CSE321: Operating Systems

Introduction

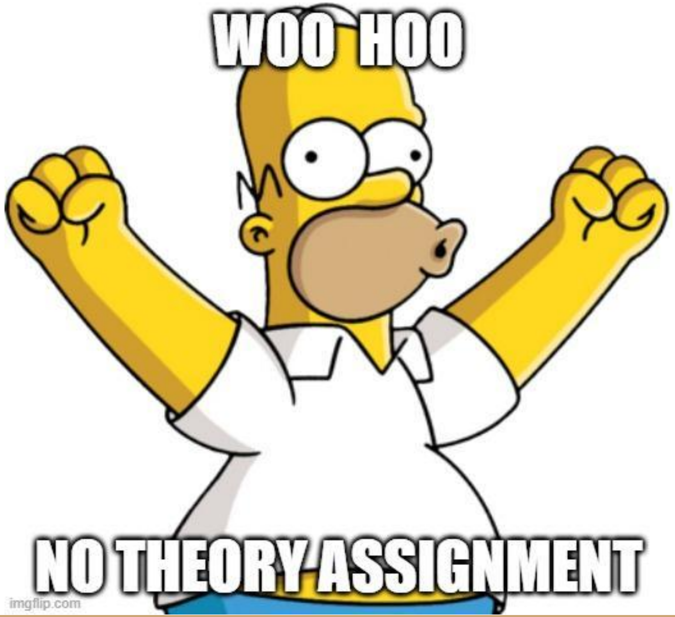
FBA

Course Outcome

- **To understand** the fundamental concepts of computer system organization and the structure of operating systems.
- **To explore** various aspects of process management in operating system
- **To know** how different CPU scheduling algorithm works and their respective importance
- **To develop practical knowledge on** the concept of threads
- **To inspect** process synchronization mechanisms and deadlocks
- **To be able to analyze** the management of main and virtual memory

Marks Distribution

- Theory – 80%
  - Class participation – 5%
  - Quiz – 15%
  - Mid – 25%
  - Final – 35%
- Lab – 20%



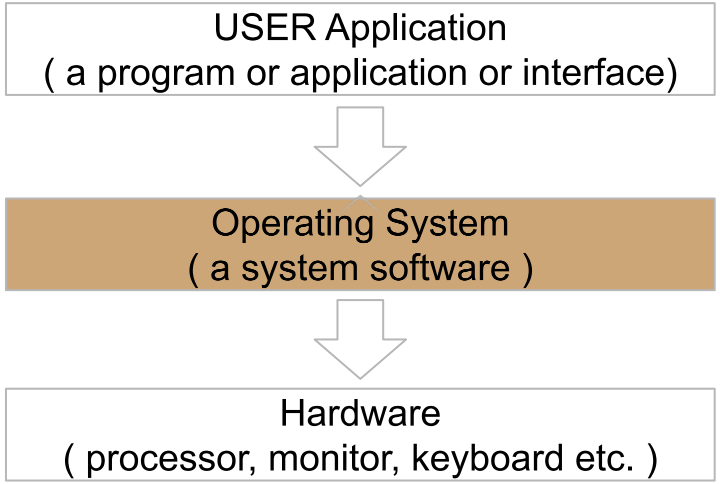
Operating Systems

"Actual" Introduction

FBA

What is an Operating System?

A program that acts as an intermediary
between a user of a computer and the computer hardware.


USER Application
(a program or application or interface)
Operating System
(a system software)
Hardware
(processor, monitor, keyboard etc.)

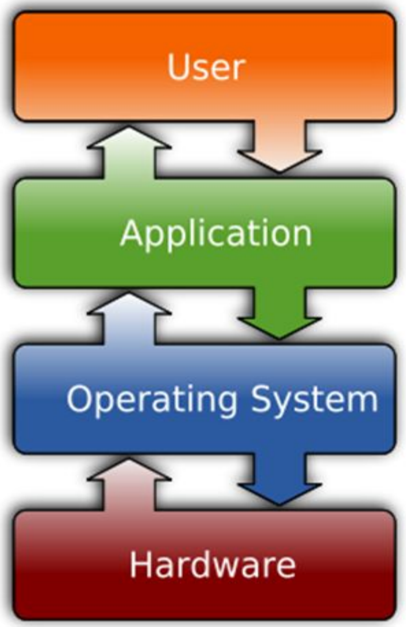
System Software Vs Application Software

System Software:

- System Software refers to the operating system and all utility programs that manage computer resources at a low level.
- Systems software includes compilers, loaders, linkers, and debuggers.

Application Software:

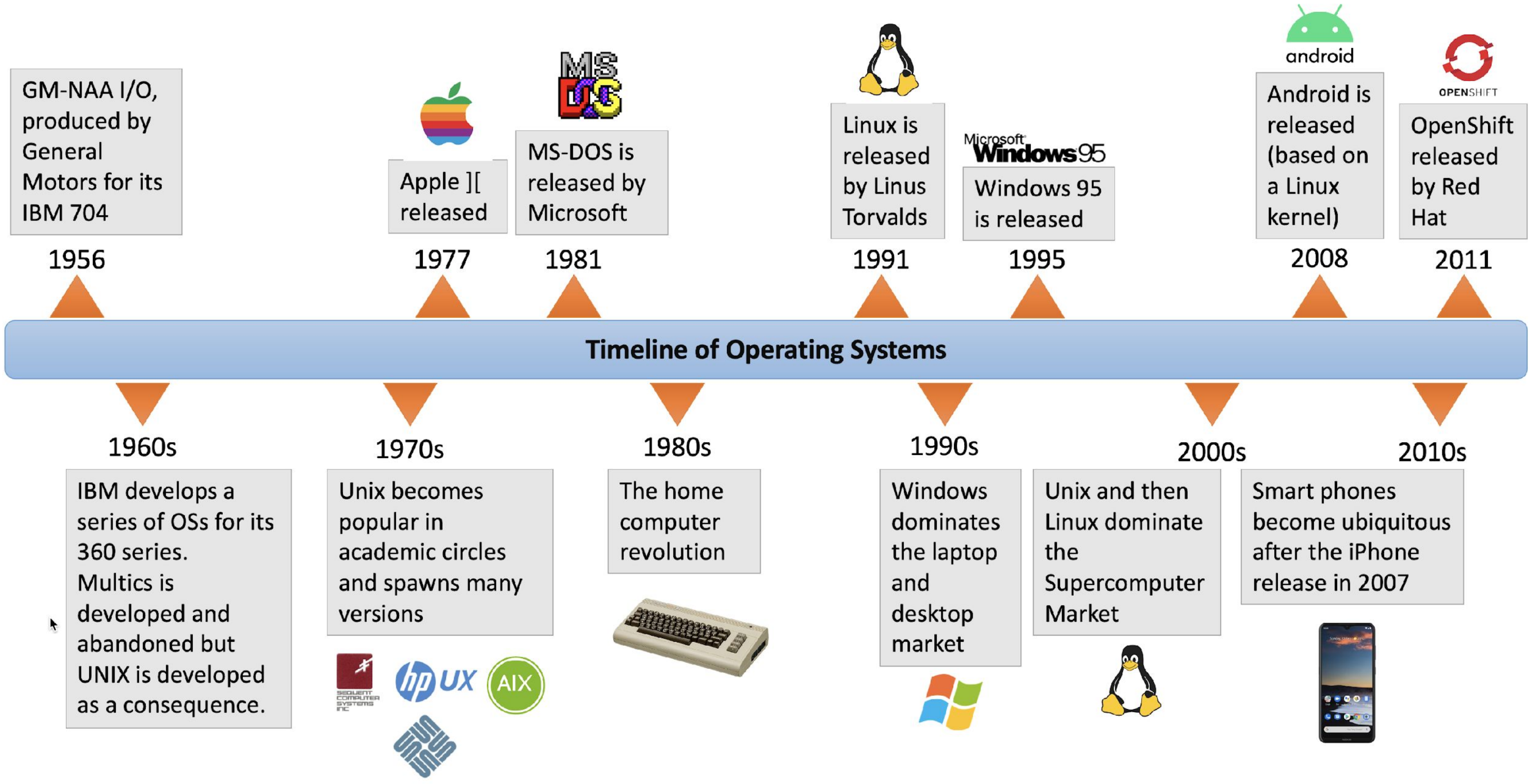
- Applications software comprises programs designed for an end user, such as word processors, database systems, and spreadsheet programs.



Major Goals of OS

- Execute user programs.
- Make the computer system convenient to use.
- Use the computer hardware in an efficient manner
- Manages and allocate all resources
- Controls the execution of user programs and operations of I/O devices

Timeline of OS



Timeline of Operating Systems

1956

GM-NAA I/O, produced by General Motors for its IBM 704

1977

Apple][released

1981

MS-DOS is released by Microsoft

1991

Linux is released by Linus Torvalds

1995

Windows 95 is released

2008

Android is released (based on a Linux kernel)

2011

OpenShift released by Red Hat

1960s

IBM develops a series of OSs for its 360 series. Multics is developed and abandoned but UNIX is developed as a consequence.

1970s

Unix becomes popular in academic circles and spawns many versions

1980s

The home computer revolution

1990s

Windows dominates the laptop and desktop market

2000s

Unix and then Linux dominate the Supercomputer Market

2010s

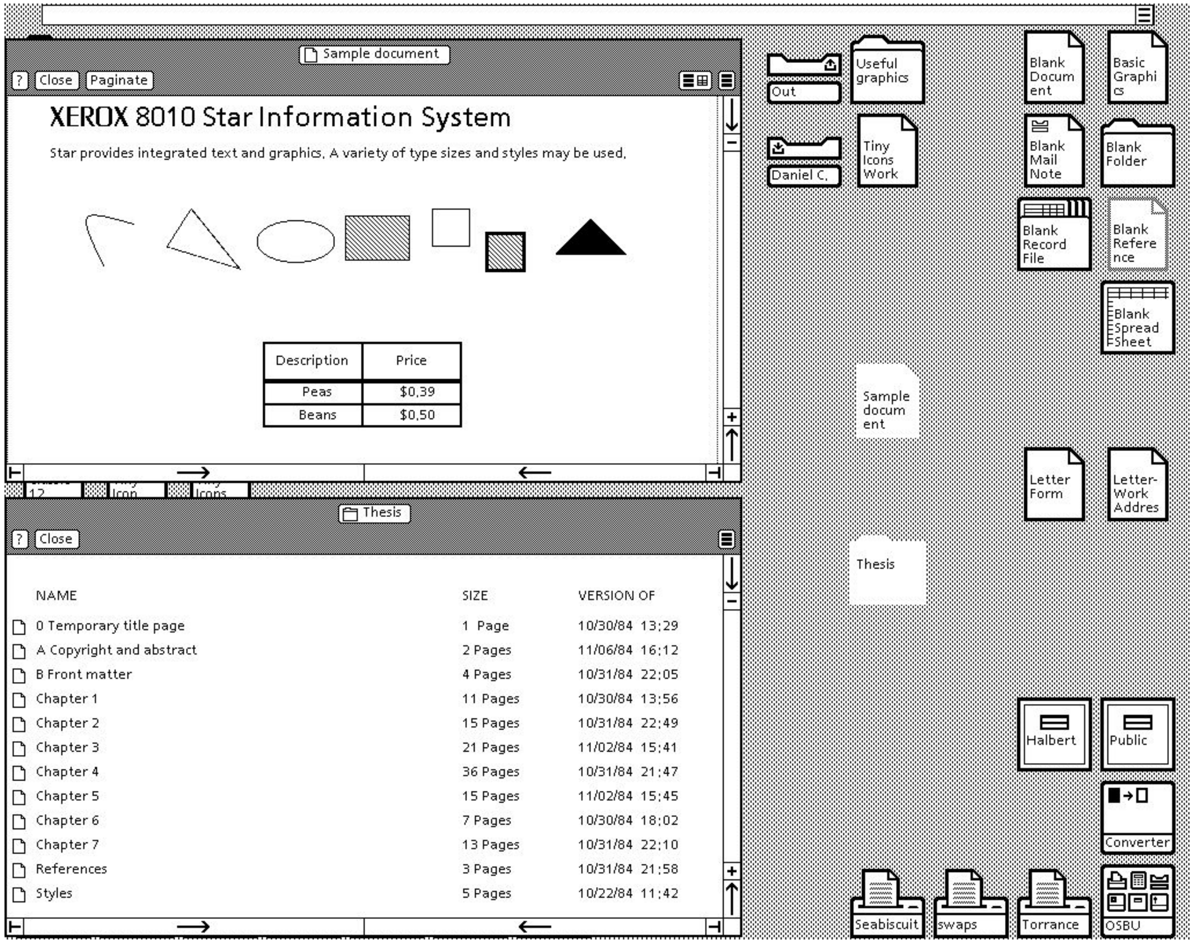
Smart phones become ubiquitous after the iPhone release in 2007

Timeline of OS

- IBM 704 was the first mass-produced computer
- GM-NAA (General Motors-North American Aviation) I/O is the OS used in the IBM 704
- There was no UI in the OS as we know now
- It was mostly based on command prompts (terminals)

Timeline of OS

- Xerox 8010 Star (released in 1981) was the first system that was referred to as a fully integrated desktop computer including applications and a GUI



XEROX 8010 Star Information System

Star provides integrated text and graphics. A variety of type sizes and styles may be used.

Description	Price
Peas	$0.39
Beans	$0.50

NAME	SIZE	VERSION OF
0 Temporary title page	1 Page	10/30/84 13:29
A Copyright and abstract	2 Pages	11/06/84 16:12
B Front matter	4 Pages	10/31/84 22:05
Chapter 1	11 Pages	10/30/84 13:56
Chapter 2	15 Pages	10/31/84 22:49
Chapter 3	21 Pages	11/02/84 15:41
Chapter 4	36 Pages	10/31/84 21:47
Chapter 5	15 Pages	11/02/84 15:45
Chapter 6	7 Pages	10/30/84 18:02
Chapter 7	13 Pages	10/31/84 22:10
References	3 Pages	10/31/84 21:58
Styles	5 Pages	10/22/84 11:42

Timeline of OS

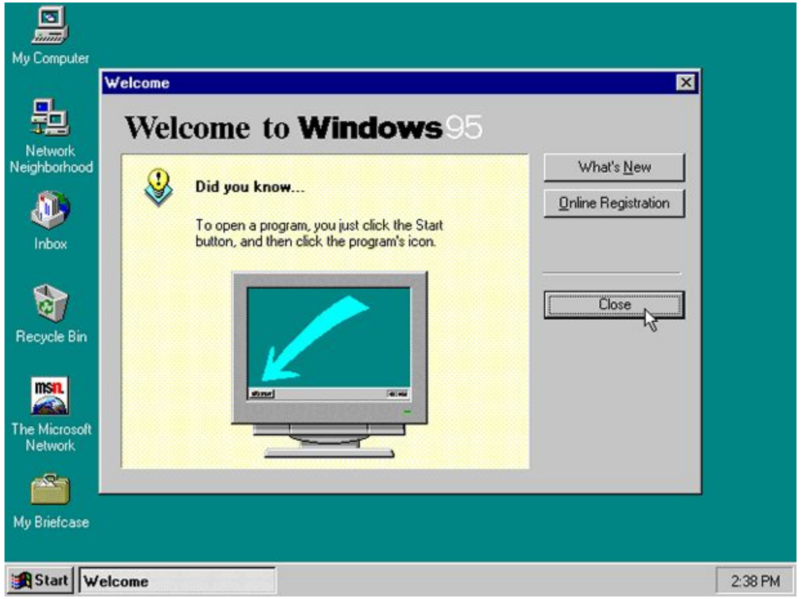
- Bill Gates got lucky to sign the so-called “Deal of the Century” with IBM to design a new OS for IBM machines
- The deal allowed Microsoft to add an OS with $50 per PC and IBM did not own copyright for the OS
- This allowed Microsoft to start the their OS dominance on the PC domain

Timeline of OS

- Developed by Apple Computer, Inc for their new product, the Macintosh home PC, The Macintosh 128K, was released in 1984
- was widely advertised (the famous 1984 commercial is available below).
- Mac OS was the first OS with a GUI built-in
- It was also one of the first consumer computers with mouse!
- Even though Microsoft introduced mouse in their PCs a bit earlier
- The use of GUI with mouse was not Steve Job’s idea, the idea was taken during his visit in Xerox PARC (Palo Aalto Research Center)
- Jobs reportedly traded US $1 million in stock options to Xerox for a detailed tour of their facilities and current projects
- One of the things Xerox showed Jobs was the Alto, which sported a GUI and a three-button mouse

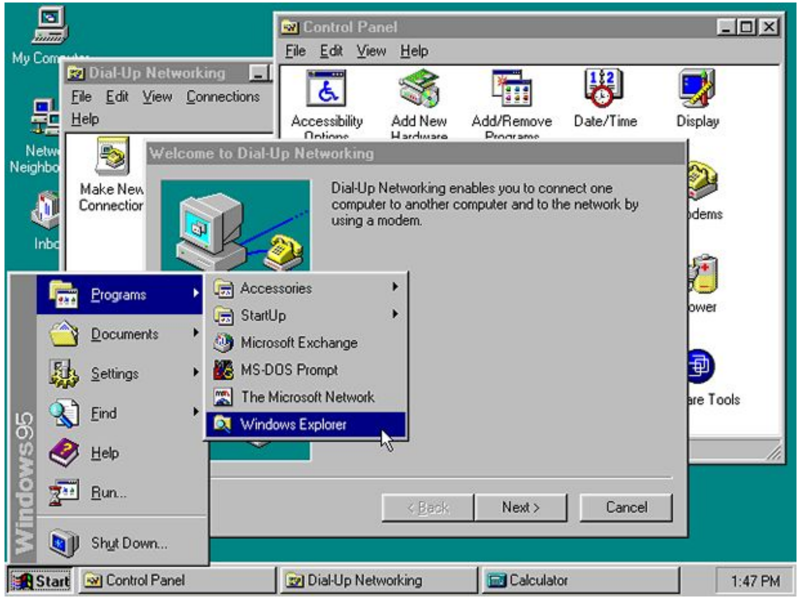
Timeline of OS

Timeline of OS


Welcome to Windows 95

Did you know...

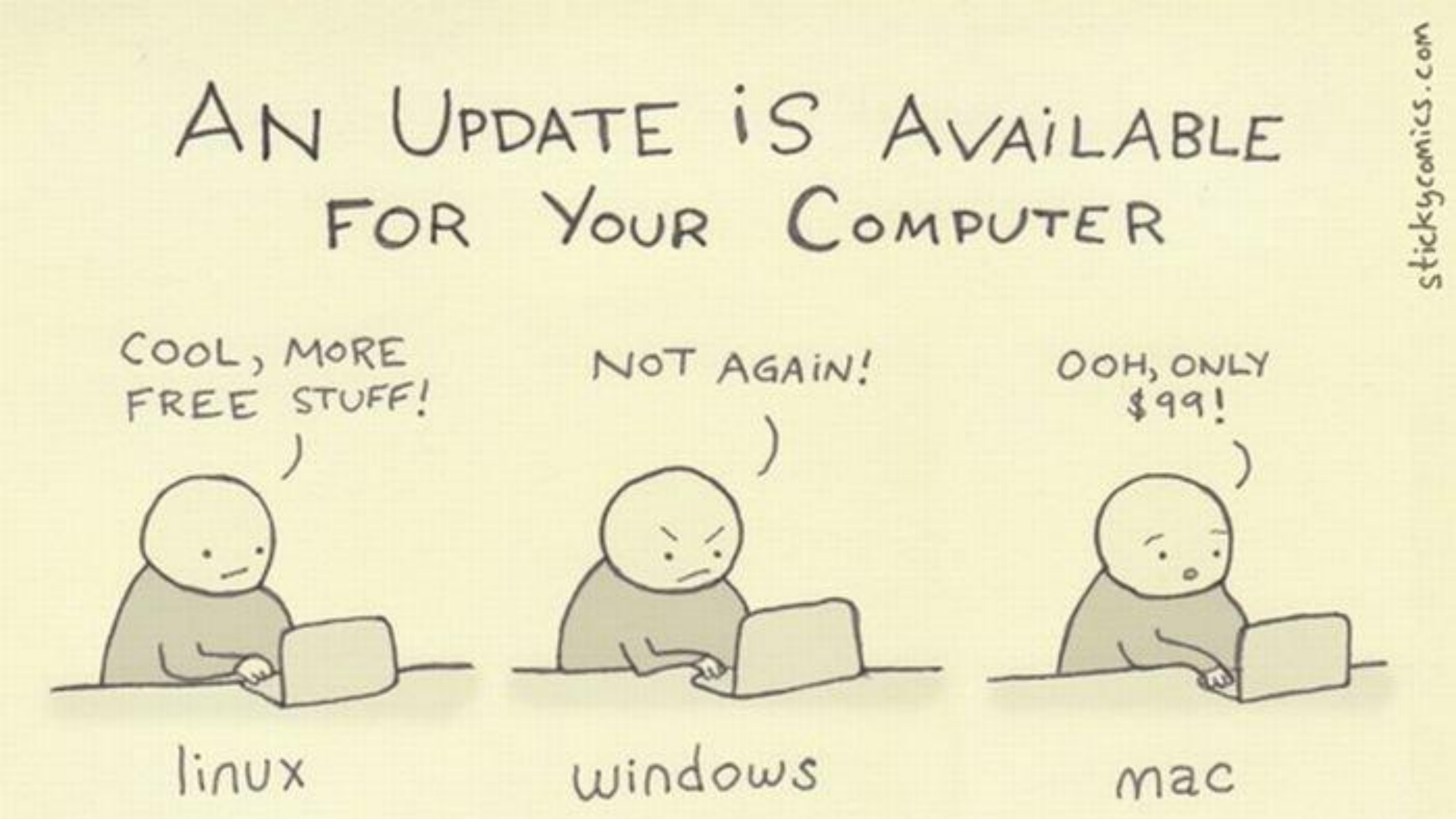
To open a program, you just click the Start button, and then click the program's icon.


Welcome to Dial-Up Networking

Dial-Up Networking enables you to connect one computer to another computer and to the network by using a modem.

Timeline of OS

- Another revolution of OS came in the mobile computing domain, when Steve Jobs introduced iPhone with iOS in 2007
- The iPhone introduction video is now regarded as a classic advertise video

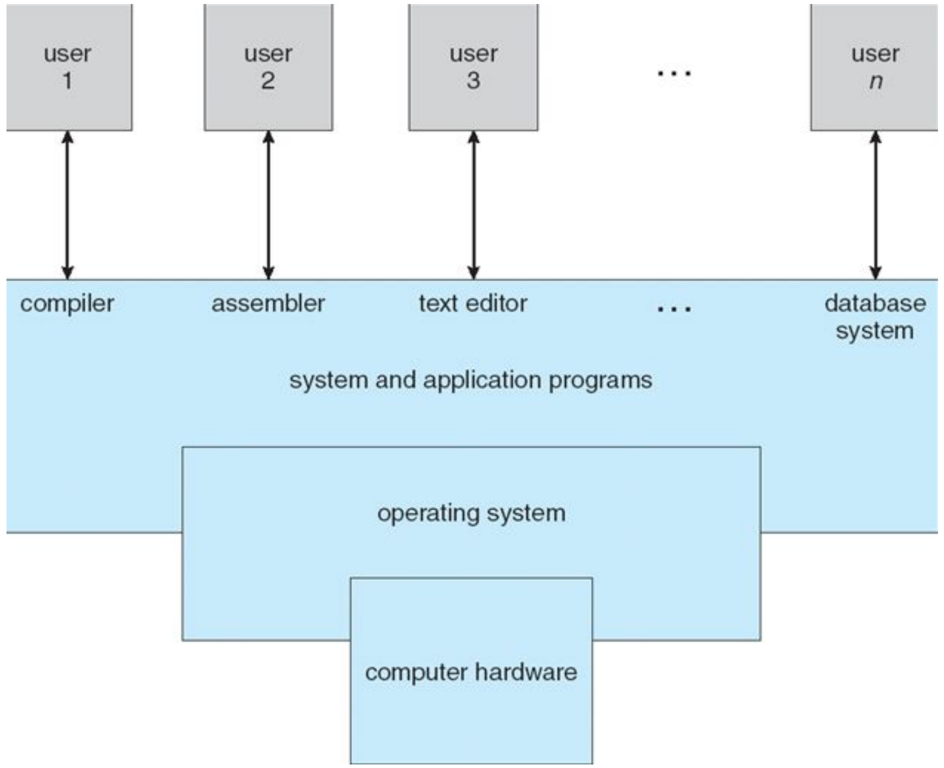

AN UPDATE IS AVAILABLE FOR YOUR COMPUTER
COOL, MORE FREE STUFF!
NOT AGAIN!
OOH, ONLY $99!
linux
windows
mac
stickycomics.com

Operating Systems

Computer System Organization

FBA

Components of a Computer System



Kernel

The one program running at all times.

- Kernel is the central module of an operating system
- Part of OS that loads first, and it remains in main memory.
- As small as possible
- Provide all the essential services required by other parts of the operating system and applications.
- Kernel code is usually loaded into a protected area of memory to prevent it from being overwritten.

Bootstrap Program

An initial program executed when a computer starts running.

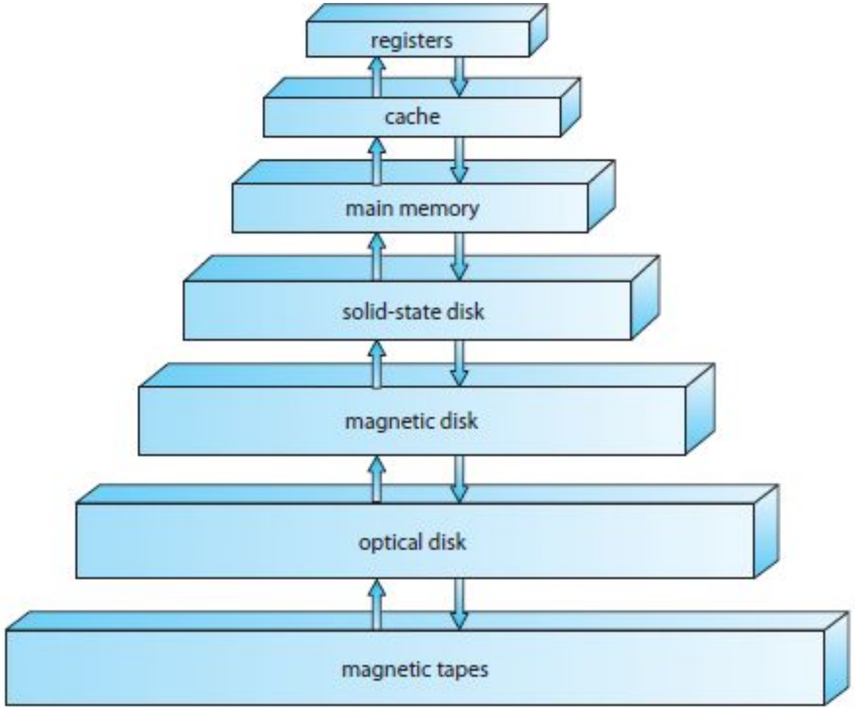
- When a computers is powered up or rebooted, it is executed first.
- Stored in the ROM or EEPROM, known as firmware
- Initializes all aspects of the system, from CPU registers to device controllers to memory contents
- bootstrap program must know how to load the operating system and how to start executing
- Once the OS kernel is loaded and executing, it can start providing services to the system and its users

Storage Structure

- ❑ Main memory – only large storage media that the CPU can access directly
  - ❑ Random access
  - ❑ Typically volatile
- ❑ Secondary storage – extension of main memory that provides large nonvolatile storage capacity

- CPU can load instructions only from main memory.
- General-purpose computers run most of their programs from rewritable memory, called main memory (also called RAM)
- Computers use other forms of memory as well - Read Only Memory (ROM) and electrically erasable programmable read-only memory (EEPROM)
- Only static programs, such as the bootstrap program described earlier, are stored here.

Storage Device Hierarchy



Operating Systems

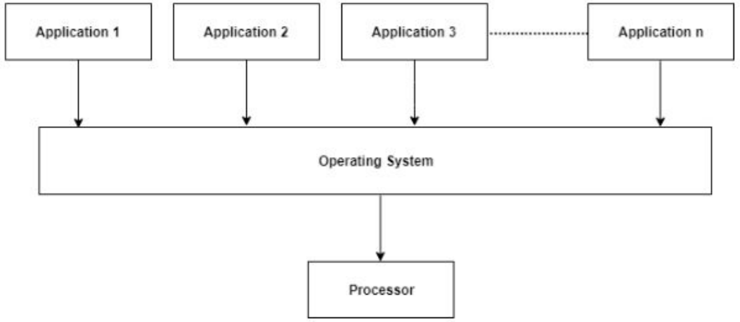
OS Architecture

FBA

Operating System Architecture

Single-Processor Systems:

- One main CPU capable of executing a general-purpose instruction set.
- Almost all single processor systems have other special-purpose processors (device-specific processors), which run a limiter instruction set.

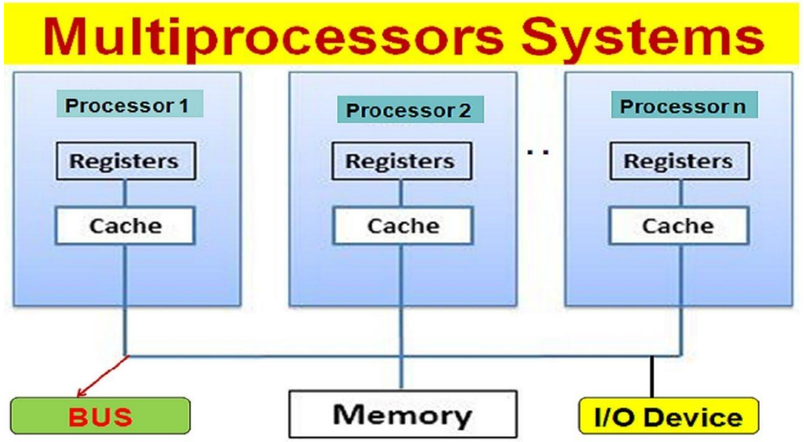


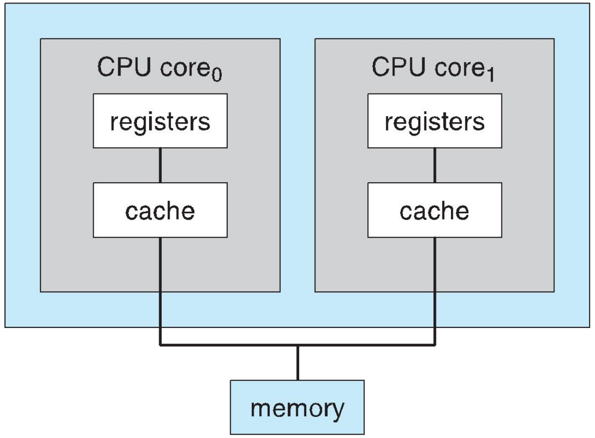
Single Processor System

Operating System Architecture

Multiprocessor Systems (parallel systems or multicore systems):

- Such systems have two or more processors in close communication, sharing the computer bus and sometimes the clock, memory, and peripheral devices.



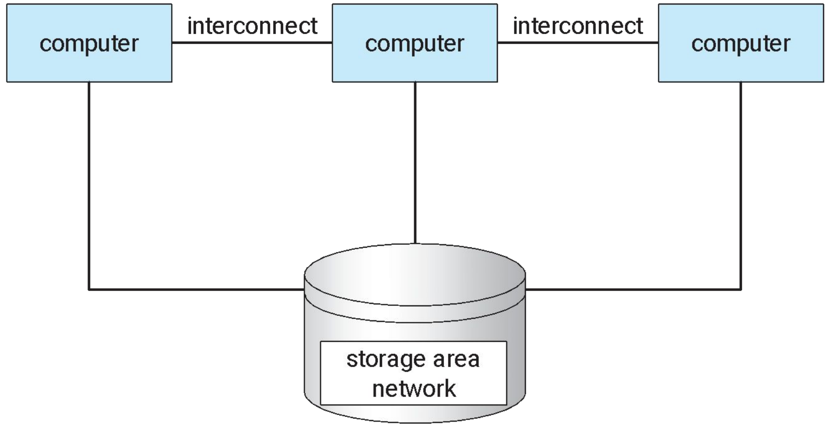


A dual-core system

Operating System Architecture

Clustered Systems:

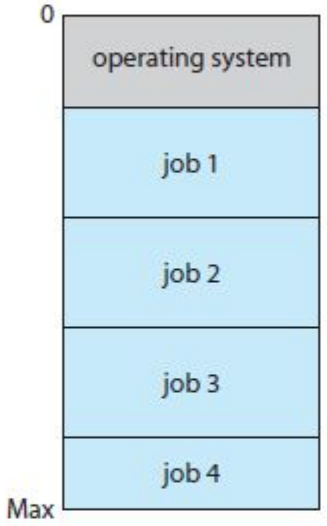
- Special kind of multiprocessor system which gathers together multiple CPUs. They are composed of two or more individual systems- or nodes - joined together.
- Clustered computers share storage and are closely linked via a local-area network (LAN) or a faster interconnect, such as InfiniBand



Operating System Structure

Multiprogramming:

- Increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute.
- OS keeps several jobs in memory simultaneously. As main memory is small, it keeps the jobs on the disk (job pool) which waits there to be allocated in the main memory.
- OS picks and begins to execute one of the jobs in memory. Eventually, the job may have to wait for some task. OS picks another job from the pool to execute while the previous one waits.



Memory layout for a multiprogramming system.

Requirements of Multiprogramming

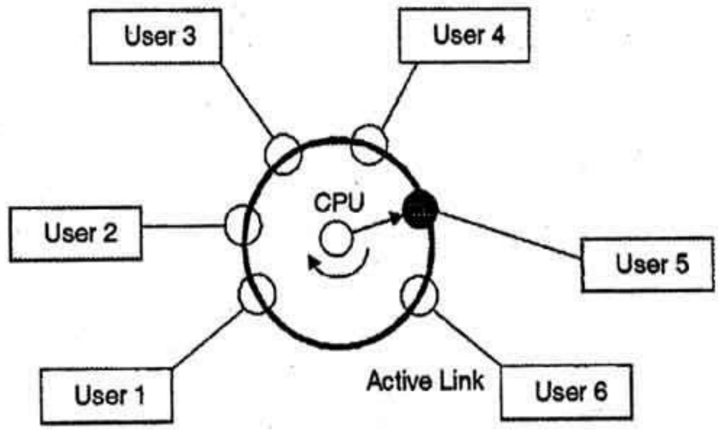
- **Job Scheduling:** If several jobs are ready to be brought into memory, and if there is not enough room for all of them, then the system must choose among them.
- When OS selects a job from the job pool, it loads that job into memory for execution. Having several programs in memory at the same time requires some form of memory management.
- **CPU Scheduling:** if several jobs are ready to run at the same time, the system must choose which job will run first.
- Running multiple jobs concurrently requires that their ability to affect one another be limited in all phases of the operating system.

★ If processes don't fit in memory, swapping moves them in and out to run from main memory achieving this goal is virtual memory.

Operating System Structure

Time Sharing:

- CPU executes multiple jobs by switching among them.
- Switches occur so frequently that the users can interact with each program while it is running.
- requires an interactive computer system, which provides direct communication between the user and the system.
- Response time should be short.

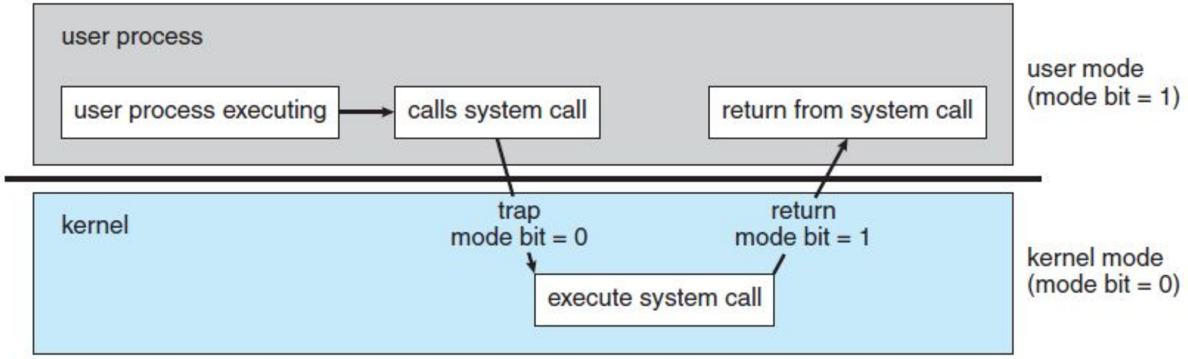


Operating System Operations

- Modern operating systems are interrupt driven
- Events are almost always signaled by the occurrence of an interrupt or a trap
- A trap (or an exception) is a software-generated interrupt
- For each type of interrupt, separate segments of code determine what action should be taken
- Errors can occur when one erroneous program modify another program, the data of another program, or even the operating system itself.
- A properly designed operating system must ensure that an incorrect (or malicious) program cannot cause other programs to execute incorrectly.

Dual Mode Operation

- Need to distinguish between the execution of operating-system code and user defined code.
- A bit, called the mode bit, is added to the hardware of the computer to indicate the current mode: kernel (0) or user (1).
- dual mode of operation provides protection of the operating system from errant users
- this protection is provided by designating some of the machine instructions that may cause harm as privileged instructions that are executed only in kernel mode.



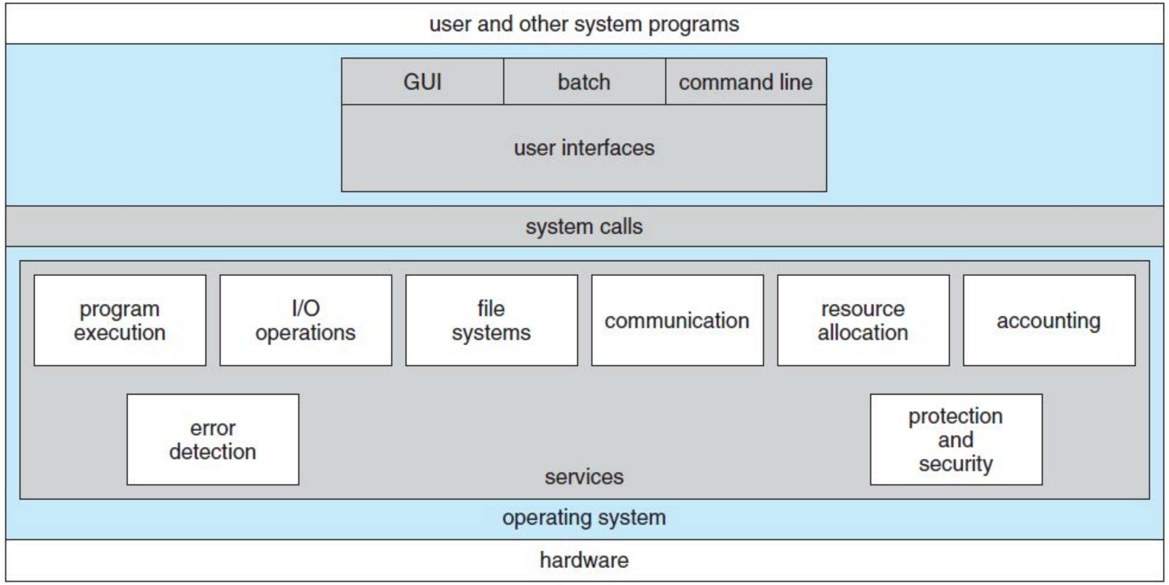
Transition from user to kernel mode.

Operating Systems

OS Services

Operating System Services

- OS provides an environment for the execution of programs.
- Specific services provided, differ from one operating system to another, but there are some common classes
- Services are provided for the convenience of the programmer



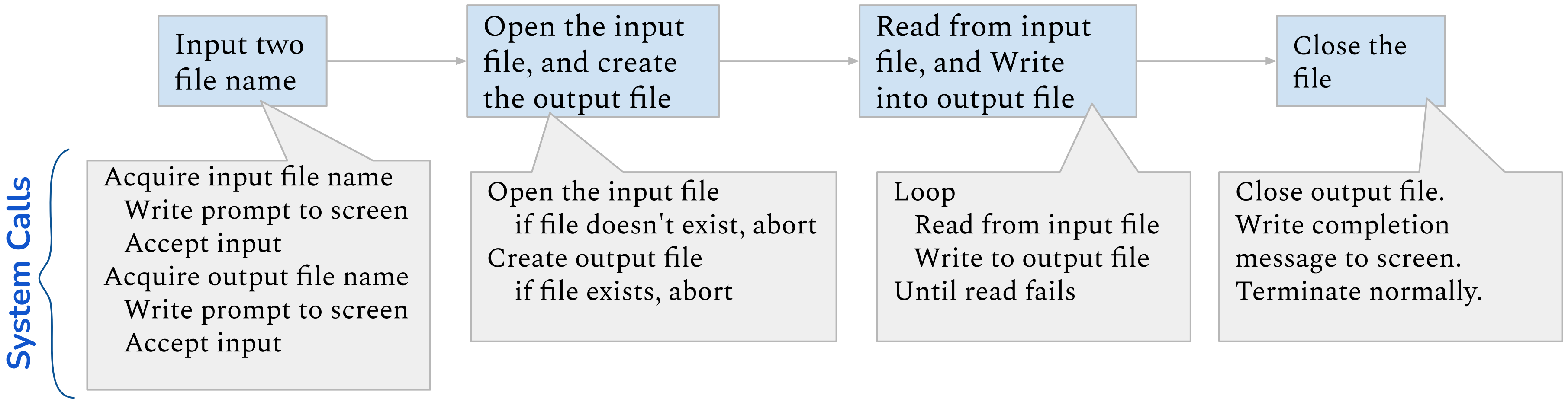
Operating Systems

System Call, System Program, System Boot

System Call

- System calls provide an interface to the services made available by an operating system.
- These calls are generally available as routines.
- Routines are written in C or C++. Some low level tasks are written in assembly language.

A program to copy the contents of a file to another file !



Input two file name → Open the input file, and create the output file → Read from input file, and Write into output file → Close the file

System Calls

```
Acquire input file name
  Write prompt to screen
  Accept input
Acquire output file name
  Write prompt to screen
  Accept input
```

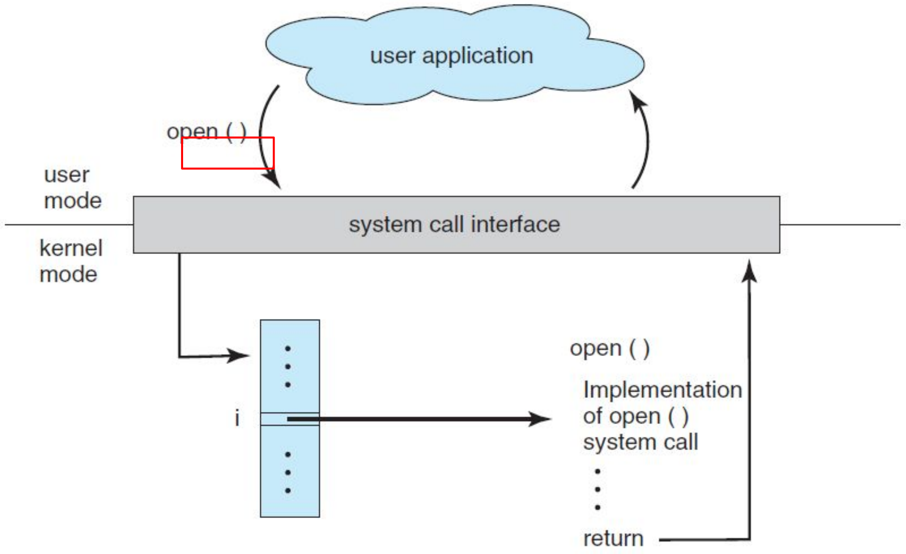
```
Open the input file
  if file doesn't exist, abort
Create output file
  if file exists, abort
```

```
Loop
  Read from input file
  Write to output file
Until read fails
```

```
Close output file.
Write completion
message to screen.
Terminate normally.
```

System Call Interface

- Serves as the link to system calls made available by the operating system.
- A number is associated with each system call, and the system-call interface maintains a table indexed according to these numbers.
- 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 Call

Type	Windows OS	Linux OS

Process Control	**CreateProcess()**
	ExitProcess()
	WaitForSingleObject()
		fork()
		exit()
		wait()

File Manipulation	**CreateFile()**
	ReadFile()
	WriteFile()
	CloseHandle()
		open()
		read()
		write()
		close()

Device Manipulation	**SetConsoleMode()**
	ReadConsole()
	WriteConsole()
		ioctl()
		read()
		write()

Types of System Call

Type	Windows OS	Linux OS

Information Maintenance	**GetCurrentProcessID()**
	SetTimer()
	Sleep()
		getpid()
		alarm()
		sleep()

Communication	**CreatePipe()**
	CreateFileMapping()
	MapViewOfFile()
		pipe()
		shm_open()
		mmap()

Protection	**SetFileSecurity()**
	InitlializeSecurityDescriptor()
	SetSecurityDescriptorGroup()
		chmod()
		umask()
		chown()

System Programs

System programs, also known as system utilities, provide a convenient environment for program development and execution.

These system programs provide -

- File management
- Status information
- File modification
- Programming-language support
- Program loading and execution
- Communications
- Background services

System Boot

- When power initialized on system, execution starts at a fixed memory location.
  - ➔ Firmware ROM used to hold initial boot code

- Operating system must be made available to hardware so hardware can start it.
  - ➔ Small piece of code – bootstrap loader, stored in ROM locates the kernel, loads it into memory, and starts it
  - ➔ Sometimes two-step process where boot block at fixed location loaded by ROM code, which loads bootstrap loader from disk

- Common bootstrap loader, GRUB, allows selection of kernel from multiple disks, versions, kernel options

- Kernel loads and system is then running.

Operating Systems

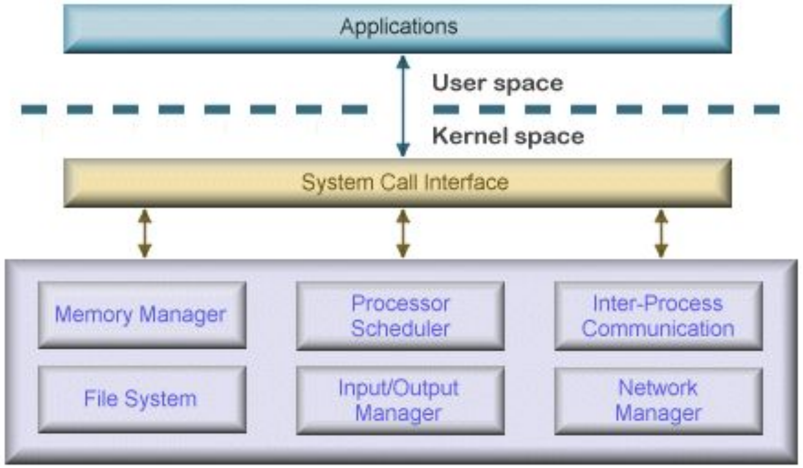
OS Structures

FBA

OS Structure

Simple/Monolithic structure:

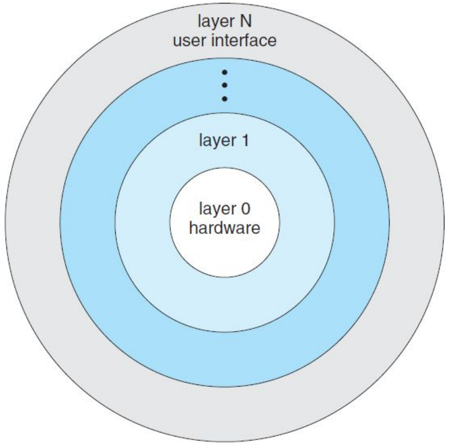
- Earliest and most common architecture
- Every component of OS is in the kernel and can communicate with each other directly
- Complex and Large (millions line of code ,) hard to maintain



OS Structure

Layered structure:

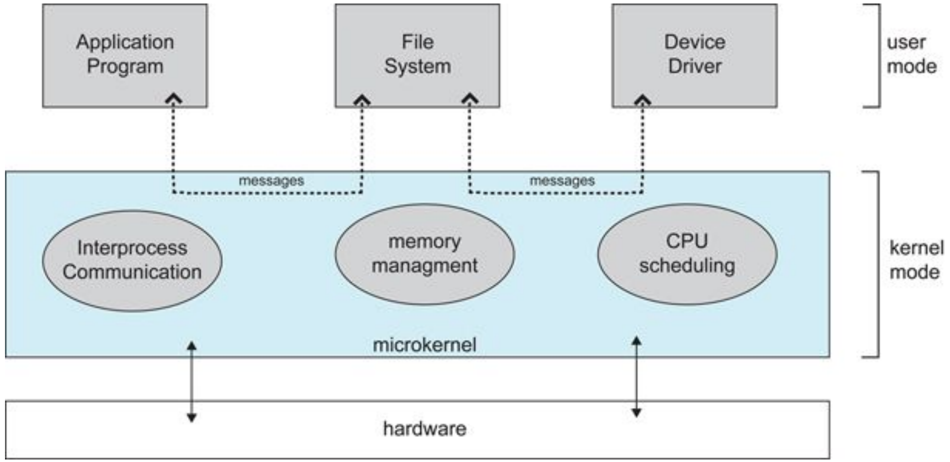
- OS is divided into layers.
- Each layer can use services of its lower layers.
- Easy to debug and develop.
- Less efficient as each layer adds some overhead



OS Structure

Microkernel structure:

- Moves as much from kernel into user space
- Communication takes place between user modules using message passing
- Easier to extend a microkernel
- More reliable(less code running in kernel mode) and secure
- Performance overhead



BRACE YOURSELF
THERE'S MORE TO COME
makeameme.org

Upcoming Episodes

Process:

- **Process States:** A condition of the process at a specific instant of time.
- **Process Architecture:** The structural design of general process systems.
- **Operations on Process:** process creation, preemption, blocking, and termination etc
- **Process Scheduling:** A task that schedules processes of different states like ready, waiting, and running
- **Process Synchronization:** The task of coordinating the execution of processes

Upcoming Episodes

Thread:

- **Definition & Purpose:** a sequential flow of tasks within a process.
- **Multithreading Models:** Ways of achieving multithreading
- **Thread Library:** provides the programmer with an Application program interface for creating and managing thread
- **Threading Issues:** Common problems and pitfalls with multi-thread programming

Upcoming Episodes

Deadlock:

- Reason
- Necessary Conditions for Deadlock
- Methods of Handling Deadlocks

Upcoming Episodes

Memory Management:

- **Background and importance of memory management**
- **Memory Management Techniques**
- **Virtual Memory**

THANK YOU
END OF LECTURE