COMP 3318 – Operating Systems Spring 2015

Ahmet E Sonmez, Dr.

Class: M W 11:30 am – 1:00 pm Room: 815
Office: 808
Phone: (832) 230-5130
Office Hours:
MW: 2pm-3:30pm
Additional hours will be posted
aesonmez@na.edu

Course Description

- This course focuses on operating system structure and design techniques; process management, CPU and disk scheduling; process synchronization, concurrency, and memory and file management, virtual memory; mass storage and I/O systems and OS security.

Assessment Criteria

Quizzes	20%
HW	20%
Presentation	10%
Midterm	20%
Final	30%

A	96-100
A-	91-95
B+	86-90
B	81-85
B-	76-80
C+	71-75
C	66-70
C-	61-65
D+	56-60
D	50-55
F	Below 50

Important Dates

- MIDTERM :10/10/2016
- FINAL : To Be Announced

Understanding Operating Systems Sixth Edition

Chapter 1
Introducing Operating Systems

Learning Objectives

After completing this chapter, you should be able to describe:

- Innovations in operating systems development
- The basic role of an operating system
- The major operating system software subsystem managers and their functions
- The types of machine hardware on which operating systems run

Learning Objectives (cont'd.)

- The differences among batch, interactive, real-time, hybrid, and embedded operating systems
- Design considerations of operating systems designers

Introduction

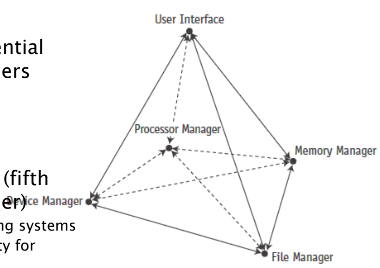
- Operating systems
  - Manage computer system hardware and software
- We will explore:
  - What they are
  - How they work
  - What they do
  - Why they do it
- This chapter describes:
  - How operating systems work
  - The evolution of operation systems

What is an Operating System?

- **Computer System**
  - Software (programs)
  - Hardware (physical machine and electronic components)
- **Operating System**
  - Part of computer system (software)
  - Manages all hardware and software
    - Controls every file, device, section of main memory and nanosecond of processing time
    - Controls *who* can use the system
    - Controls *how* system is used

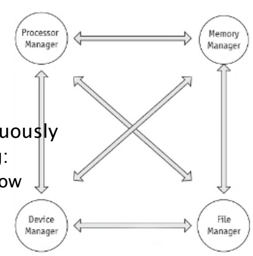
Operating System Software

- Includes four essential subsystem managers
  - Memory Manager
  - Processor Manager
  - Device Manager
  - File Manager
- Network Manager (fifth subsystem manager)
  - In all modern operating systems
  - Assumes responsibility for networking tasks
  - Discussed further in Chapters 9 & 10



Operating System Software (cont'd.)

- Each manager:
  - Works closely with other managers
  - Performs a unique role
- Manager tasks
  - Monitor its resources continuously
  - Enforce policies determining:
    - Who gets what, when, and how much
  - Allocate the resource (when appropriate)
  - [illegible] source (when appr[illegible]



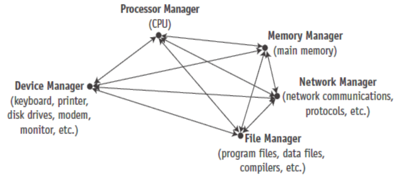
Operating System Software (cont'd.)

- **Network Manager**
  - Coordinates the services required for multiple systems to work cohesively together
    - Shared network resources: memory space, processors, printers, databases, applications, etc.

Operating System Software (cont'd.)

(figure 1.2)

Networked systems have a Network Manager that assumes responsibility for networking tasks while working harmoniously with every other manager.



Main Memory Management

- In charge of main memory
  - Random Access Memory (RAM)
- Responsibilities include:
  - Checking validity and legality of memory space request
  - Reallocating memory to make more useable space available
  - Deallocating memory to reclaim it
  - Protecting space in main memory occupied by operating system

Main Memory Management

- Read-only memory (ROM)
  - Another type of memory
  - Critical when computer is powered on
  - Holds firmware: programming code
    - When and how to load each piece of the operating system after the power is turned on
  - Non-volatile
    - Contents lost when the power is turned off

Processor Management

- In charge of allocating **Central Processing Unit** (CPU)
- Tracks **process** status
  - Program's "instance of execution"
- Two levels of responsibility:
  - Handle jobs as they enter the system
    - Handled by Job Scheduler
  - Manage each process within those jobs
    - Handled by Process Scheduler

Device Management

- In charge of connecting with every available device
  - Printers, ports, disk drives, etc.
- Responsibilities include:
  - Choosing most efficient resource allocation method
    - Based on scheduling policy
  - Identifying the device
  - Starting device operation
  - Monitoring device progress
  - Deallocating the device

File Management

- In charge of tracking every file in the system
  - Data files, program files, compilers, application programs
- Responsibilities include:
  - Enforcing user/program resource access restrictions
    - Uses predetermined access policies
  - Controlling user/program modification restrictions
    - Read-only, read-write, create, delete
  - Allocating resource
    - Opening the file
    - Deallocating file (by closing it)

User Interface

- Portion of the operating system
  - Direct interaction with users
- Two primary types
  - Graphical user interface (GUI)
    - Input from pointing device
    - Menu options, desktops, and formats vary
  - Command line interface
    - Keyboard-typed commands that display on a monitor
    - Strict requirements for every command: typed accurately; correct syntax; combinations of commands assembled correctly

Cooperation Issues

- No single manager performs tasks in isolation
- Each element of an operating system
  - Performs individual tasks *and*
  - Harmoniously interacts with other managers
    - Incredible precision required for operating system to work smoothly
    - More complicated when networking is involved

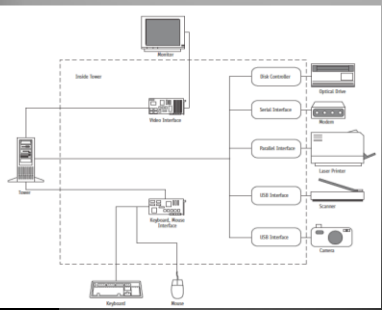
A Brief History of Machine Hardware

- **Hardware:** physical machine and electronic components
  - **Main memory (RAM)**
    - Data/Instruction storage and execution
  - **Input/Output devices (I/O devices)**
    - All peripheral devices in system
    - Printers, disk drives, CD/DVD drives, flash memory, and keyboards
  - **Central processing unit (CPU)**
    - Controls interpretation and execution of instructions
    - Controls operation of computer system

A Brief History of Machine Hardware (cont'd.)

(figure 1.5)

A logical view of a typical computer system hardware configuration. The tower holds the central processing unit, the arithmetic and logic unit, registers, cache, and main memory, as well as controllers and interfaces shown within the dotted lines.



A Brief History of Machine Hardware (cont'd.)

- Computer classification
  - By capacity and price (until mid-1970s)
- Mainframe
  - Large machine
    - Physical size and internal memory capacity
  - Classic Example: 1964 IBM 360 model 30
    - CPU required 18-square-foot air-conditioned room
    - CPU size: 5 feet high x 6 feet wide
    - Internal memory: 64K
    - Price: $200,000 (1964 dollars)
  - Applications limited to large computer centers

IBM 360 Model 40

Figure 1. IBM System/360 Model 40 Data Processing System

A Brief History of Machine Hardware (cont'd.)

- **Minicomputer**
  - Developed for smaller institutions
  - Compared to mainframe smaller in size and memory capacity
    - Cheaper
  - Example: Digital Equipment Corp. minicomputer
    - Price: For PDP8 less than $18,000
    - Video 1
    - Video 2
  - Today
    - Known as **midrange computers**
    - Capacity between microcomputers and mainframes

A Brief History of Machine Hardware (cont'd.)

- **Supercomputer**
  - Massive machine
  - Developed for military operations and weather forecasting
  - Example: Cray supercomputer
    - 6 to 1000 processors
    - Performs up to 2.4 trillion floating-point operations per second (teraflops)
  - Uses:
    - Scientific research Video
    - Customer support/product development

A Brief History of Machine Hardware (cont'd.)

- **Microcomputer**
  - Developed for single users in the late 1970s
  - Example: microcomputers by Tandy Corporation and Apple Computer, Inc.
    - Very little memory (by today's standards)
    - 64K maximum capacity
  - Microcomputer's distinguishing characteristic
    - Single-user status

A Brief History of Machine Hardware (cont'd.)

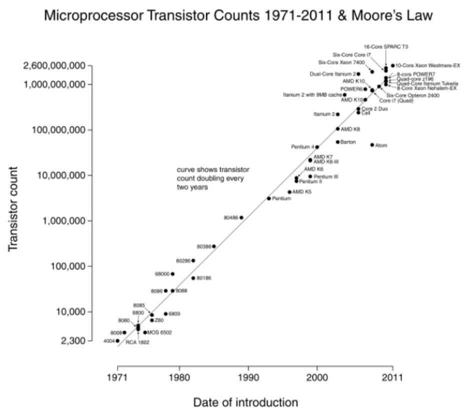
- **Workstations**
  - Most powerful microcomputers
  - Developed for commercial, educational, and government enterprises
  - Networked together
  - Support engineering and technical users
    - Massive mathematical computations
    - Computer-aided design (CAD)
  - Applications
    - Requiring powerful CPUs, large main memory, and extremely high-resolution graphic displays

A Brief History of Machine Hardware (cont'd.)

- **Servers**
  - Provide specialized services
    - To other computers or client/server networks
  - Perform critical network task
  - Examples:
    - Print servers
    - Internet servers
    - Mail servers

A Brief History of Machine Hardware (cont'd.)

- **Advances in computer technology**
  - Dramatic changes
    - Physical size, cost, and memory capacity
  - Networking
    - Integral part of modern computer systems
  - Mobile society information delivery
    - Creating strong market for handheld devices
  - New classification
    - By processor capacity, not memory capacity
  - Moore's Law
    - Computing power rises exponentially



An Evolution of Computing Hardware (cont'd.)

- Computer classification
  - At one time: based on memory capacity
- Current platforms

Platform	Operating System
Telephones, tablets	Android, iOS, Windows
Laptops, desktops	Linux, Mac OS X, UNIX, Windows
Workstations, servers	Linux, Mac OS X Server, UNIX, Windows Server
Mainframe computers	Linux, UNIX, Windows, IBM z/OS
Supercomputers	Linux, UNIX

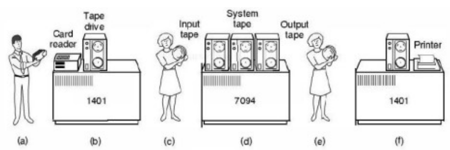
(table 1.1)
A brief list of platforms and a few of the operating systems designed to run on them, listed in alphabetical order.

Types of Operating Systems

- Five categories
  - Batch
  - Interactive
  - Real-time
  - Hybrid
  - Embedded
- Two distinguishing features
  - Response time
  - How data enters into the system

Types of Operating Systems (cont'd.)

- **Batch Systems** (1955-1965)
  - Input relied on punched cards or tape
  - Efficiency measured in throughput



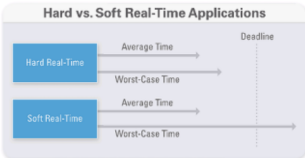
An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

Types of Operating Systems (cont'd.)

- **Interactive Systems**
  - Faster turnaround than batch systems
  - Slower than real-time systems
  - Introduced to provide fast turnaround when debugging programs
  - Time-sharing software developed for operating system

Types of Operating Systems (cont'd.)

- **Real-time systems**
  - Reliability is critical
  - Fast and time limit sensitive
  - Used in time-critical environments
    - Space flights, airport traffic control, high-speed aircraft
    - Industrial processes
    - Sophisticated medical equipment
    - Distribution of electricity
    - Telephone switching
  - Must be 100% responsive, 100%



Example: Hard Real Time Sytem

- *Hard real-time systems*
  - Must guarantee that all deadlines will always be met
  - ***Any failure*** could have ***catastrophic consequences***:
    - The reactor could overheat and explode
    - The rocket could be lost

Example: Soft Real Time System

- *Soft real-time systems*
  - Guarantee that most deadlines will be met
  - A DVD decoder that miss a deadline will spoil our viewing pleasure for a fraction of a second

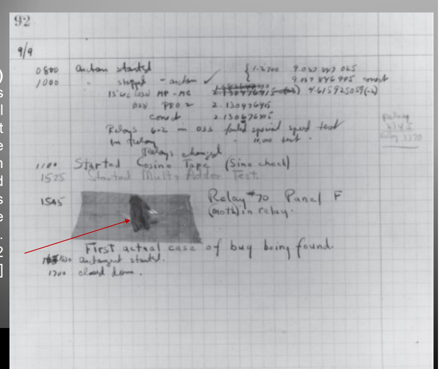
Types of Operating Systems (cont'd.)

- **Hybrid systems**
  - Combination of batch and interactive
  - Accept and run batch programs in the background
    - Interactive load is light
- **Embedded systems**
  - Computers placed inside other products
  - Adds features and capabilities
  - Operating system requirements
    - Perform specific set of programs
    - Not interchangeable among systems
    - Small kernel and flexible function capabilities

Brief History of Operating Systems Development

- **1940s: first generation**
  - Computers based on vacuum tube technology
  - No standard operating system software
  - Typical program included every instruction needed by the computer to perform the tasks requested
  - Poor machine utilization
    - CPU processed data and performed calculations for fraction of available time
  - Early programs
    - Designed to use the resources conservatively
    - Understandability is not a priority

(figure 1.8)
Dr. Grace Hopper's research journal included the first computer bug, the remains of a moth that became trapped in the computer's relays, causing the system to crash. [Photo © 2002 IEEE]



Brief History of Operating Systems Development (cont'd.)

- **1950s: second generation**
  - Focused on cost effectiveness
  - Computers were expensive
    - IBM 7094: $200,000
  - Two widely adopted improvements
    - Computer operators: humans hired to facilitate machine operation
    - Concept of job scheduling: group together programs with similar requirements
  - Expensive time lags between CPU and I/O devices

Understanding Operating Systems, Sixth Edition

Brief History of Operating Systems Development (cont'd.)

- **1950s: second generation (cont'd.)**
  - I/O device speed gradually became faster
    - Tape drives, disks, and drums
    - (Tape drives are still in use LTO 6)
  - Records blocked *before* retrieval or storage
  - Buffer between I/O and CPU introduced
    - Reduced speed discrepancy
  - Timer interrupts developed
    - Allowed job-sharing
    - Prevent infinite loops in programs

Understanding Operating Systems, Sixth Edition

Types of Interrupts

- ***I/O completion interrupts***
  - Notify the OS that an I/O operation has completed,
- ***Timer interrupts***
  - Notify the OS that a task has exceeded its quantum of CPU time,
- ***Traps***
  - Notify the OS of a ***program error*** (division by zero, illegal op code, illegal operand address, ...) or a *hardware failure*
- ***System calls***
  - Notify OS that the running task wants to submit a request to the OS

Interrupts

- When an interrupt occurs:
  a) The ***current state of the CPU*** (program counter, program status word, contents of registers, and so forth) is saved, normally on the top of a stack
  b) A ***new CPU state*** is fetched
- New state includes a new ***hardware-defined*** value for the program counter

Lecture Notes from Dr. Paris, University of Houston

Context Switches

- Each interrupt will result into t***wo context switches***:
  - One when the running task is interrupted
  - Another when it regains the CPU
- Context switches are ***not cheap***
- The overhead of any simple system call is ***two context switches***

Lecture Notes from Dr. Paris, University of Houston

Prioritizing Interrupts

- Interrupt requests may occur while the system is processing another interrupt
- All interrupts are not equally urgent (as it is also in real life
  - Some are more urgent than other
  - *Also true in real life*

Lecture Notes from Dr. Paris, University of Houston

Example in Real life

- Let us try to prioritize
  - Phone is ringing
  - Washer signals end of cycle
  - Dark smoke is coming out of the kitchen
  - …
- With vectorized interrupts, a phone call will never interrupt another phone call

The Solution

Smoke in the kitchen
Phone is ringing
End of washer cycle
More low-priority stuff

Brief History of Operating Systems Development (cont'd.)

- **1960s: third generation**
  - Faster CPUs
  - Speed caused problems with slower I/O devices
  - Multiprogramming
    - Allowed loading many programs at one time
    - Passive multiprogramming: interrupts
    - Active multiprogramming: time slicing
  - Program scheduling
    - Initiated with second-generation systems
    - Continues today
  - Few advances in data management
  - system customization
    - ds

Brief History of Operating Systems Development (cont'd.)

- 1970s
  - Faster CPUs
  - Speed caused problems with slower I/O devices
  - Main memory physical capacity limitations
    - Multiprogramming schemes used to increase CPU usage
    - Virtual memory developed to solve physical limitation
  - Database management software
    - Became a popular tool
  - A number of query systems introduced
  - Programs started using English-like words, modular structures, and standard operations

Multiprogramming

- With multiprogramming, a computer lets its CPU divide its time among different tasks: the CPU works for, say, one tenth of a second on a given program, then for another tenth of a second on another one and so forth.
- Note that a single-core CPU is only working on *one single task* at any given time.
- The major direct benefit of multiprogramming is that the CPU does not waste any time waiting for the completion of an I/O because it can use the free time to work on another task.

Brief History of Operating Systems Development (cont'd.)

- 1980s
  - **Cost/performance ratio** improvement of computer components
  - More flexible hardware (upgrades made possible)
  - **Multiprocessing**
    - Allowed parallel program execution
  - Evolution of personal computers
  - Evolution of high-speed communications
  - **Distributed processing** and **networked systems** introduced

Multiprocessor Systems

- These systems are designed for multiprocessor architectures
- Two major approaches:
- ***master-slave*** system: all system functions are performed by *one* processor; the other processors can only execute user programs;
- ***symmetric*** system: any processor can perform all functions; there can be multiple copies of the OS running in parallel and we must prevent them from interfering with

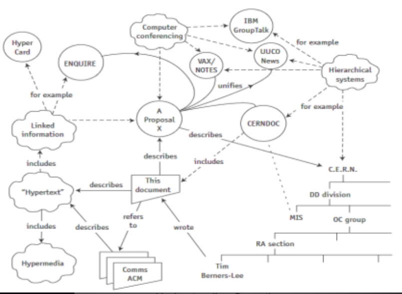
Brief History of Operating Systems Development (cont'd.)

- 1990s
  - Demand for Internet capability
    - Sparked proliferation of networking capability
    - Increased networking
    - Increased tighter security demands to protect hardware and software
  - Multimedia applications
    - Demanding additional power, flexibility, and device compatibility for most operating systems

Brief History of Operating Systems Development (cont'd.)

(figure 1.12)

Illustration from the first page of the 1989 proposal by Tim Berners-Lee describing his revolutionary "linked information system." Based on this research, he designed the first World Wide Web server and browser, making it available to the general public in 1991.



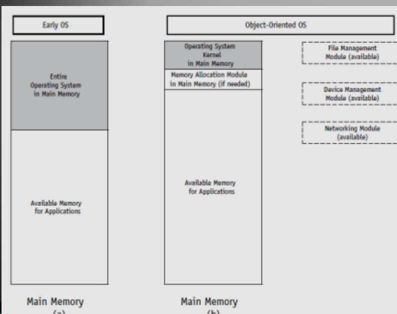
Brief History of Operating Systems Development (cont'd.)

- 2000s
  - Primary design features support:
    - Multimedia applications
    - Internet and Web access
    - Client/server computing
  - Computer systems requirements
    - Increased CPU speed
    - High-speed network attachments
    - Increased number and variety of storage devices
  - Virtualization
    - Single server supports different operating systems

Object-Oriented Design

- Driving force in system architecture improvements
  - **Kernel** (operating system nucleus)
    - Resides in memory at all times, performs essential tasks, and protected by hardware
  - **Kernel reorganization**
    - Memory resident: process scheduling and memory allocation
    - Modules: all other functions
  - Advantages
    - Modification and customization without disrupting integrity of the remainder of the system
    - Software development more productive

Object-Oriented Design (cont'd.)



(figure 1.15)

Early operating systems (a) loaded in their entirety into main memory. Object-oriented operating systems (b) load only the critical elements into main memory and call other objects as needed.

Summary

- Operating system overview
- Functions of OS
  - Manages computer system
    - Hardware and software
  - Four essential managers
    - Work closely with the other managers and perform unique role
  - Network Manager
    - Operating systems with networking capability
  - Essential hardware components
    - Memory chips, I/O, storage devices, and CPU

Summary (cont'd.)

- Evolution of OSs
  - Run increasingly complex computers
  - Run increasingly complex computer systems
  - Prior to mid-1970s
    - Computers classified by capacity and price
  - Dramatic changes over time
    - Moore's Law: computing power rises exponentially
    - Physical size, cost, and memory capacity
- Mobile society information delivery
  - Creates strong market for handheld devices
  - Integral in modern computer systems

Summary (cont'd.)

- Five categories of operating systems
  - Batch, interactive, real-time, hybrid, and embedded
- Use of object-oriented design improves the system architecture
- Several ways to perform OS tasks
- Designer determines policies to match system's environment
- Next:
  - Explore details of operating system components

References

- Understanding Operating Systems, Sixth Edition
- J. F. Paris, University of Houston Computer Science, Lecture Notes.