Introduction to Operating Systems

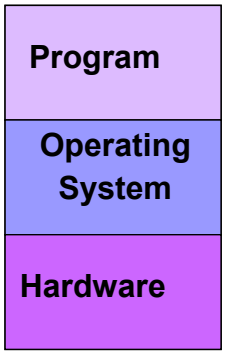
Computer Science Department
CS2211a: Software Tools & Systems Programming
Fall 2013
Instructor: Mahmoud R. El-Sakka
Office: MC-419
Email: elsakka@csd.uwo.ca
Phone: 519-661-2111 x86996

Introduction

- A computer consists of many hardware components
- If application programmers (e.g., *you*) had to understand how all these components work in details, no code would ever get written!!
- Even if programmers have such knowledge, maintaining and using such knowledge is an extremely challenging job
- If the hardware is changed, the program that drives this hardware must be changed as well
- For these reasons, a computer is equipped with a layer of software called the *operating system* that runs the computer

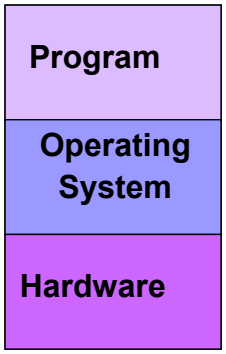
What is an Operating System?

- An operating system
  - runs directly on the hardware
  - is in charge of managing the hardware
  - hides the details of hardware from software – provides a much simpler interface for programs



Benefits of Operating Systems

- If the hardware is changed, the software (your program) should not – the operating system should handle it
- Carefully managing hardware resources
  - several programs can share the existing computer resources
- A software becomes much easier and cheaper to develop



Operating System Kernel

- When a computer is started (*booted*), it goes through a sequence of actions to initialize itself
- At the end of this process, it passes the control to a very complex program called *Kernel*
- The Kernel will keep running till the computer is shutdown
- The job of the kernel is to provide
  - Memory management
  - Process management
  - Inter-process communication
  - Input/output
  - File management
  - Security and access control
  - Network access

Operating System Kernel

- The kernel is the core of the operating system
- You can reach and deal with the kernel through the shell

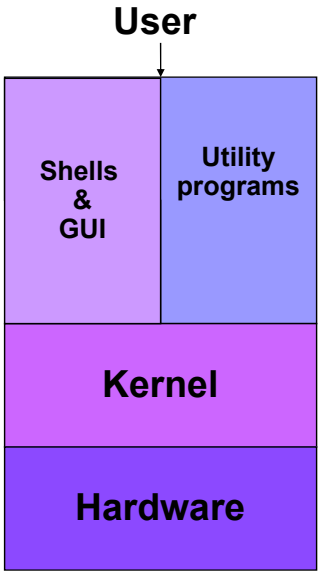
- A shell is a special type of program (*a command processor*) that surrounds the kernel and acts as our personal interface to the system

Operating System Kernel

- Kernels can be divided into two main categories
  - *Monolithic Kernels*
    - One very large program
    - Performs every thing by itself
    - Fast and efficient
    - Difficult to design and maintain
  - *Microkernels*
    - A much smaller program
    - Performs the most basic tasks only
    - To perform the rest of the functions, a microkernel calls upon a set of other programs (called *servers*)
      - NB: *servers* are ***programs***, ***not machines***
    - Slower and less efficient
    - A lot easier to design and maintain (due to *modular design*)

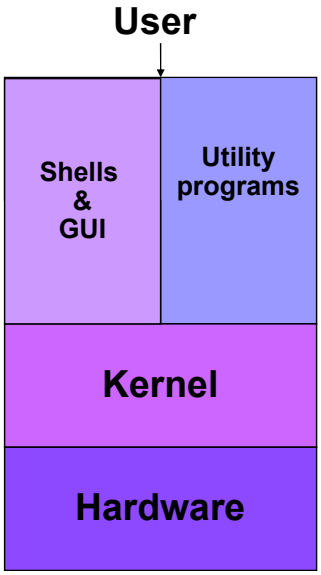
Operating Systems

- Operating systems are packaged with many *utility programs* that allow users to edit files, communicate with each other, use the internet, develop programs, …
- The *utility programs* also include those that provide interface with the computer, e.g.,
  - *Shell programs*
    provide text-based interface
  - *GUI programs*
    provide graphical user interface



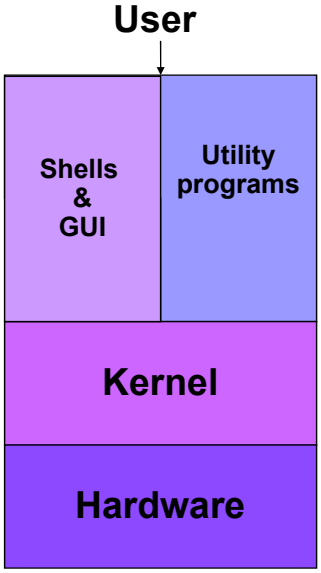
Unix Operating Systems

- A Unix is an operating system consists of
  - *A* Unix *kernel*
  - Unix *utility programs*
- A Unix operating system can
  - Run more than one program at a time (*multitasking*)
  - Support more than one user at a time (*multiuser*)
- FYI: Microsoft Windows is a *multitasking, **single-user*** operating system



Unix Operating Systems

- Unix utilities work pretty much the same on every Unix system
- Unix utilities are simple
- Each utility does one thing, and does it really well
- More complex tasks can be accomplished by combining existing utilities together in scripts or pipelines



Unix Operating Systems

- Unix was *not* designed to be *learned*!!
- Unix was designed to be *used*

- Unix was *not* meant to be a *user-friendly* operating system
- Unix was meant to be a *user-helpful* operating system

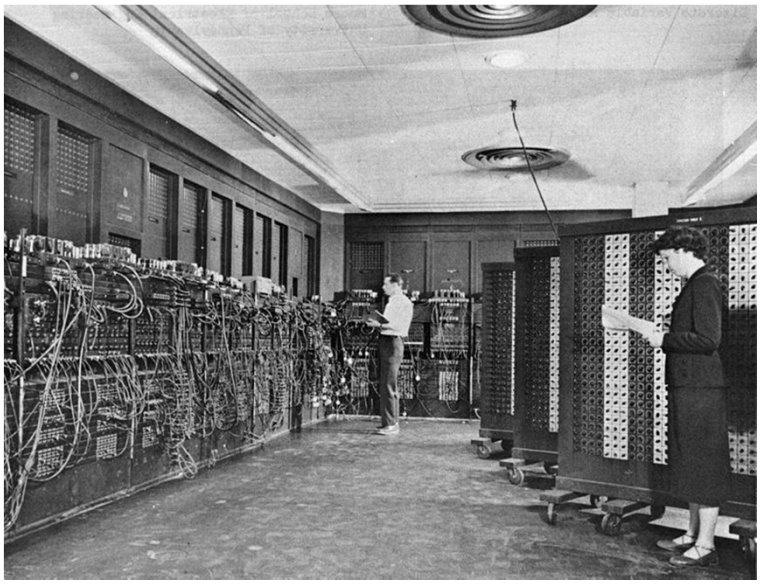
- It can be *confusing* and *time consuming* to learn Unix
- Yet, once you have mastered the skills you need, working with Unix becomes *easier* and *faster*

- It is exactly like learning how to drive a car

History of Operating Systems

- Operating systems
  - have been closely tied to the architecture of computers on which they run
  - can be roughly mapped to computer generations
- 1st generation (mid 40's to mid 50's) vacuum tubes
  - Programming was done by wiring up electrical circuits *can you image how the life of a programmer was ?*
  - Programming languages were unknown (even assembly)
  - Operating systems were unheard of

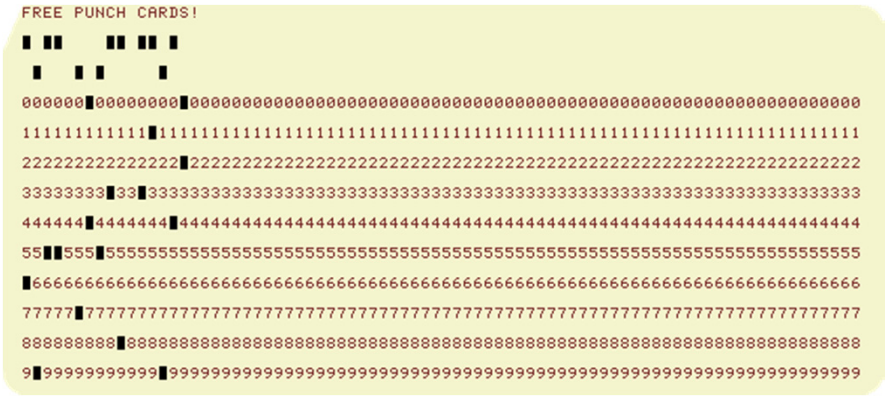
History of Operating Systems



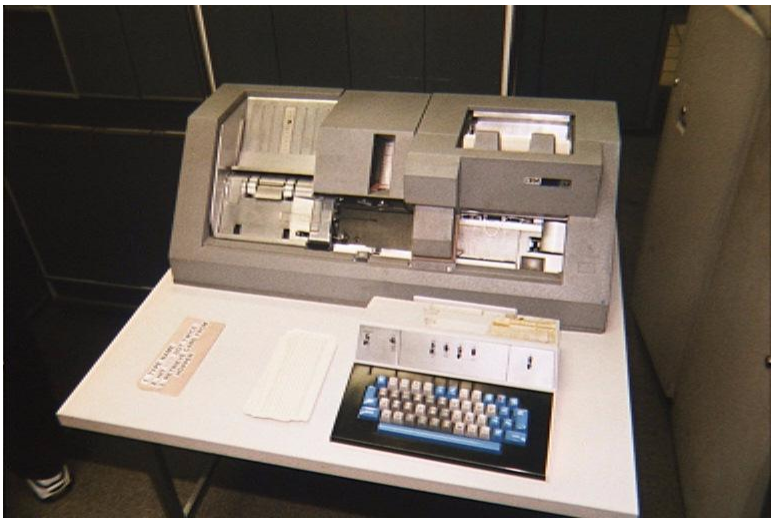
History of Operating Systems

- 2nd generation (mid 50's to mid 60's) transistors and batch systems
  - With transistors, computers became reliable enough that they could be manufactured and sold to customers
  - One computer may have cost *multimillion dollars*
  - Machines (*mainframes*) are locked away in specially air conditioned rooms
  - *FORTRAN* and *assembly* were the dominant programming languages
  - Offline punch machines were utilized
  - Tapes were the main storage devices
  - Much of the computer time was wasted by operators walking around the machine room
  - Batch systems were utilized

History of Operating Systems



History of Operating Systems



History of Operating Systems

History of Operating Systems

- 3rd generation (mid 60's to early 80's) ICs and multiprogramming
  - Small-Scale Integrated (SSI) circuits were utilized
  - Scalability (IBM/360 series) with backward compatibility
  - Teletype and screen terminals
  - Disks were used as storage devices
  - Spooling (*Simultaneous Peripheral Operation On Line*)
  - Time sharing
  - *M.I.T.*, *Bell Labs* and *General Electric* decided to develop their own time sharing model called *MULTIplexed Information and Computing Service* (MULTICS)
  - While Bell Labs and General Electric dropped out of the MULTICS project, M.I.T. persisted until MULTICS got working

History of Operating Systems

- *Digital Equipment Corporation* (DEC) started to build its PDP minicomputers series
- PDP-1 has 4K of RAM at $120,000 per machine
- Unlike IBM/360, PDP series were not backward compatible
- Unix was developed in 1969 on a PDP-7 with 8K of RAM

History of Operating Systems

Ken Thompson (sitting), Dennis Ritchie (Standing), and PDP-11

History of Operating Systems

- 4rd generation (early 80's till now) Personal Computers
  - Large-Scale Integrated (LSI) circuits were utilized
  - A PC was initially called a microcomputer
  - It became possible for a single individual to have a PC
  - In 1974, Intel came out with its 8080 micro processor
  - Gary Kildall wrote an operating system for the 8080 processor, which was called *Control Program for Microcomputer* (CP/M)
  - In the early 1980's, IBM designed its own PC
  - IBM contacted Microsoft (Bill Gates) to license its BASIC interpreter and to ask for an operating system
  - Bill Gates recommended CP/M, but Gary Kildall refused to meet IBM and to sign any non-disclosure agreement with IBM
  - At that time, Bill Gates bought an operating system called *Disk Operating System* (DOS) for $75,000 from Seattle Computer Products

History of Operating Systems

- Bill Gates decided to sell DOS/BASIC to computer companies for bundling with their hardware (at least in the early days), not to the end user as the case of CP/M
- Gary Kildall died and CP/M's progress was put to an end
- DOS kept producing new versions to accommodate the available new hardware
- From 1985-1995, Windows was just a graphical environment under DOS
- Various versions of the Windows operating system were introduced
- Linux (Unix based operating system) is becoming a popular alternative to Windows for many students and companies

History of Operating Systems