CSE 421/521 - Operating Systems
Fall 2013

LECTURE - I

INTRODUCTION

Tevfik Koşar

University at Buffalo
August 27th, 2013

Contact Information

- Instructor: Prof. Tevfik Kosar
  - Office: 338J Davis Hall
  - Phone: 645-2323
  - Email: tkosar@buffalo.edu
  - Web: http://www.cse.buffalo.edu/~tkosar
  - Office hours: Wed 1:00pm - 2:00pm, Thu 1:00pm - 2:00pm
    (Or anytime by appointment)
- Teaching Assistants:
  - Sonali Batra <sonaliba@buffalo.edu>
    - Office hours: Mon & Fri 11:00am-12:00pm (Davis 302)
  - Fatih Bulut <mbulut@buffalo.edu>
    - Office hours: Tue & Thu 10:00am - 11:00am (Davis 302)
  - Kyungho Jeon <kyunghoj@buffalo.edu>
    - Office hours: Mon & Wed 12:00pm - 1:00pm (Davis 302)

Recitations

- The undergrads need to attend one of the following recitations:
  - Tue 11:00am - 11:50am (Bell 138)
  - Wed 10:00am - 10:50am (Bell 337)

- Recitations will include:
  - Clarification of some important course material
  - Solutions of quiz, HW, and other exercise questions
  - Project guidance
  - Programming tips

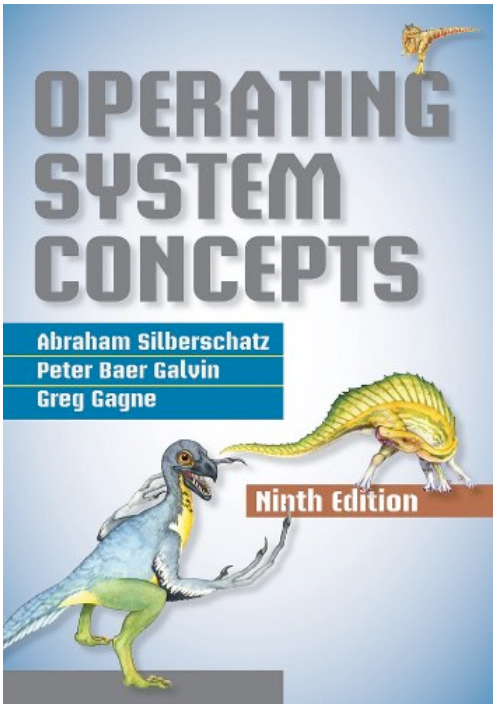
- PS: undergrads only, no grads allowed in recitations!

Course Web Page

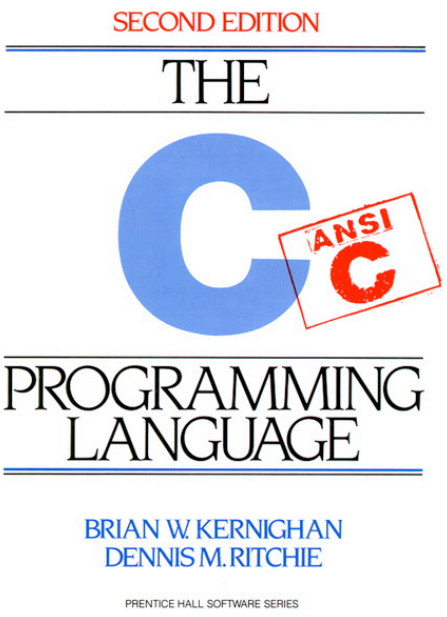
- Course web page:
  - http://www.cse.buffalo.edu/faculty/tkosar/cse421-521/
  - All lecture notes will be available online
  - As well as homework assignments, projects and other important course information

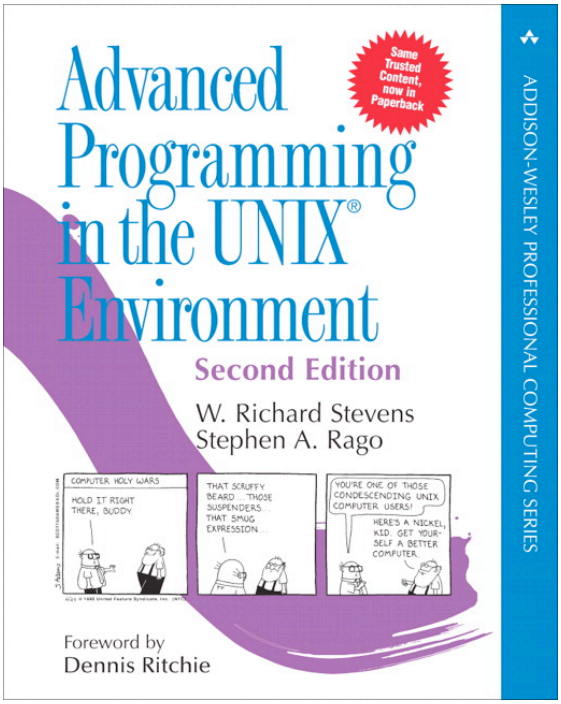
Date	Lect.	Title	Notes
Aug 27	1	Introduction	*Read Ch.1*
Aug 29	2	Operating System Structures	
Sep 3	3	Processes	
Sep 5			Rosh Hashanah
Sep 10	4	Threads	
Sep 12	5	CPU Scheduling – I	
Sep 17	6	CPU Scheduling – II	
Sep 19	7	Project-I Discussion	*Project-I out*
Sep 24	8	Process Synchronization – I	
Sep 26	9	Process Synchronization – II	
Oct 1	10	Deadlocks – I	

Textbook: Required

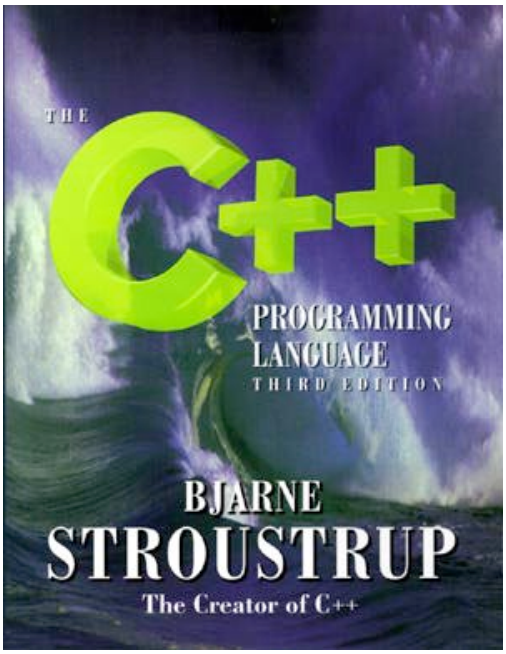


Recommended Supplementary Text





Recommended Supplementary Text



Grade Components

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

- Pop Quizzes	: 5%	(5; 4 counted)
- Projects	: 40%	(2)
- Midterm	: 25%	(1)
- Final	: 30%	(1)

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

Grading Scale

- Final grades will be given according to this scale -->

* I will use "curve" to adjust grades (up) to this scale.

* There will be separate curves for graduate & undergraduate students!

90-100: A
85-89.9: A-
- 80-84.9: B+
- 75-79.9: B
- 70-74.9: B-
- 65-69.9: C+
- 60-64.9: C
- 55-59.9: C-
- 50-54.9: D+
- 40-49.9: D
- 0 - 39.9: F

Rules

- No use of laptops/phones during the lectures!
- No late 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 "very" seriously!

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 homework or a lab in your own words concisely.

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?

- It is a program
- It is a big hairy program
  - The Linux source code has more than 1.7 M lines of C code
- 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

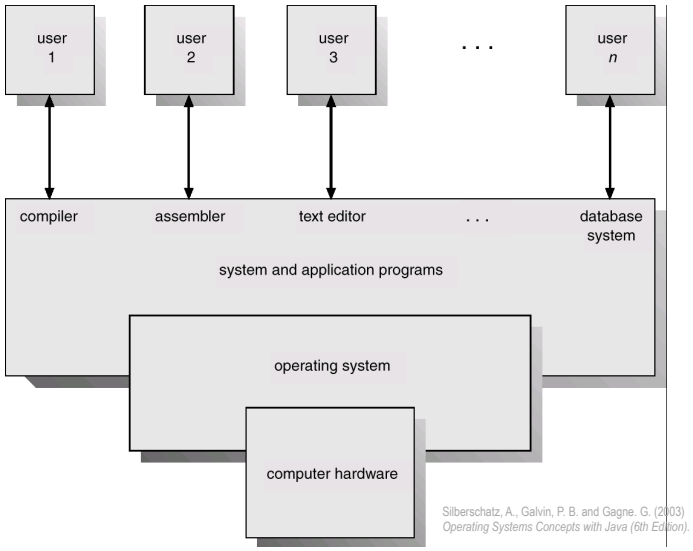
1. Hardware
   - ✓ provides basic computing resources
   - ✓ CPU, memory, disk, other I/O devices
2. Firmware (BIOS)
   - ✓ software permanently stored on chip (but upgradable)
   - ✓ loads the operating system during boot
3. Operating system
   - ✓ controls and coordinates the use of the hardware among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

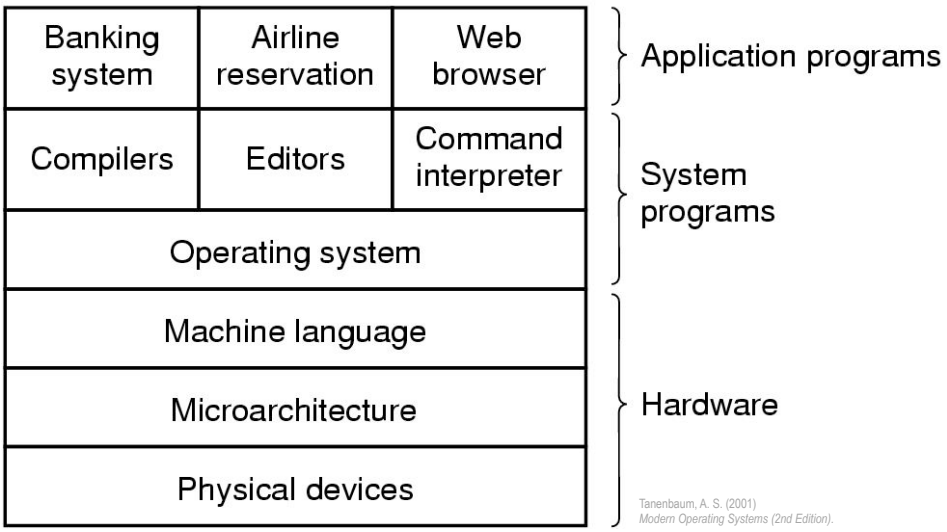
- **The Silberschatz "pyramid" view**



Abstract view of the components of a computer system

Role of an Operating System

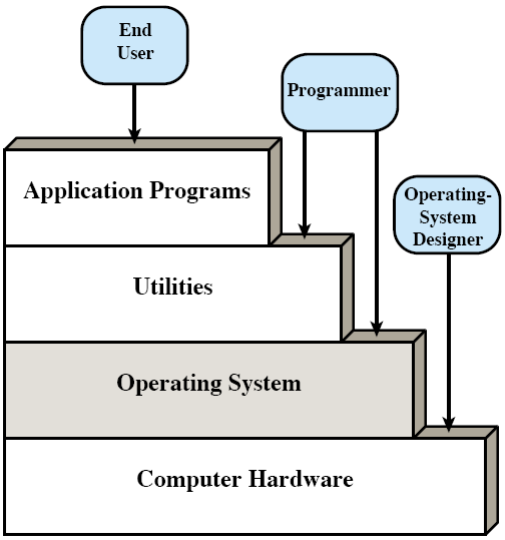
- **The Tanenbaum "layered" view**



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

Role of an Operating System

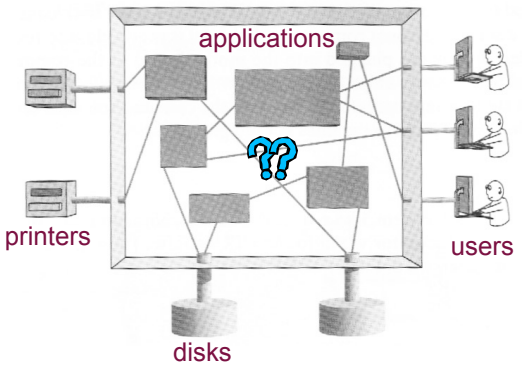
- **The Stallings "layered & stairs" view**



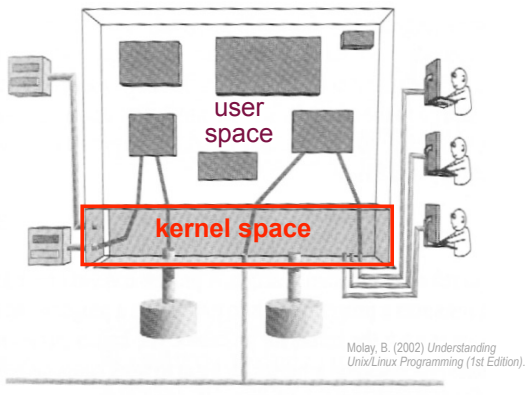
Layers and views of a computer system

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?



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

The Major OS Issues

- **structure**: how is the OS organized?
- **sharing**: how are resources shared across users?
- **naming**: how are resources named (by users or programs)?
- **security**: how is the integrity of the OS and its resources ensured?
- **protection**: how is one user/program protected from another?
- **performance**: how do we make it all go fast?
- **reliability**: what happens if something goes wrong (either with hardware or with a program)?
- **extensibility**: can we add new features?
- **communication**: how do programs exchange information, including across a network?

More OS Issues..

- **concurrency**: how are parallel activities (computation and I/O) created and controlled?
- **scale**: what happens as demands or resources increase?
- **persistence**: how do you make data last longer than program executions?
- **distribution**: how do multiple computers interact with each other?
- **accounting**: how do we keep track of resource usage, and perhaps charge for it?

There are tradeoffs, not right and wrong!

Summary

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

- Reading Assignment: Chapters 1 & 2 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, Ed Lazowska from UWashington