Lecture 1 - Operating System Overview

Instructor : Bibhas Ghoshal (bibhas.ghoshal@iiita.ac.in)

Autumn Semester, 2015

Lecture Outline

- Defining an Operating System
- Computer System Structure
- Illustration of the role of Operating System
- Hardware control aspect of Operating System
- Goals of Operating System
- Multitasking and Multiuser Operating System
- Operating System structure

Some slides have been taken from:

- lecture slides of the book - Operating System Concepts by Silberschatz, Galvin and Gagne, 2005
- lecture slides of CS140 – (Instructors: Adam Belay et al.), Stanford University
- lecture slides of CSE 30341: Operating Systems (Instructor : Surendar Chandra),

What is an Operating system?

Abstract View

College Student - something that allows me to browse the internet on my smart phone

Salesman - allows me to use the sales package on the computer

Programmer - allows me to develop programs efficiently (utilizing the compiler's ability)

Technician at chemical plant - component of the computer system that controls the plant (control)

What is an Operating system?

Deductions from the four views of the OS

- All views are correct, but none complete
- Each view considers use of computer system for intended purpose but ignores other elements
- Commonality in the views : Utilization of computer system for specific purpose

OS : Something that allows the user to achieve the intended purpose of using the computer system in a fast and efficient manner

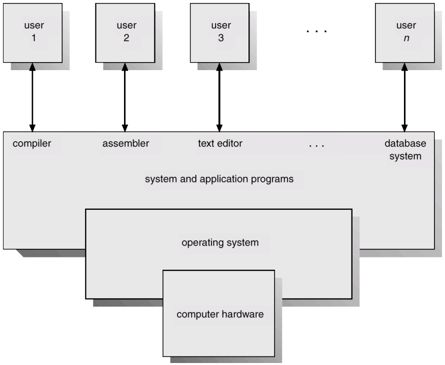
Computer System Components

- Computer system can be divided into :
  - Hardware – provides basic computing resources
    - Processing (CPU, graphics controller), storage(disk,memory), I/O devices (keyboard, mouse, CD/Tape, printer)
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
- Users (people, machines) want to run applications on hardware

Operating system controls and coordinates use of hardware among various applications and users

- User view : Concerned with ease of use
- System view : Resource utilization / controlling user programs and I/O

Abstract View of System Components



A C program to illustrate the use of OS - it has bugs !!!!

```
1) int main(int argc, char *argv[], char *envp[]) {
2) char buf[100];
3) char *ptr = malloc(100);
4) printf("Hello world\n");
5) int fd = open("/dir1/file", O_WRONLY, 0666);
6) *(ptr + 1000) = '\0';
7) write(fd, ptr, 100);
8) fsync(fd);
9) close(fd);
10) exit(0);
11) }
```

- Compile the program ($gcc file.c) - OS makes sure the file (residing in HDD) is provided and takes care of the H/W on which it runs
- array call (line 2) - OS allocates a space for array in the memory according to the required format and it ensures that when your friend runs the same program, it does not overwrite contents of your program.
- dynamic memory allocation (line 3) - provides more memory than alloted.
- library call (e.g printf in line 4)- OS makes sure to write the result on the screen when the library call is made.
- system call (e.g. open in line 5) - the OS interprets the string as a file/directory residing in hard disk and transfers it from hard disk to main memory

- segmentation fault (line 6) - you write 1000 bytes to the alloted 100 bytes. OS informs that some condition of the code has been violated.
  Two ways you mess up the code
  - you harm yourself - OS informs you
  - you harm others - OS ensures you do not do that; prevents writing to memory code alloted to others (open file in read only mode)
- write call (line 7) - OS system call transfering something from memory to disk
- exit code (line 10) - return to some other program (say console)

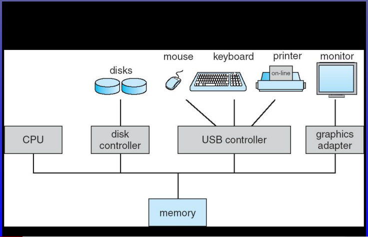
Key concerns of Operating System

- Programs - Initiation and termination
- Resources - Ensuring availability of resources and allocating them to programs
- Scheduling - Deciding when and how long to devote the CPU to a program
- Protection - Protect data and programs against interference from other users and their programs

Hardware Control Aspect of Operating System

Computer Hardware

- Components
  - Processing - (CPU) performs all computations
  - Input/Output - Mouse, Keyboard, printer
  - Storage - Memory, Hard disk, CD
- Hardware cost forces use of limited resources
  - Need to share resources - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Hardware Control Aspect of Operating System

- Technology limitations and costs leads to hierrchy of components
  - Storage
    - Hard disks are cheap, provide persistence and are slowdown
    - Main memory is expensive, not persistent but faster
    - Cache/Registers very expensive, not persistent but fastest
    - CD is persistent, very slow, cheap and removable
  - Processing
    - CPU is expensive and general purpose
    - Dedicated controllers (graphics processors, SCSI controllers)

Hardware Control Aspect of the Operating System

- OS acts as an intermediary between a user of a computer and the computer hardware
  - Challenge is to manage resources for competing users: simultaneously playing interactive games and sending a print out to an laser printer
- OS is a resource allocator
  - Manages all resources (CPU, memory, disks etc.)
  - Decides between conflicting requests for efficient and fair resource use
- OS is a control program
  - Controls execution of programs to prevent errors and improper use of the computer
- “The one program running at all times on the computer” is the kernel. Everything else is either a system program (ships with the operating system) or an application program

Goals of an Operating System

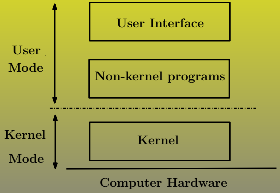


Figure : Abstract view of Operating System

- Layer between application and hardware
- Makes hardware useful to programmer
- Provides abstractions for applications
    - Manages and hides details of hardware
    - Accesses hardware through low level applications unavailable to applications
- Provides protection
    - Prevents one user from clobbering another

Operating System Structure

Primitive OS

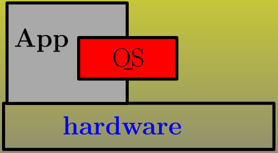


Figure : Abstract view of primitive OS

- Just a library of standard services [no protection]
- Standard interface above hardware-specific drivers, etc.
- System runs one program at a times
- Simplifying assumptions - No bad users or programs (often bad assumption)
- Poor utilization of hardware (e.g., CPU idle while waiting for disk)
- Poor uitilization of human user (must wait for each program to finish)

Operating System Structure

Features of Modern OS

- Multi-programming
- Multi-user OS
- Time sharing (multi-tasking)
- Protection mechanism

Multiprogramming

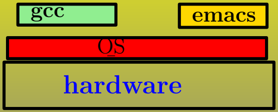
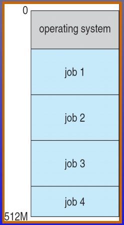


Figure : Multiprogramming

- Idea: Run more than one process at once; When one process blocks (waiting for disk, network, user input, etc.) run another process
- Problem: What can ill-behaved process do?
  - Go into infinite loop and never relinquish CPU
  - Scribble over other processes' memory to make them fail
- OS provides mechanisms to address these problems
  - Pre-emption – take CPU away from looping processed
  - Memory protection – protect process's memory from one another

Multiprogramming - needed for efficiency

- Single user cannot keep CPU and I/O devices busy at all times
- Multiprogramming organizes jobs (code and data) so CPU always has one to execute
- A subset of total jobs in system is kept in memory
- One job selected and run via job scheduling
- When it has to wait (e.g, for I/O), OS switches to another job
- Need to be careful for IO. For example, printer cannot be directly shared between two jobs



Memory Layout for Multiprogramming

Multi-user OS

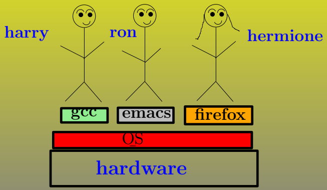


Figure : Multi-user OS

- Many OSes use protection to serve distrustful users/apps
- With N users, system not N times slower
  - Users' demands for CPU, memory, etc. are bursty
  - Win by giving resources to users who actually need them
- What can go wrong?
  - Users are gluttons, use too much CPU, etc. (need policies)
  - Total memory usage greater than in machine (must virtualize)
  - Super-linear slowdown with increasing demand (thrashing)

Multi-tasking

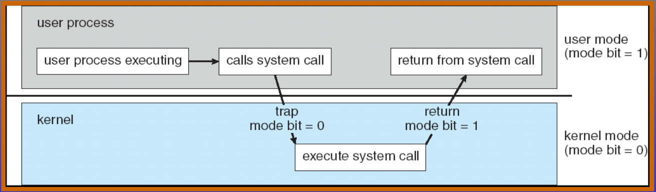
- Time sharing (multitasking) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing
  - Response time should be small ($<$ 1 second)
  - Each user has programs executing in memory $\Longrightarrow$ process abstraction
  - If several jobs ready to run at the same time $\Longrightarrow$ process management
  - If processes don't fit in memory, swapping moves them in and out to run $\Longrightarrow$ memory and storage management
  - Virtual memory allows execution of processes not completely in memory

Protection Structure

- OS should protect the users/processes from each other as well as protect itself from users
- Dual-mode operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as privileged, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user

Transition from User to Kernel Mode

- Timer to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time



Process Management

- A process is a program in execution. It is a unit of work within the system.
  Program is a passive entity, process is an active entity.
- Process needs resources to accomplish its task- CPU, memory, I/O, files and Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one program counter specifying location of next instruction to execute
- Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
- Concurrency by multiplexing the CPUs among the processes / threads

Process Management

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

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
- Optimizing CPU utilization and computer response to users
- Memory management activities
- Keeping track of which parts of memory are currently being used and by whom
- Deciding which processes (or parts thereof) and data to move into and out of memory
- Allocating and deallocating memory space as needed

Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - file
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Protection and Security

- Protection – any mechanism for controlling access of processes or users to resources defined by the OS
- Security – defense of the system against internal and external attacks
    - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do whatever
    - User identities (user IDs, security IDs) include name and associated number, one per user
    - User ID then associated with all files, processes of that user to determine access control
    - Group identifier (group ID) allows set of users to be defined and controls managed, then also associated with each process, file
    - Privilege escalation allows user to change to effective ID with more rights

Examples of Operating Systems

- Desktop and server OS: Microsoft Windows, UNIX and variants (Linux, Solaris, FreeBSD, Mac OSX)
  - Resource utilization is important
- Embedded and **Realtime** systems: smart phones, runs your cars, TV, washing machines, nuclear plants etc. (e.g. QNX, Vxworks, Embedded linux (Android))
  - Meeting timing constraints is a major concern
- **Distributed Systems** : Distribute the computation among several physical processors
  - Loosely coupled system – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines
  - Advantages of distributed systems - Resources Sharing, Reliability, Load sharing, Communication

Wrap Up

- Operating System is a program that acts as an intermediary between a user of a computer and the computer hardware, allowing users to conveniently and efficiently use the computer system so as to execute user programs and make problem easier.
- Operating Systems cannot make hardware go faster. However, OS can make h/w appear faster.
- Trade-off depends on specific operating scenario - distributed/centralized