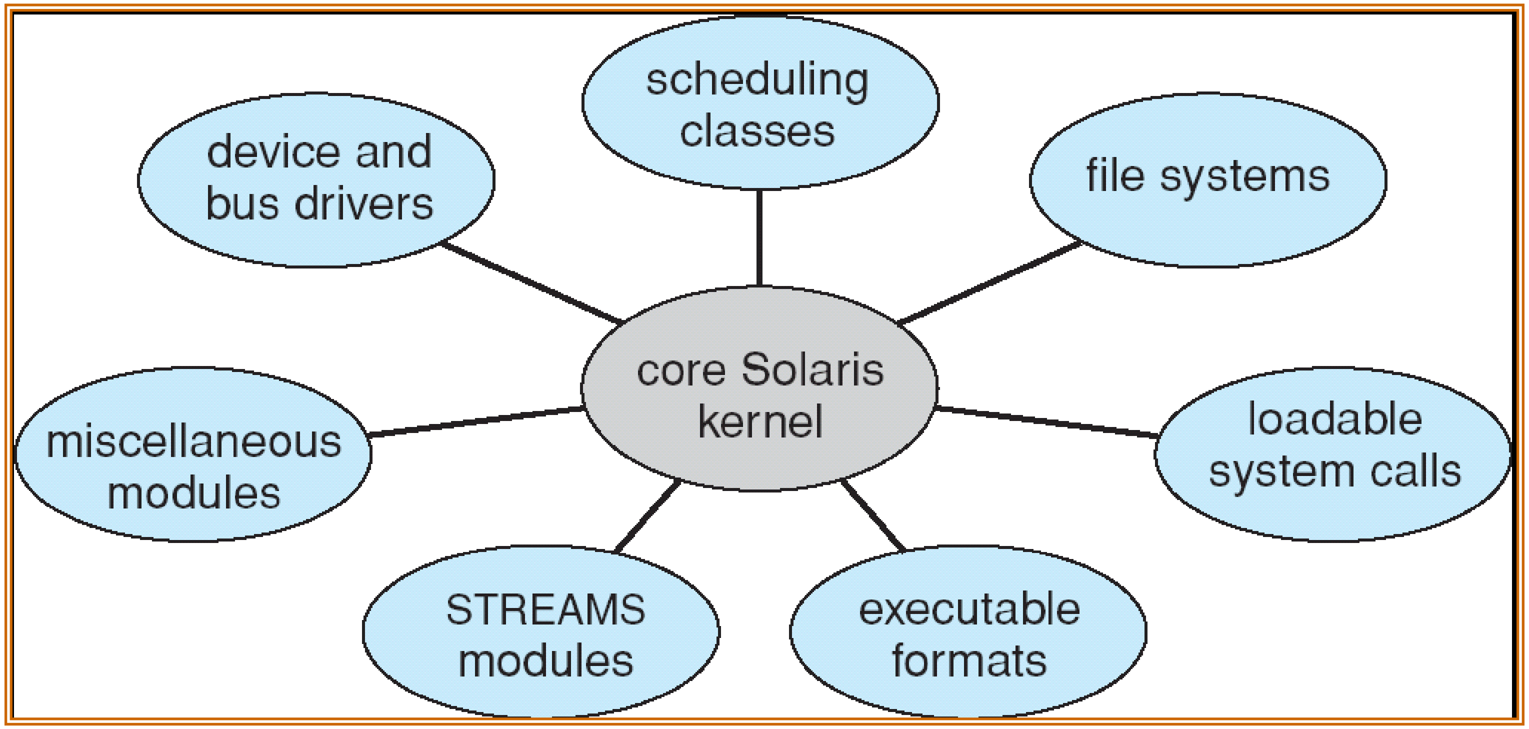
Mac OS X Structure



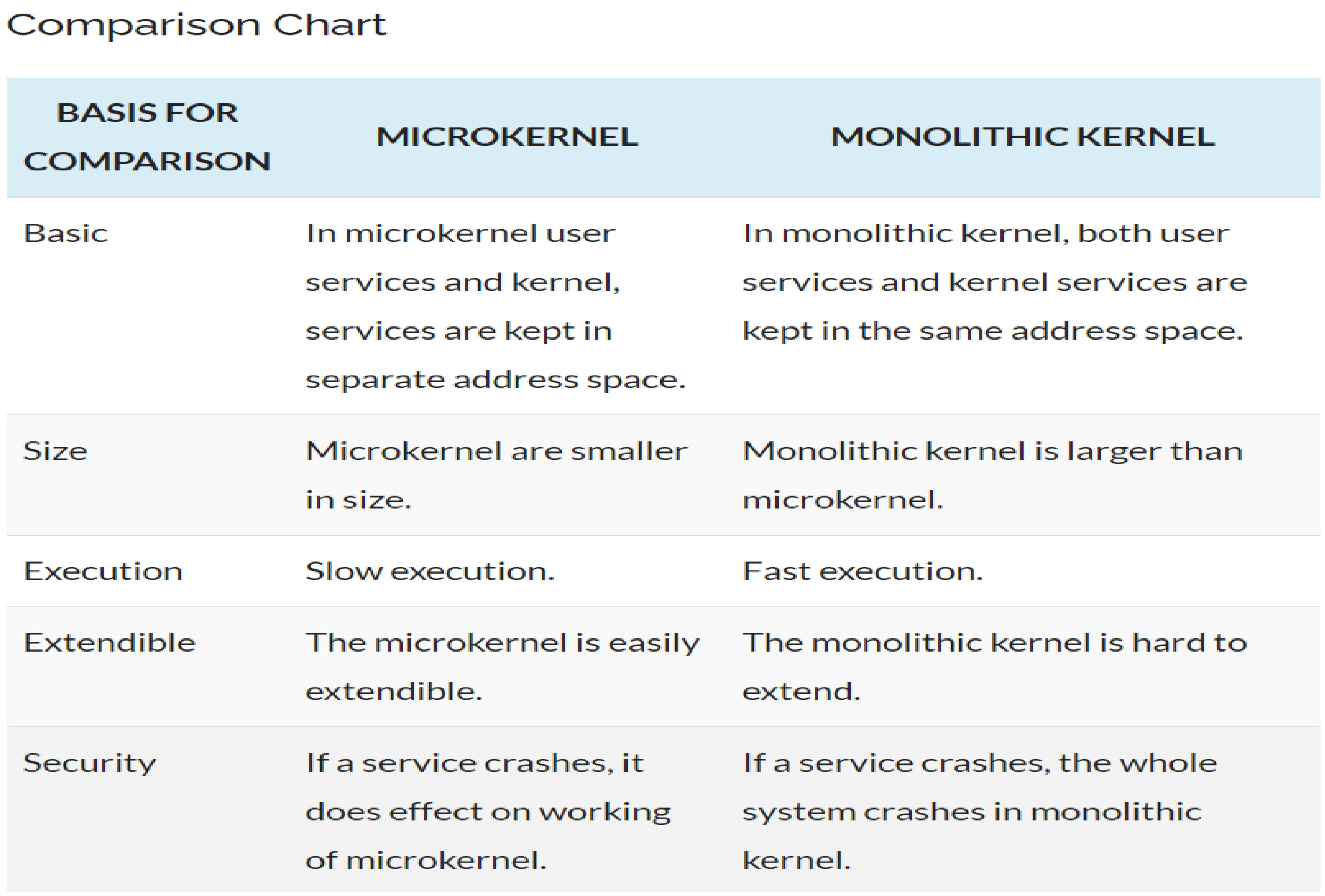
Modules

- Most modern operating systems implement kernel modules
  - Uses object-oriented approach
  - Each core component is separate
  - Each talks to the others over known interfaces
  - Each is loadable as needed within the kernel
- Overall, similar to layers but with more flexible

Solaris Modular Approach



Monolithic V/S Micro Kernel



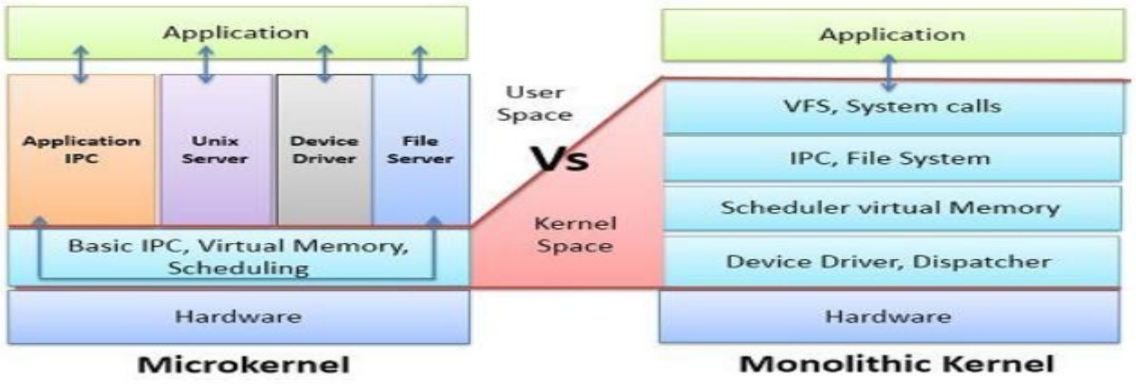
Comparison Chart

BASIS FOR COMPARISON	MICROKERNEL	MONOLITHIC KERNEL
Basic	In microkernel user services and kernel, services are kept in separate address space.	In monolithic kernel, both user services and kernel services are kept in the same address space.
Size	Microkernel are smaller in size.	Monolithic kernel is larger than microkernel.
Execution	Slow execution.	Fast execution.
Extendible	The microkernel is easily extendible.	The monolithic kernel is hard to extend.
Security	If a service crashes, it does effect on working of microkernel.	If a service crashes, the whole system crashes in monolithic kernel.

Monolithic V/S MicroKernel

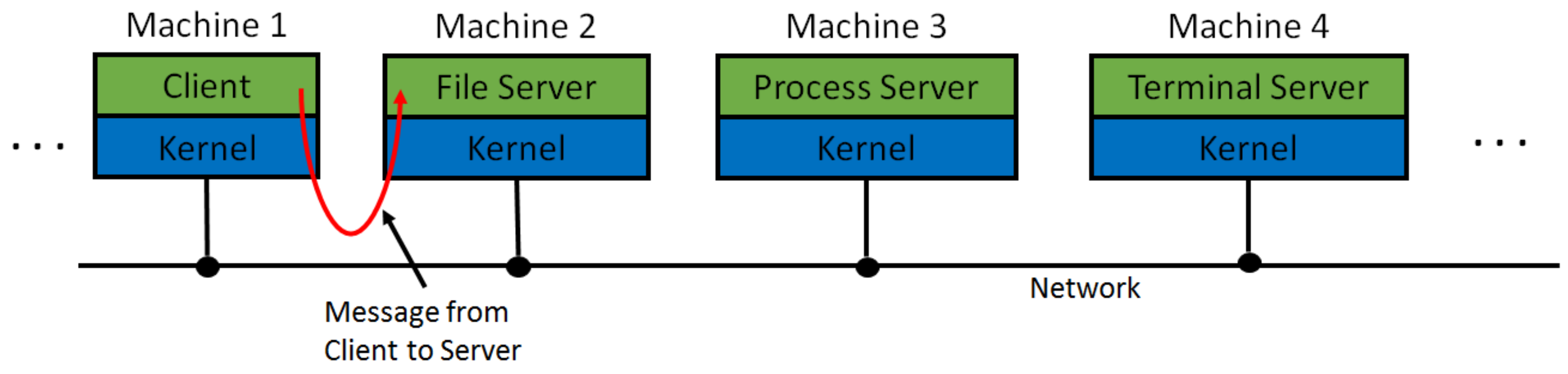
Code	To write a microkernel, more code is required.	To write a monolithic kernel, less code is required.
Example	QNX, Symbian, L4Linux, Singularity, K42, Mac OS X, Integrity, PikeOS, HURD, Minix, and Coyotos.	Linux, BSDs (FreeBSD, OpenBSD, NetBSD), Microsoft Windows (95,98,Me), Solaris, OS-9, AIX, HP-UX, DOS, OpenVMS, XTS-400 etc.

Difference Between Microkernel and Monolithic Kernel



Client-Server model

- Processes are divided into two categories
  - **Servers**: provide services
  - **Clients**: uses services
- Client and server **run on different computers**, **connected by LAN or WAN and communicate via message passing**.
- To obtain a service, a **client construct a message** saying **what it wants and send it to server**.
- The **server then does the work and send back the answer**.



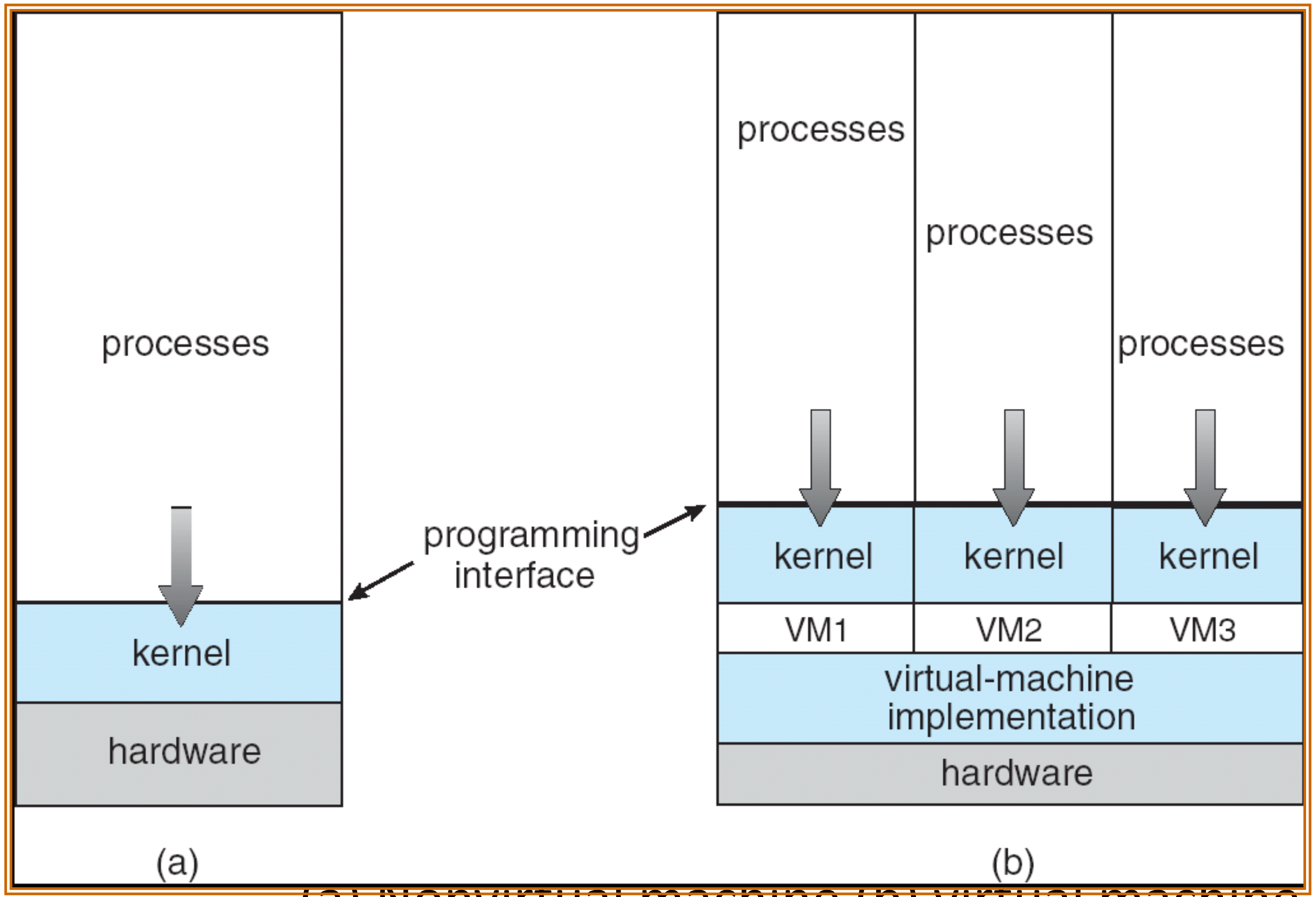
Virtual Machines

- A *virtual machine* takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware
- A virtual machine provides an interface *identical* to the underlying bare hardware
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory

Virtual Machines (Cont.)

- The resources of the physical computer are shared to create the virtual machines
  - CPU scheduling can create the appearance that users have their own processor
  - Spooling and a file system can provide virtual card readers and virtual line printers
  - A normal user time-sharing terminal serves as the virtual machine operator’s console

Virtual Machines (Cont.)

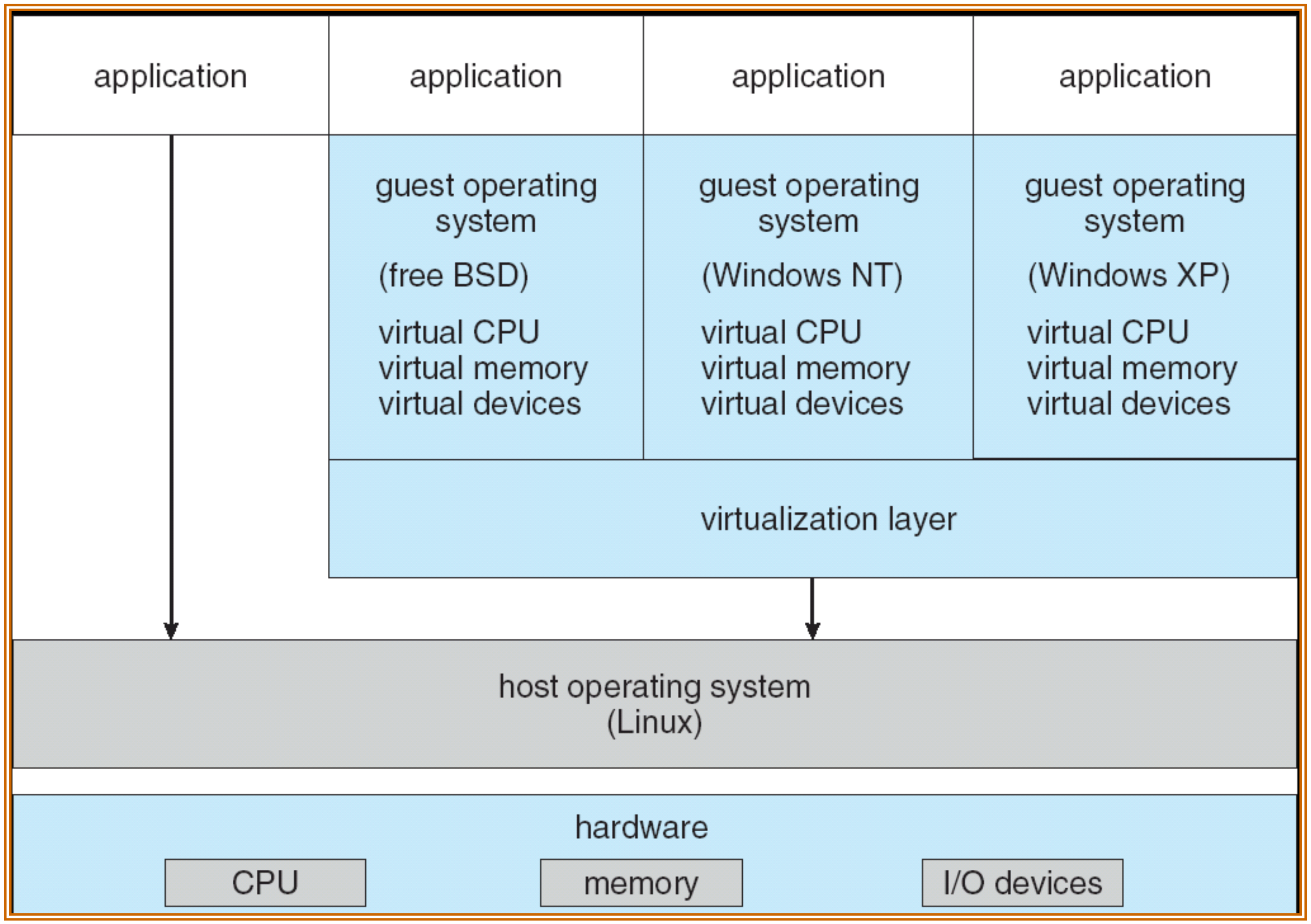


(a) Nonvirtual machine (b) virtual machine

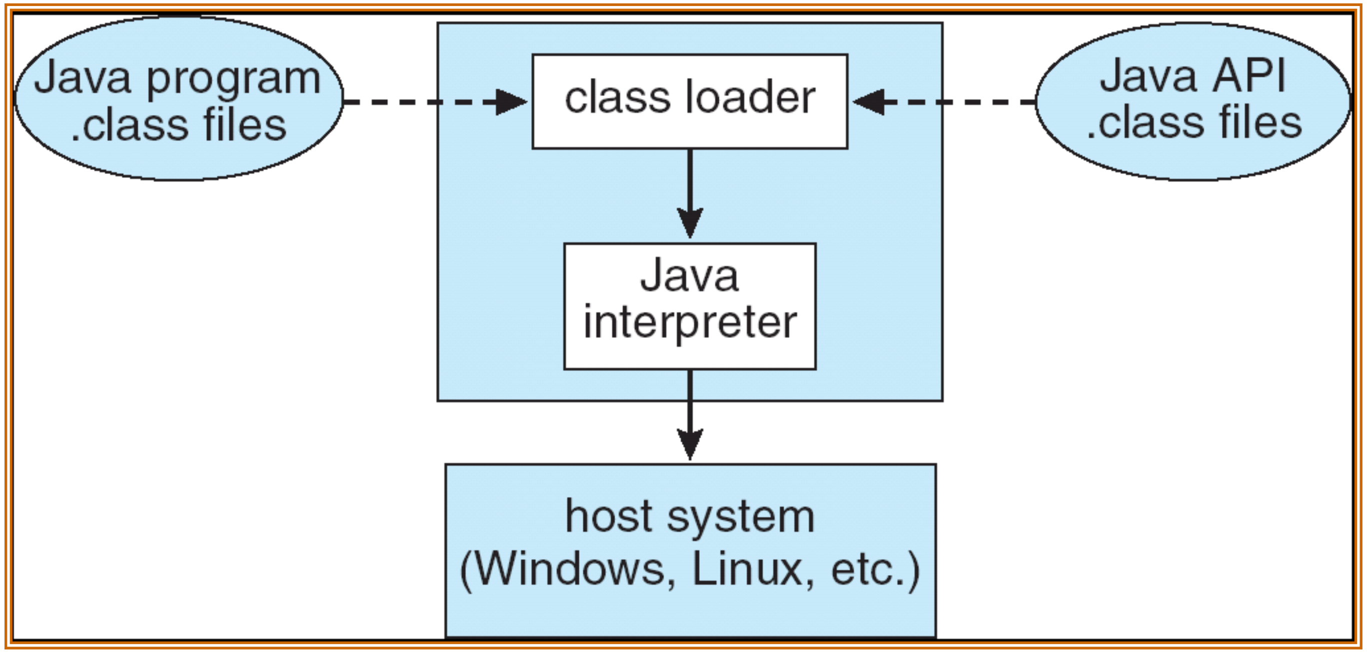
Virtual Machines (Cont.)

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating-systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an *exact* duplicate to the underlying machine

VMware Architecture



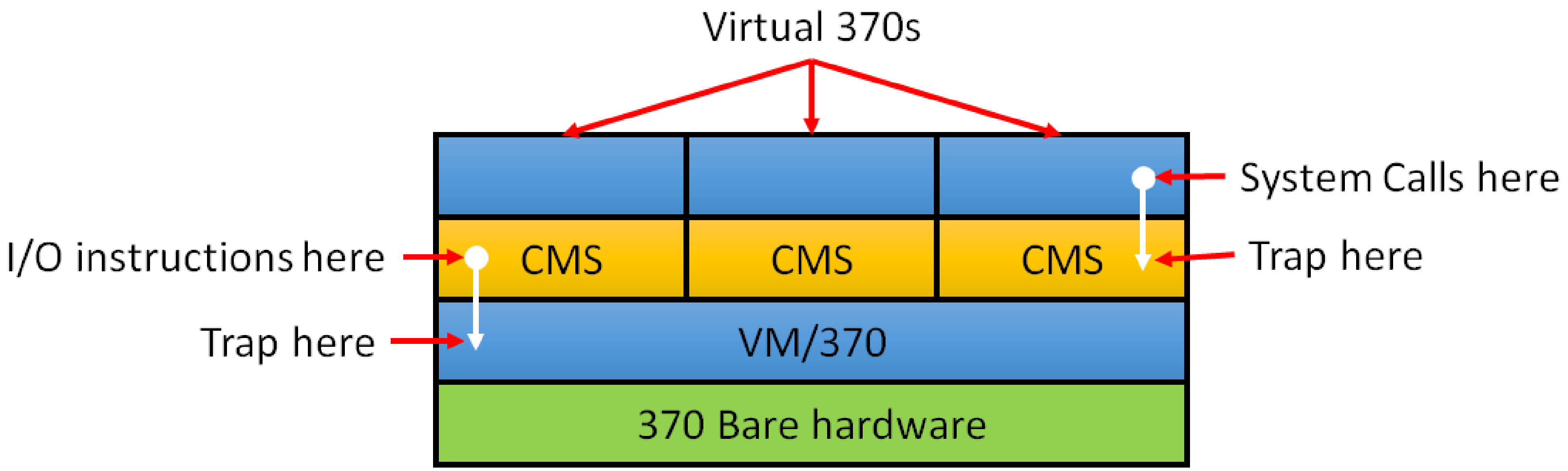
The Java Virtual Machine



Virtual machines

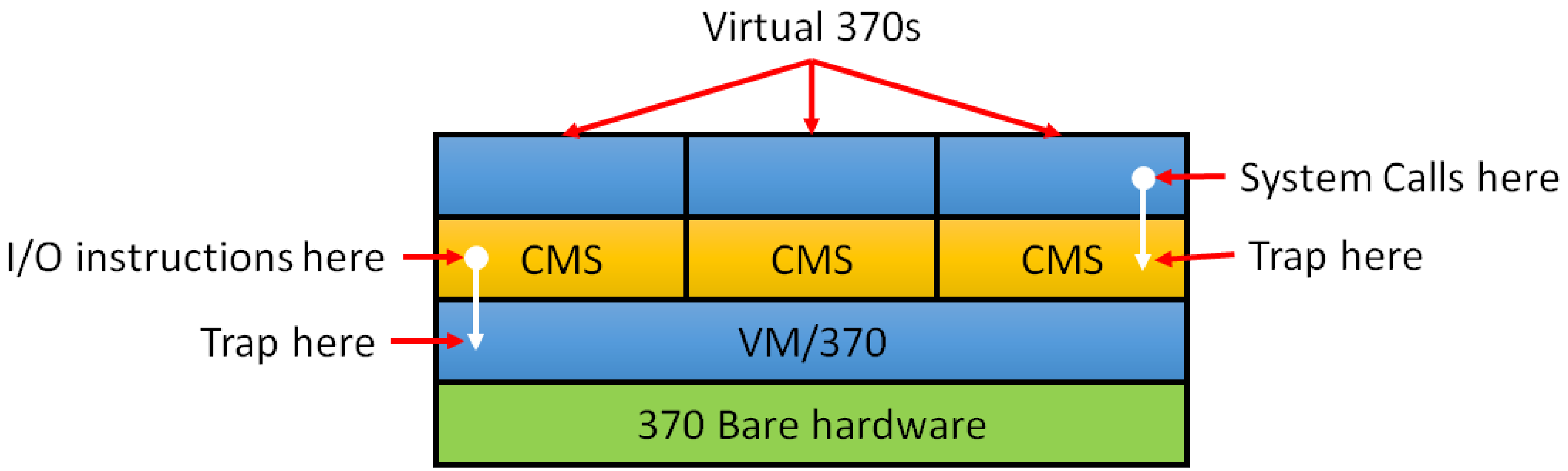
- The **initial releases of OS/360 were strictly batch systems**.
- But many users wanted to be able to work interactively at a terminal, so OS designers decided to write timesharing systems for it.
- Types of Virtual machines are:
  - VM/370
  - Virtual Machines Rediscovered
  - The Java Virtual Machine

VM/370



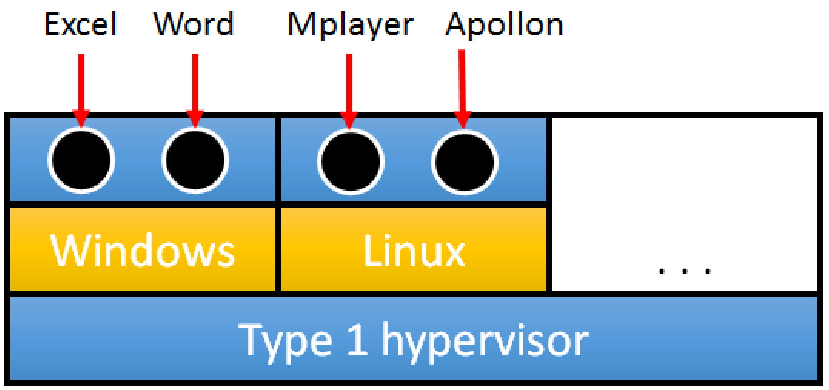
- Virtual machine monitor **run on the bare hardware** and **does the multiprogramming**.
- **Each virtual machine is identical to the true hardware**; each one can run any OS (may be different) that will run directly on the bare hardware.
- On VM/370, some run OS/360 while the others run single user interactive system called CMS (Conversational Monitor System) for interactive time sharing users.

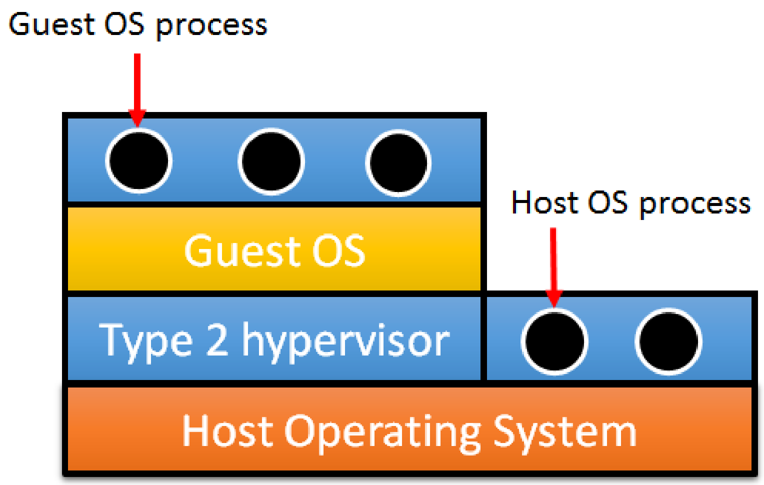
VM/370 (Cont…)



- When CMS program executed a system call, a call was trapped to the operating system in its own virtual machine, not on VM/370.
- CMS then issued the normal hardware I/O instruction for reading its virtual disk or whatever was needed to carry out the call.
- These I/O instructions were trapped by VM/370 which then performs them.

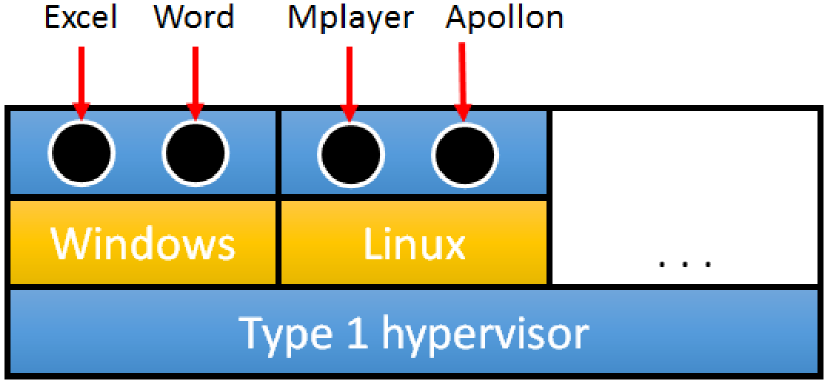
Virtual Machines Rediscovered

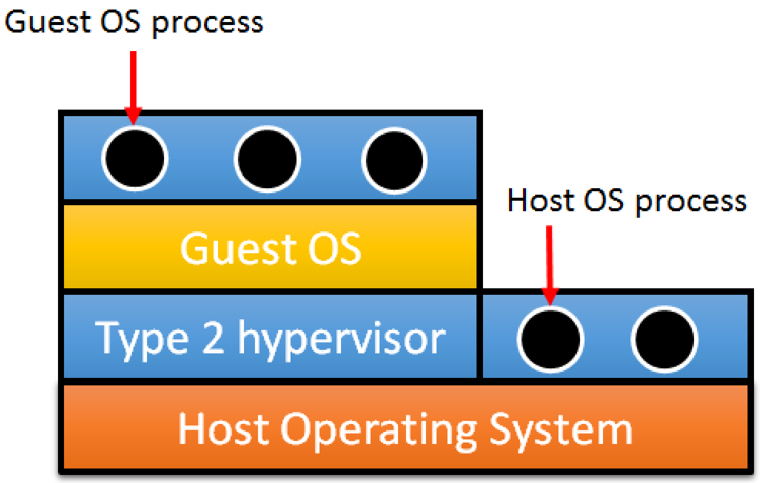




- Companies can run their mail servers, web servers, FTP servers and other servers on the same machine without having a crash of one server bring down the rest.
- Web hosting company offers virtual machines for rent, where a single physical machine can run many virtual machines; each one appears to be a complete machine.
- Customers who rent a virtual machine can run any OS or software.

Virtual Machines Rediscovered (Cont…)

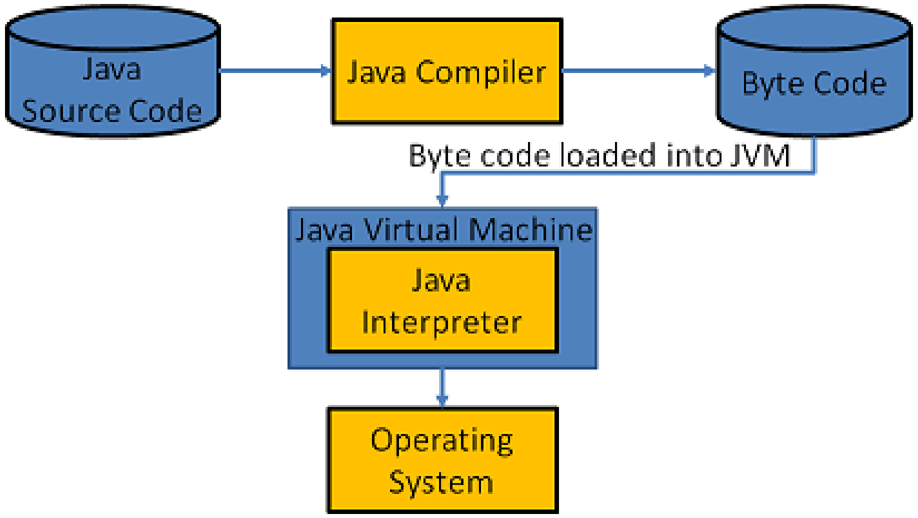




- Another use of virtualization is for end users who want to be able to **run two or more operating systems at the same time**, say Windows and Linux.
- Type 1 hypervisors runs on the bare hardware.
- Type 2 hypervisors run as application programs on top of Windows, Linux, or some other operating system, known as the host operating system.

Java Virtual Machine

- When Sun Microsystems invented the Java programming language, it also invented a virtual machine called the JVM (Java Virtual Machine).
- The Java compiler produces code for JVM, which then typically is executed by a software JVM interpreter.
- The **advantage** is that the **JVM code can be shipped over the internet to any computer** that has a JVM interpreter and run there.



Exokernels

- **Rather than cloning (copying)** the actual machine, another strategy is **partitioning it** (giving each user a subset of the resource).
- For example one virtual machine might get disk blocks 0 to 1023, the next one might get block 1024 to 2047, and so on.
- **Program running at the bottom layer** (kernel mode) is called the **exokernel**.
- Its **job is to allocate resources** to virtual machines and then **continuously check** to make sure no machine is trying to use somebody else's resources.
- The **advantage** of the **exokernel** scheme is that it **saves a layer of mapping**.

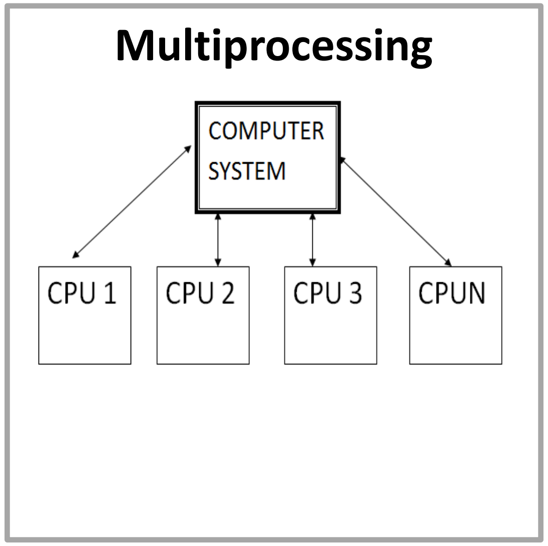
Operating System Generation

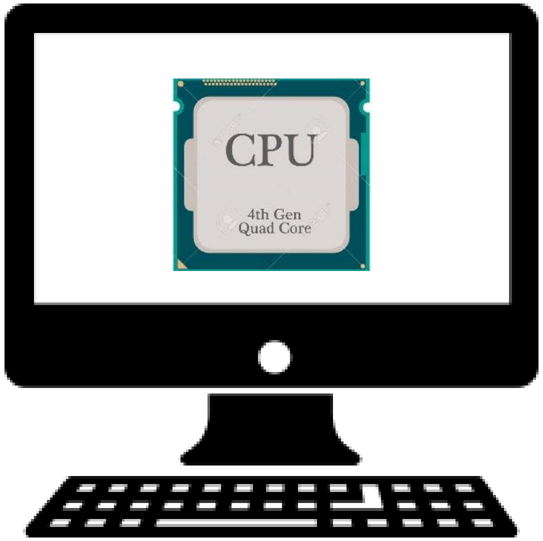
- Operating systems are designed to run on any of a class of machines; the system must be configured for each specific computer site
- SYSGEN program obtains information concerning the specific configuration of the hardware system
- *Booting* – starting a computer by loading the kernel
- *Bootstrap program* – code stored in ROM that is able to locate the kernel, load it into memory, and start its execution

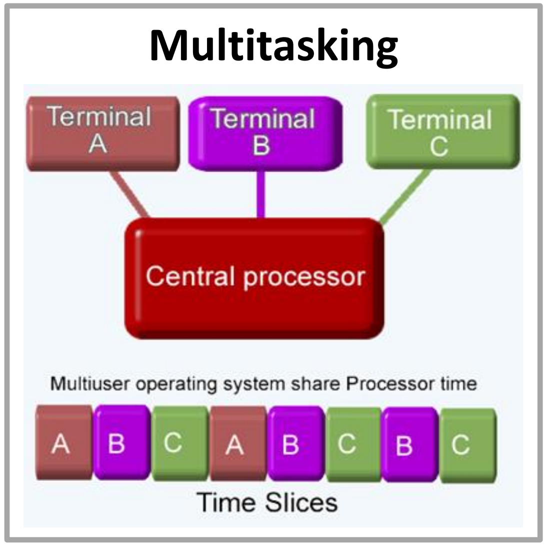
System Boot

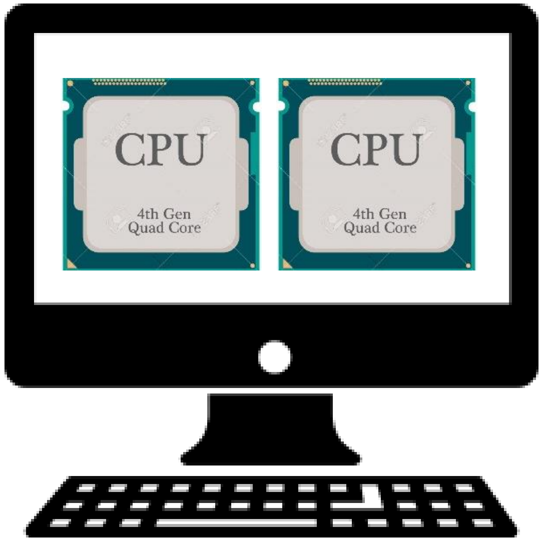
- Operating system must be made available to hardware so hardware can start it
  - Small piece of code – **bootstrap loader**, locates the kernel, loads it into memory, and starts it
  - Sometimes two-step process where **boot block** at fixed location loads bootstrap loader
  - When power initialized on system, execution starts at a fixed memory location
    - Firmware used to hold initial boot code

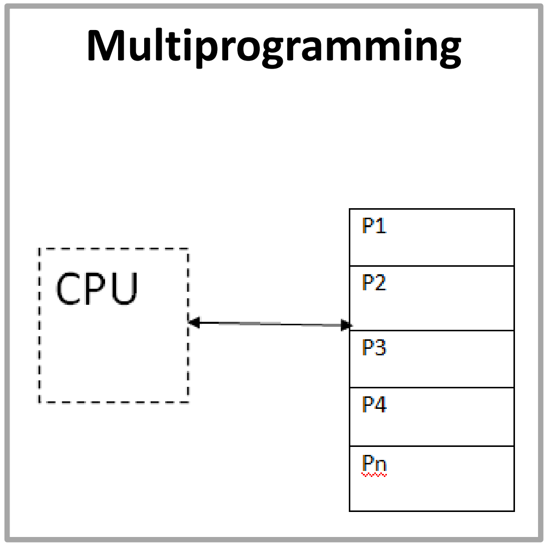
Multiprogramming v/s Multiprocessing v/s Multitasking

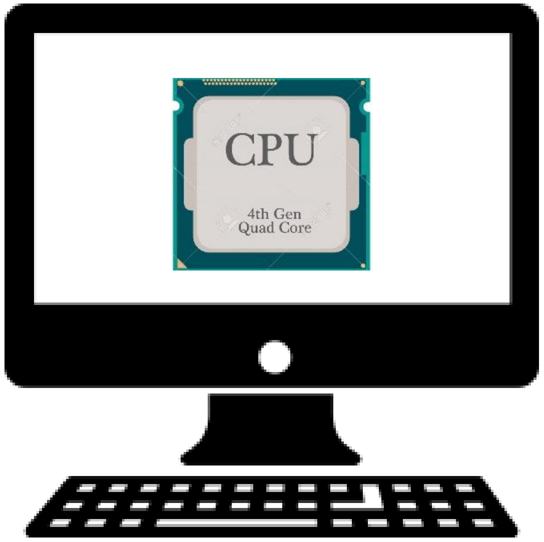










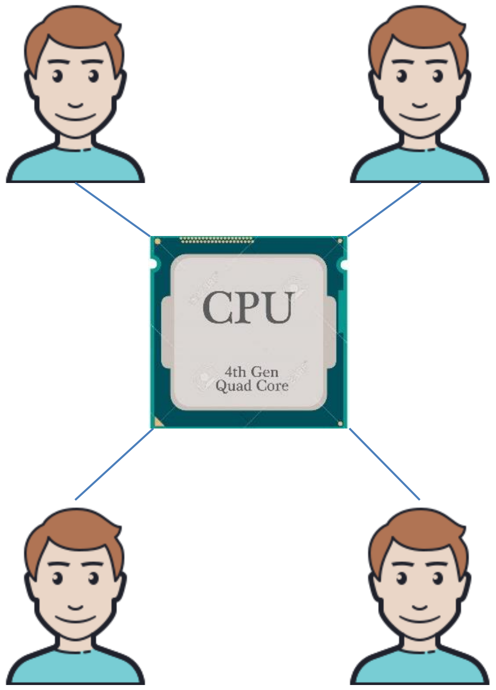


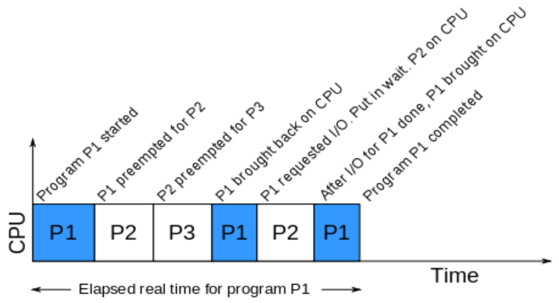
Multiprogramming v/s Multiprocessing v/s Multitasking

Multiprogramming	Multiprocessing	Multitasking
The concurrent residency of more than one program in the main memory is called as multiprogramming.	The availability of more than one processor per system, which can execute several set of instructions in parallel is called as multiprocessing.	The execution of more than one task simultaneously is called as multitasking.
Number of processor: one	Number of processor: more than one	Number of processor: one
One process is executed at a time.	More than one process can be executed at a time.	One by one job is being executed at a time.

Time Sharing Operating System

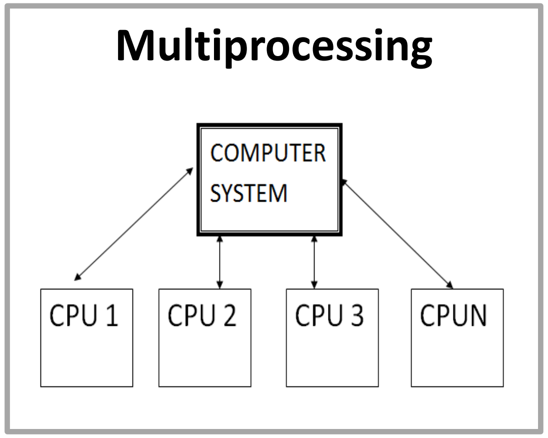
- A time sharing operating system **allows many users to share the computer resources simultaneously**.
- In other words, time sharing refers to the **allocation of computer resources in time slots** to several programs simultaneously.
- For example a mainframe computer that has many users logged on to it.
- Each user uses the resources of the mainframe i.e. memory, CPU etc.

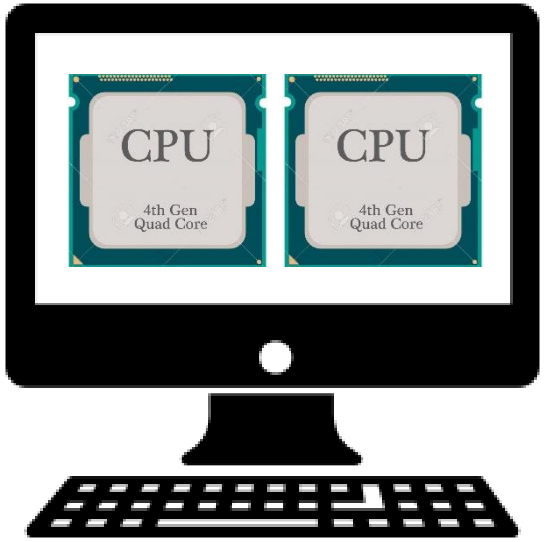




Parallel Processing Operating System

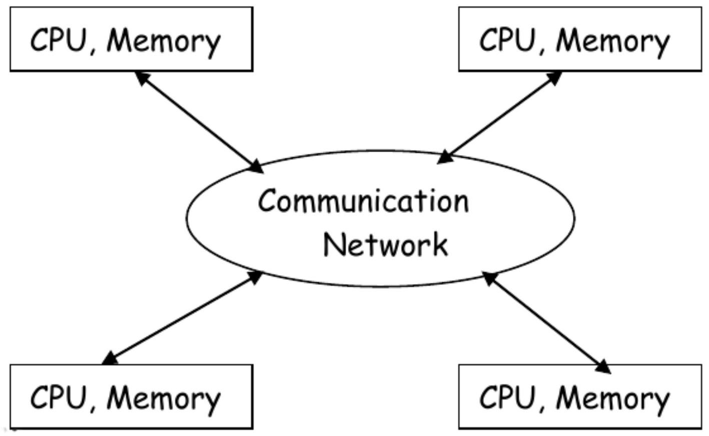
- Parallel Processing Operating Systems are designed to **speed up the execution of programs by dividing the program into multiple fragments and processing these fragments simultaneously**.
- Such systems are multiprocessor systems.
- Parallel systems deal with the **simultaneous use of multiple computer resources that can include a single computer with multiple processors**.





Distributed Operating System

- Distributed Operating System is a model where **distributed applications are running on multiple computers linked by communications**.
- A distributed operating system is an extension of the network operating system that supports higher levels of communication and integration of the machines on the network.



Questions asked in GTU

1. What is Kernel? Differentiate between Monolithic Kernel and Micro Kernel.
2. Explain different service/functions provided by operating system.
3. Discuss role of OS as a resource manager.
4. Explain the features of Time sharing system.
5. What is operating system? Give the view of OS as a resource manager.
6. What is system call? Explain steps for system call execution.
7. Write different types of system call.
8. List out types of operating system and explain batch OS and time sharing OS in brief.

Thank you

Types of OS

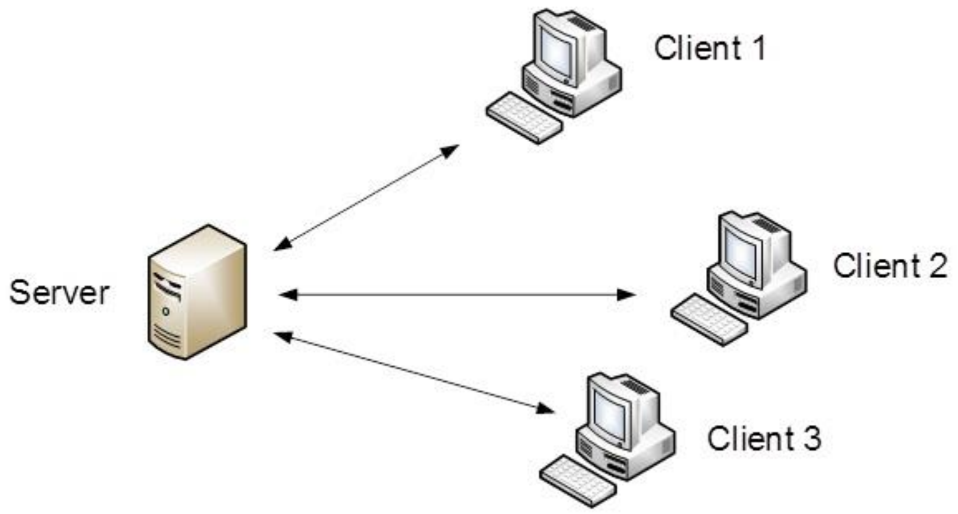
1. Mainframe operating systems

Types of OS

1. Mainframe operating systems
   - OS **found in room sized computers** which are still found in major corporate data centres.
   - They offer three kinds of services:
     1) **Batch OS** – **processes routine jobs** without any interactive user presents i.e. claim processing in insurance
     2) **Transaction processing** – **handles large numbers of small processes** i.e. cheque processing at banks
     3) **Timesharing** – **allows multiple remote users to run their jobs at once** i.e. querying a database, airline booking system
   - **Examples**: OS/390, OS/360.

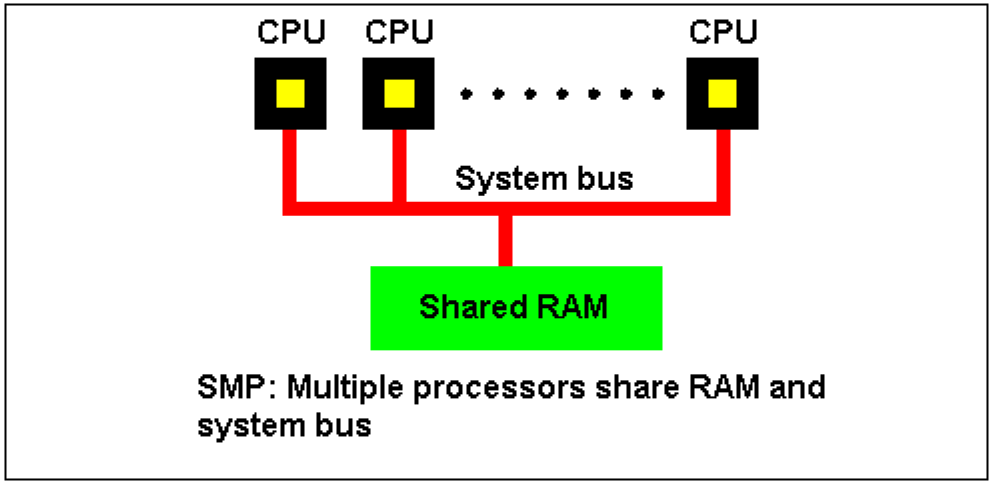
Types of OS (Cont…)

2. Server operating systems
   - This OS **runs on servers** which are very large PC, workstations or even mainframes.
   - They **serve multiple users at once** over a network and **allow the users to share hardware & software resources**.
   - It provides **print services, file service or web service**.
   - It **handles the incoming requests** from clients.
   - **Examples**: Solaris, FreeBSD, and Linux and Windows Server 200x.



Types of OS (Cont…)

3. Multiprocessor operating systems
    - A computer system **consist two or more CPUs** is called multiprocessor.
    - It is also called **parallel computers, multicomputer or multiprocessor**.
    - They **need special OS** or some variations on server OS with special features for **communication, connectivity and consistency**.
    - **Examples**: Windows and Linux.



Types of OS (Cont…)

4. Personal computer operating systems
   - The operating systems **installed on our personal computer and laptops** are personal OS.
   - Job of this OS is to **provide good support to single user**.
   - This OS is widely used for **word processing, spreadsheet and internet access**.
   - **Examples**: Linux, Windows vista and Macintosh.

Types of OS (Cont…)

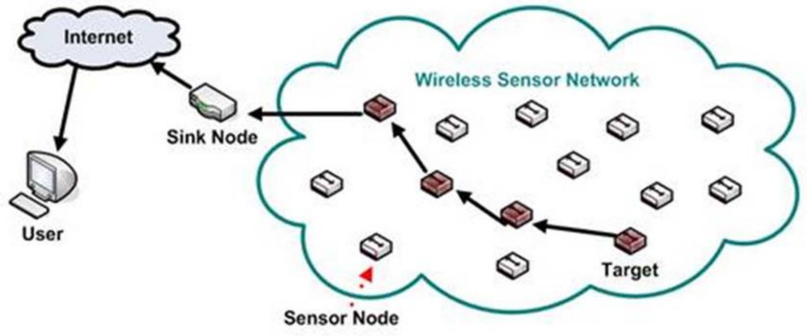
5. Handhelds computer operating systems
   - A handheld computer or **PDA (Personal Digital Assistant)** is small computer that **fit in a Pocket and perform small number of functions** such as **electronic address book, memo pad**.
   - The OS runs on these devices are handheld OS.
   - These OS also provides **ability to handle telephony, digital photography and other functions**.
   - **Examples**: Symbian OS, Palm OS.

Types of OS (Cont…)

6. Embedded operating systems
   - This OS is **installed in ATMs, printers, calculators and washing machine**.
   - It runs on the **computer that control devices**.
   - It **neither allow to download new software nor accept user installed software**. So there is **no need for protection**
   - **Examples**: QNX, VxWorks.

Types of OS (Cont...)

7. Sensor node operating systems
   - Network of tiny sensor nodes are being developed for numerous purpose.
   - Each nodes are **tiny computers with a CPU, RAM, ROM and one or more environmental sensors**.
   - The OS installed in these nodes are sensor node OS.
   - They **communicate with each other** and **with base station using wireless communication**.
   - These sensor network are used to **protect area of building, detect fires in forest, measure temperature**.
   - **Examples**: TinyOS.



Types of OS (Cont…)

8. Real time operating systems
   - These systems having **time as a key parameter**.
   - Real time OS has **well defined fixed time constraints**.
   - Processing **must be done within defined time constraints otherwise system fails**.
   - Two types of real time OS:
     1. **Hard real time** – **missing an occasional deadline can cause any permanent damage**. Many of these are found in **industrial process control, car engine control system**.
     2. **Soft real time** – **missing an occasional deadline does not cause any permanent damage**. Used in **digital audio, multimedia system**.
   - **Examples**: e-Cos.

Types of OS (Cont…)

9. Smart card operating systems
    - Smallest OS run on **smart cards** which are **credit card sized devices containing CPU chip**.
    - These OS are installed on **electronic payments cards** such as **debit card, credit card** etc.
    - They have **limited processing power**.
    - Some smart cards are Java oriented. ROM on smart card holds an interpreter for the JVM – small program.

Types of OS (Revision)

1. Mainframe operating systems
2. Server operating systems
3. Multiprocessor operating systems
4. Personal computer operating systems
5. Handhelds computer operating systems
6. Embedded operating systems
7. Sensor node operating systems
8. Real time operating systems
9. Smart card operating systems