CSC 4103 - Operating Systems
Fall 2009

LECTURE - I

INTRODUCTION

Tevfik Koşar

Louisiana State University
August 25th, 2009

Contact Information

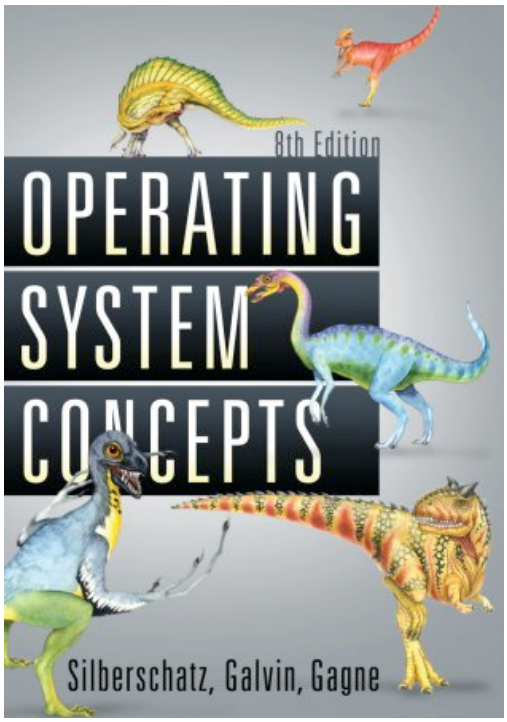
- Instructor: Prof. Tevfik Kosar
  - Office: 292 Coates (also 333 Johnston)
  - Phone: 578-9483
  - Email: kosar@csc.lsu.edu
  - Web: http://www.cct.lsu.edu/~kosar
  - Office hours: Wed & Thu, 2:00pm - 3:00pm
    (Or anytime by appointment)
- Teaching Assistant: Qing Huag (qhuang6@lsu.edu)

Logistics

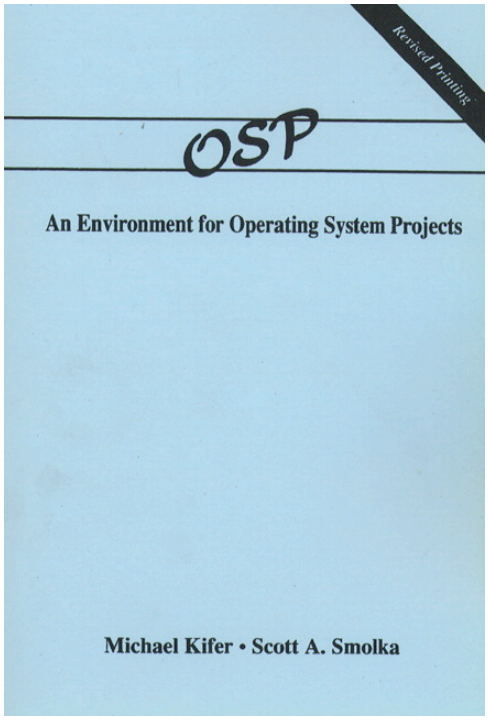
- Course web page: http://www.cct.lsu.edu/~kosar/csc4103
  - All lecture notes will be available online
  - As well as homework assignments, projects and other important course information

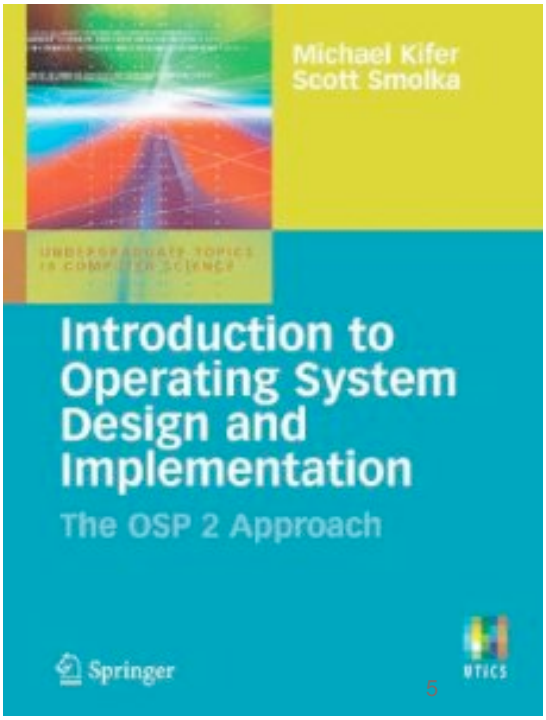
- Course mailing list: csc4103-fall09@cct.lsu.edu
  - Important course announcements including projects, homework assignments, and exams will be sent to this mailing list
  - Provide me with your active email address to be added to the class mailing list

Textbooks 1: Main Text



Textbook 2: For Project (One of following)





Grading

- The end-of-semester grades will be composed of:

– Pop Quizzes	: 10%	(4-5)
– Homework	: 15%	(5)
– Projects	: 20%	(2)
– Midterm	: 25%	(1)
– Final	: 30%	(1)

You are expected to attend the classes and actively contribute via asking and/or answering questions.

Passive vs Active Learning

Passive learning: learning through reading, hearing & seeing

Active learning: learning through saying and doing

After 2 weeks, we tend to remember:

Passive learning

- 10% of what we read
- 20% of what we hear
- 30% of what we see (i.e. pictures)
- 50% of what we hear and see

Active learning

- 70% of what we say
- 90% of what we say and do

How to Become an Active Learner

- Recall prior materials
- Answer a question
- Guess the solution first (even guessing wrong will help you to remember the right approach)
- Work out the next step before you have to read on
- Think of an application
- Imagine that you were the professor and think about how you would give a test on the subject material so that key concepts and results will be checked.
- Summarize a lecture, a set of home work or a lab in your own words concisely.

Rules

- No late homework/project submissions accepted!
- Exams will be closed book.
- You are only responsible from material covered in the class, homework, and projects.
- Academic dishonesty will be treated seriously.

What Expect to Learn?

- Key Concepts of Operating Systems
  - Design, Implementation, and Optimization
- Topics will include:
  - Processes, Threads and Concurrency
  - CPU and I/O Scheduling
  - Memory and Storage Management
  - File System Structures
  - Synchronization and Deadlocks
  - Protection and Security
  - Distributed Computing & Related Issues

Introduction

What is an Operating System?

- A program that manages the computer hardware.
- An intermediary between the computer user and the computer hardware.
- Manages hardware and software resources of a computer.

Computer System Overview

- **A computer system consists of (bottom-up):**

1. hardware
2. firmware (BIOS)
3. operating system
4. system programs
5. application programs
6. users

Computer System Overview

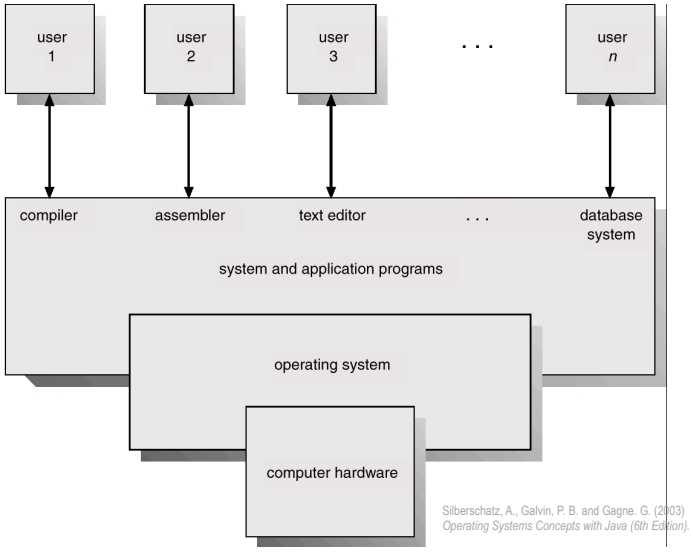
1. Hardware
   - ✓ provides basic computing resources
   - ✓ CPU, memory, disk, other I/O devices
2. Firmware (BIOS)
   - ✓ software permanently stored on chip (but upgradable)
   - ✓ loads the operating system during boot
3. Operating system
   - ✓ controls and coordinates the use of the hardware among the various application programs for the various users

Computer System Overview

4. System programs
   - ✓ basic development tools (shells, compilers, editors, etc.)
   - ✓ not strictly part of the core of the operating system
5. Application programs
   - ✓ define the logic in which the system resources are used to solve the computing problems of the users
   - ✓ database systems, video games, business programs, etc.
6. Users
   - ✓ people, other computers, machines, etc.

Role of an Operating System

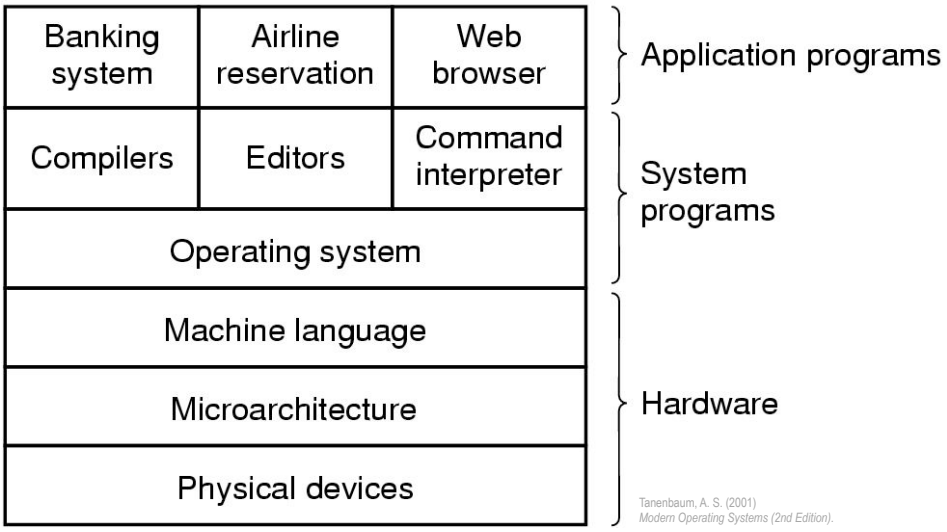
- The Silberschatz "pyramid" view



Abstract view of the components of a computer system

Role of an Operating System

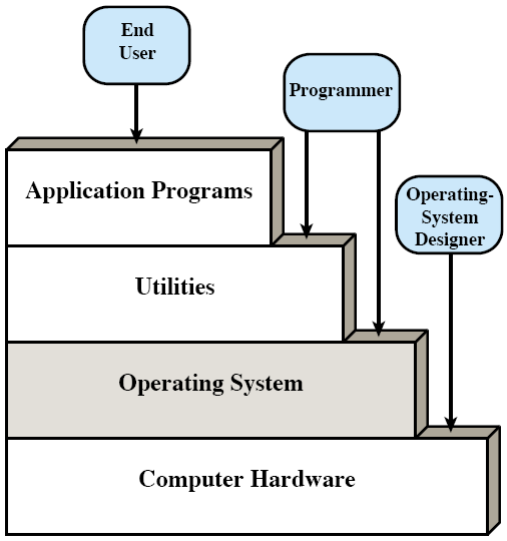
- **The Tanenbaum "layered" view**



A computer system consists of hardware, system programs and application programs

Role of an Operating System

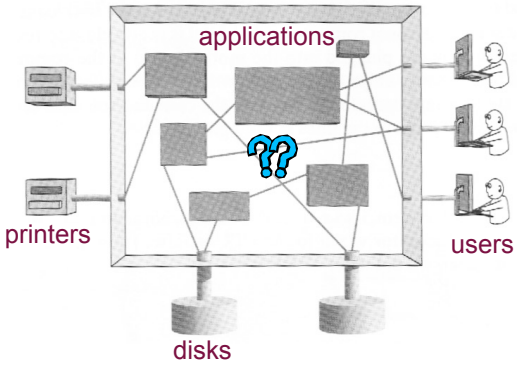
- **The Stallings "layered & stairs" view**



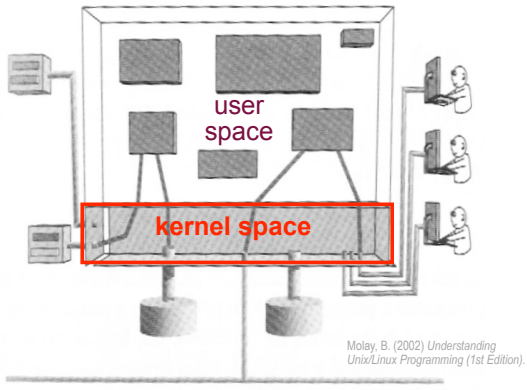
Layers and views of a computer system

Role of an Operating System

- **The Molay "aquarium" view**
  - the only not-layered view
  - everything must transit through the O/S or "kernel"



How are they all connected?



Molay, B. (2002) *Understanding Unix/Linux Programming (1st Edition).*

The kernel manages all connections

Key Point

- An operating system is a program that acts as an intermediary between users/applications and the computer hardware.

Operating System Goals

- From the user perspective:
  - Executes user programs and make solving user problems easier
  - Makes the computer system convenient to use
    - hides the messy details which must be performed
    - presents user with a virtual machine easier to use
- From the System/HW Perspective:
  - Manages the resources
  - Uses the computer hardware in an efficient manner
    - time sharing: each program gets some time to use a resource
    - resource sharing: each program gets a portion of a resource

OS Services for Users

- Program Execution
  - The OS loads programs and data into memory, initializes I/O devices and files, schedules the execution of programs
- Access to I/O Devices
  - The OS hides I/O device details from applications (direct I/O access is forbidden) and offers a simplified I/O interface
- Controlled Access to Files & Directories
  - The OS organizes data into files and directories, controls access to them (i.e. create, delete, read, write) and preserves their integrity

OS Services for Users

- Communications
  - The OS allows exchange of information between processes, which are possibly executing on different computers
- Error Detection and Response
  - The OS properly handles HW failures and SW errors with the least impact to running applications (i.e. terminating, retrying, or reporting)

OS Services for System/HW

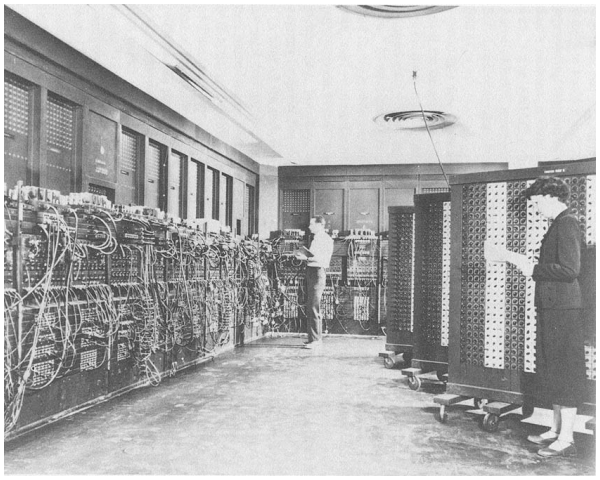
- Resource Allocation
  - The OS allocates resources to multiple users and multiple jobs running at the same time
- Operation Control
  - The OS controls the execution of user programs and operations of I/O devices
- System Access
  - The OS ensures that all access to resources is protected, including authorization, conflict resolution etc.
- Accounting and Usage Statistics
  - The OS keeps performance monitoring data

Evolution of Computer Systems

1. **Serial processing**
2. **Simple batch systems**
3. **Multiprogrammed batch systems**
4. **Personal computers**

Serial Processing Systems

- **First generation: 1945–55**
  - ✓ room full of cabinets: mechanical relays, then vacuum tubes



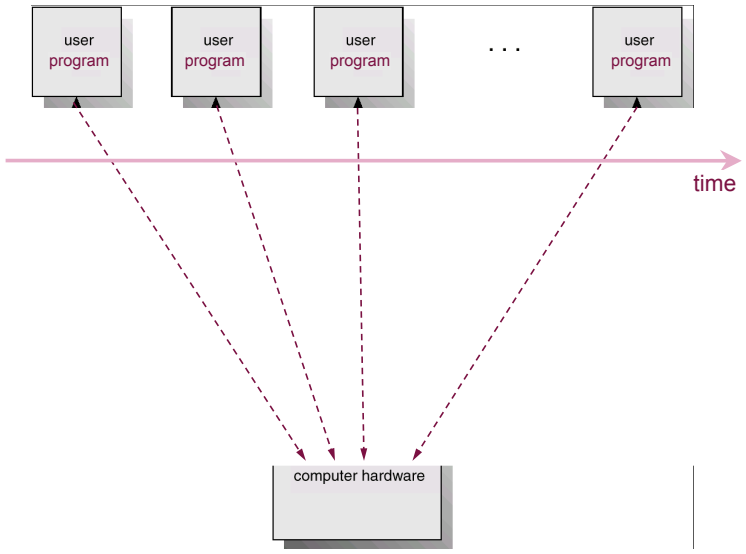
The ENIAC (Electronic Numerical Integrator And Computer)

Serial Processing Systems

- **Human operator-programmer-user**
  - ✓ the machine was run from a console that had display lights, toggle switches, a plugboard or punched cards, a printer
  - ✓ the programmer also "operated" the machine as she/he interacted directly with the bare hardware
  - ✓ at first the computer was programmed by physically re-wiring it; later, through stored programs ("von Neumann architecture")
- **Operating systems were unheard of**
  - ✓ programs were entirely written in machine or assembly language
  - ✓ one running program had complete control of the entire computer

Serial Processing Systems

- Programs directly access the hardware, one at a time



Serial Processing Systems

- Problem 1: scheduling
  - ✓ users had access to the computer one by one in series
  - ✓ machine time was reserved in blocks of half hours with a hard-copy sign-up sheet
  - ✓ either the user was finished early and computer processing time was wasted
  - ✓ or, more frequently, the user could not finish debugging her/his program during the allotted time
- Problem 2: duplication of programming efforts
  - ✓ user wrote again and again the same routines (ex: I/O devices)
  - ✓ no concept of libraries

Simple Batch Systems

- Second generation: 1955–65
  - ✓ advent of transistors and printed circuits

http://www.columbia.edu/acis/history/1965.html **The IBM 7094 at Columbia University**

Simple Batch Systems

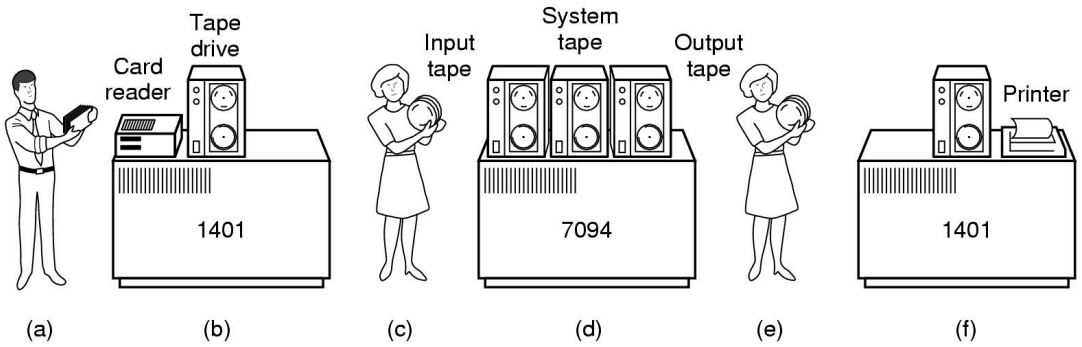
- **Separation between operators and programmers**
  - ✓ first commercially viable machines
  - ✓ the programmer prepares her/his job off-line on punched cards, brings the card deck to the machine room and waits for results
  - ✓ the human operator runs the job and delivers a printed output

- **New problem: still basically serial processing**
  - ✓ one single **job** at a time
  - ✓ huge setup time for each job: loading the compiler, the source program, saving the compiled program, loading, linking, etc.
  - ✓ also mounting and dismounting tapes, handling card decks, etc.
  - ✓ a lot of time was wasted manipulating things and walking around

Simple Batch Systems

- **Solution: batch the jobs together**
  1. the human operator pre-reads a tray full of jobs onto a magnetic tape
  2. the human operator loads a special program, the **monitor**, that will automatically read the jobs from the tape and run them sequentially
  3. the effect of the monitor program is to write the output of each job on a second magnetic tape
  4. finally, the human operator brings the full output tape for offline printing

Simple Batch Systems



An early IBM batch system

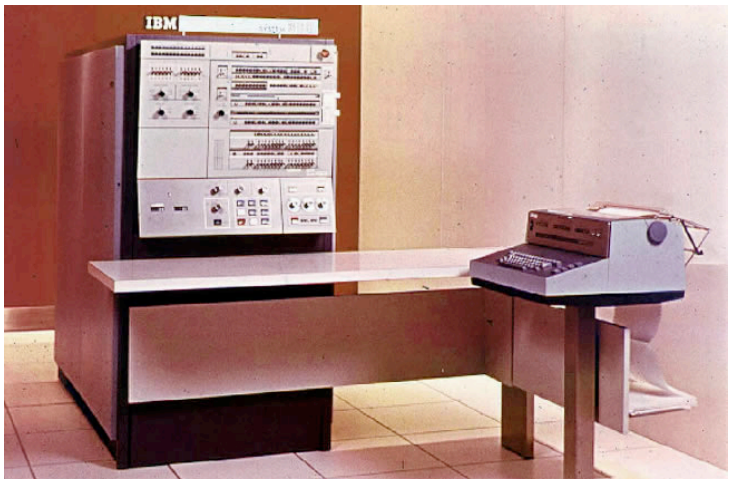
a) programmer brings cards to IBM 1401
b) 1401 reads batch of jobs onto tape
c) operator carries input tape to IBM 7094
d) 7094 does computing
e) operator carries output tape to 1401
f) 1401 prints output

Simple Batch Systems

- **The monitor program automates some of the human operator's tasks and is the ancestor of modern O/S**
  - ✓ the monitor is a special program that controls the sequence of events
  - ✓ it always resides in main memory
  - ✓ it reads in jobs one at a time, places a job in the user program area of the memory, and passes control to it
  - ✓ upon completion, the user program branches back to the monitor, which immediately loads and executes the next job
  - ✓ therefore, the CPU alternates between fetching/executing instructions from the monitor program and fetching/executing instructions from the user program

Multi-Programmed Batch Systems

- Third generation: 1965-80
  - ✓ first major use of small-scale Integrated Circuits (ICs)



http://www.thocp.net/hardware/pictures/

The IBM 360

Multi-Programmed Batch Systems

- Problem: despite batching, a lot of CPU time is still wasted waiting for I/O instructions to complete
  - ✓ I/O devices much slower than processor, e.g. tapes!

Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
TOTAL	31 μs

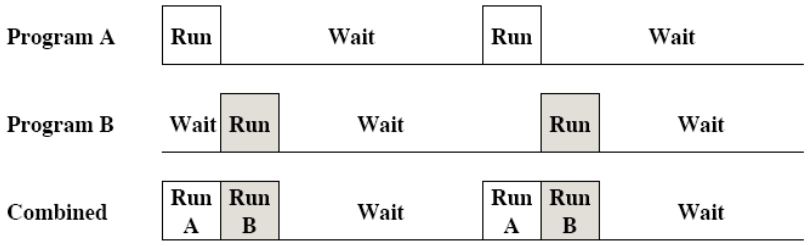
Percent CPU Utilization $= \frac{1}{31} = 0.032 = 3.2\%$

Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Example of system utilization with uniprogramming

Multi-Programmed Batch Systems

- Solution: load two jobs in memory
  - ✓ while one job is waiting for I/O, the processor could switch to the other job

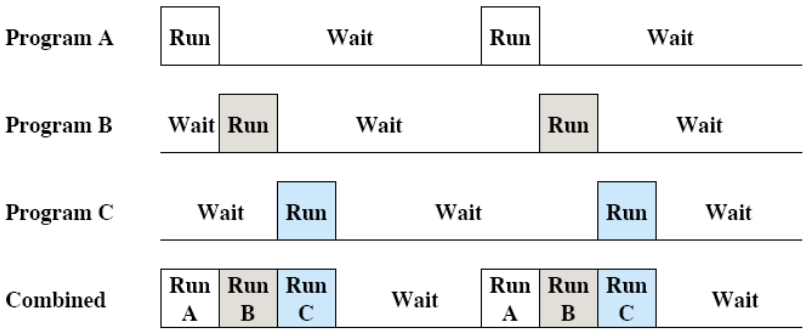


Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Multiprogramming with two programs

Multi-Programmed Batch Systems

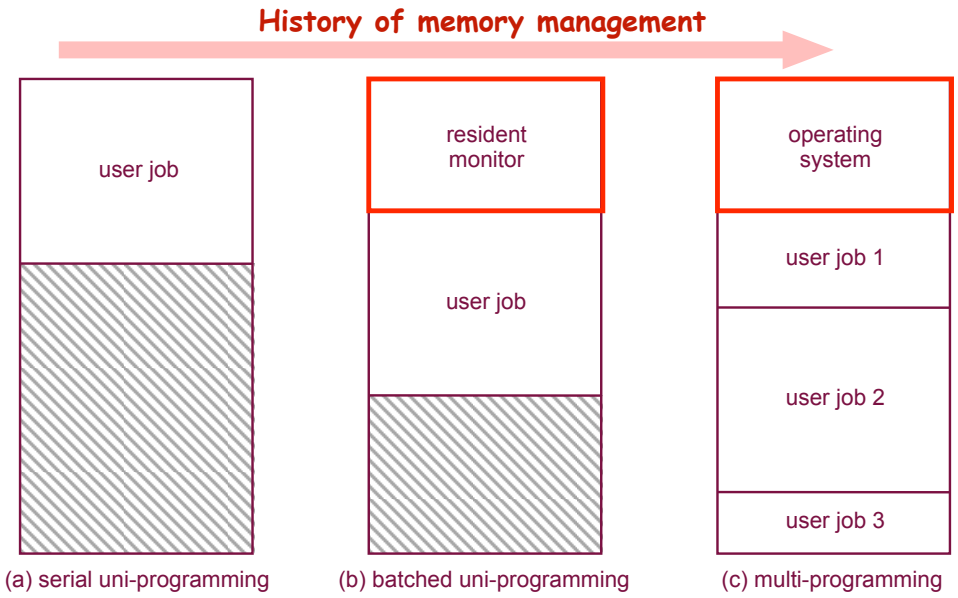
- Expand to three, four or more jobs
  - ✓ jobs are kept in main memory at the same time and the CPU is multiplexed among them, or "multi-programmed"
  - ✓ **Multi-programming** ("multitasking") is a central O/S theme



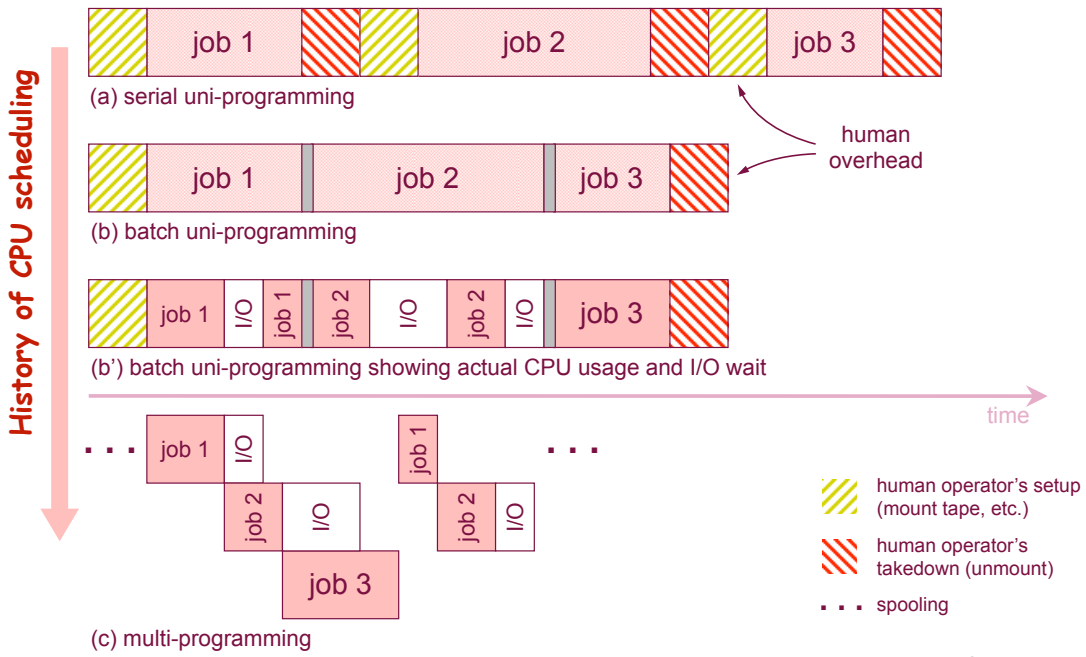
Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Multiprogramming with three programs

Multi-Programmed Batch Systems



Multi-Programmed Batch Systems



Personal Computers

- **Fourth generation: 1980-Present**
  - ✓ Large Scale Integration (LSI) makes personal computing real

Personal Computers

- **From multiple users back to a single user**
  - preemptive multitasking was developed in the 1960's to share big and costly mainframe computers among multiple users
  - since then, single-user interactive computing has become possible on dedicated personal computers (PCs)

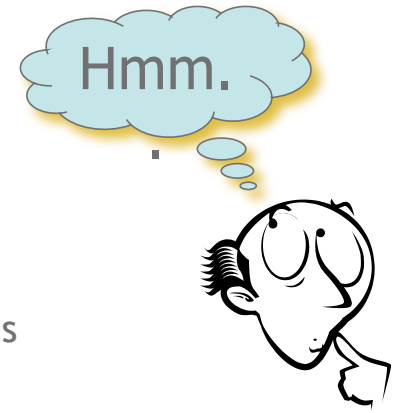
- **Resource sharing not critical anymore, yet multitasking still a central feature of modern PC operating systems**
  - a single-tasking environment is tedious: one must close the drawing application before opening the word processor, etc.
  - multitasking makes it possible for a single user to run multiple applications at the same time (or "background" processes) while retaining control of the computer

Personal Computers

- Other mainframe system features have been integrated into PC systems, for example: file protection
  - ✓ in multi-user systems, file protection was critical
  - ✓ in single-user PCs, it was not considered necessary at first, but reappeared with the advent of networking
- PC systems emphasize user convenience
  - ✓ the primary goal of the mainframe multiprogrammed systems was to maximize CPU utilization
  - ✓ as in time-sharing systems, the primary goal of PC systems is rather to maximize user convenience and responsiveness

Summary

- What is an OS?
- Role of an OS
- Operating System Goals
  - User View vs System View
- Operating System Services
  - For Users and HW
- Evolution of Computing Systems



- Reading Assignment: Chapter 1 from Silberschatz.

Acknowledgements

- “Operating Systems Concepts” book and supplementary material by A. Silberschatz, P. Galvin and G. Gagne
- “Operating Systems: Internals and Design Principles” book and supplementary material by W. Stallings
- “Modern Operating Systems” book and supplementary material by A. Tanenbaum
- R. Doursat and M. Yuksel from UNR