OS

Operating Systems

Introduction

IIS & CS
Tsinghua University

Acknowledgement:
materials from Dr. Zhang Yong Guang in MSRA,
And from http://williamstallings.com/OS/OS5e.html , http://www.os-book.com

Outline

- Course Introduction
  - Contact Information
  - Why study OS?
  - Purpose of This Course
  - Reference Textbooks
  - Course Overview
  - Course Scheduling
  - Grading & Prerequisites
  - Words to Remember
- What is an Operating System?
- Evolution of Operating Systems
- Operating-System Structures

OS

Contact Information

Instructor:	CHEN, Yu 陈渝
Office:	Room 3-124/106, FIT Building
Email:	yuchen@tsinghua.edu.cn , chyyuu@gmail.com
Wiki :	http://os.cs.tsinghua.edu.cn/oscourse/OS2014
TA:	茅俊杰 eternal.n08@gmail.com
maillist:	https://piazza.com/tsinghua.edu.cn/spring2014/30240243/home https://groups.google.com/group/oscourse?hl=en

Contact Information

Instructor:	Yong XIANG （向勇）
Office:	Room 410， Section 9， East Main Building
Tel&email:	6278 5609， xyongcn@gmail.com
Wiki:	http://os.cs.tsinghua.edu.cn/oscourse/OS2014
TA:	曹睿东 crdfrank@gmail.com
maillist:	https://piazza.com/tsinghua.edu.cn/spring2014/30240243/home https://groups.google.com/group/oscourse?hl=en

Why Study OS?

The Operating System (OS) I use has already been written, and I doubt it will be my job to write another one. For example, Windows, Linux. Haven’t OS developers figured everything out already? What more is there to do?

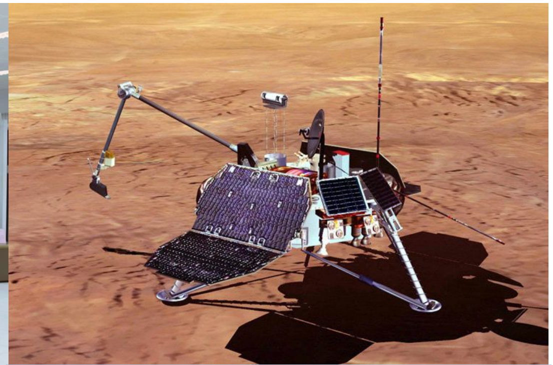
Why should I study this as an undergraduate?

OS is cool!

OS is important!

OS is challenging!

I want to be involved!



OS is important

OS: A Corner Stone of Computer Science Research

- Fundamental understanding of computer systems
- Driven by hardware advance and scale
- Advances in both academic and industry

Where are the Research Effects

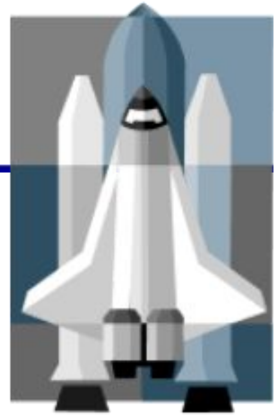
- CS departments in Top universities
- Computer industry
  - Old time: Xerox (PARC), IBM, DEC (SRC), Bell Labs
  - Now: Microsoft, Google, Yahoo, IBM, HP, Sun, Intel, VMware, Amazon, …
- Research Associations
  - ACM SIGOPS
  - USENIX

OS is important

Top Conferences on Operating System Research

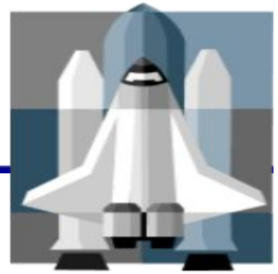
- ACM Symposium on Operating Systems Principles (SOSP)
  - ACM SIGOPS
  - Every two years (odd number: 1967-)
  - ~20 papers
- USENIX Symposium on Operating Systems Design and Implementation (OSDI)
  - USENIX
  - Every two years (even number: 1994-)
  - ~20 papers

OS is Challenging



- The OS is really large
  - Windows XP is 45 million lines
- The OS manages concurrency
  - Concurrency leads to interesting programming challenges
- OS code manages raw hardware
  - Timing dependent behavior, undocumented behavior, HW bugs
- OS code must be efficient, low CPU, memory, disk use
- OS fails = machine fails
  - OS must fail less than user programs
- OS basis of system security

OS is Challenging

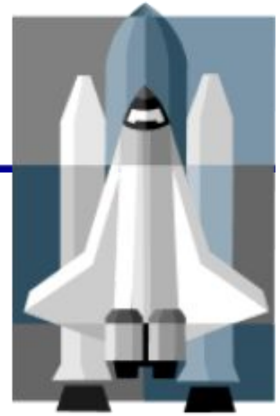


OS is not about concurrency & trivial scheduling algorithms

- concurrency is a small part
- disk scheduling is mostly irrelevant (SCSI does it for you)
- process scheduling is a small topic

- monitors and philosophers don't live in OS kernels
- the locking problems there are in kernels require too much background

OS is Challenging



OS is about: **<u>tradeoffs</u>**

time vs space

performance vs predictability

fairness vs performance (which design will work and why?)

OS is about: **<u>Hardware</u>**

how does interrupt/exception/context switch really work?

how does a TLB work and what does this mean for page tables?

if you aren't showing any assembler code you aren't teaching OS!

Purpose of This Course

Capstone course - combines things from many different courses

- Programming languages
- Data structures
- Algorithms
- Computer Architecture
- Computer Science

The materials

- OS concepts and principles, Source Code

The skills

- OS designs and implementations

Words to Remember for this Course

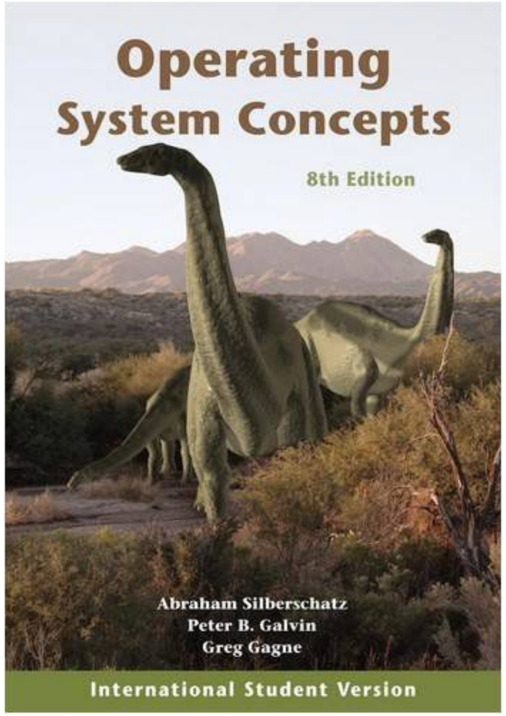
"I hear and I forget,
I see and I remember,
I do and I understand."
-- Chinese proverb

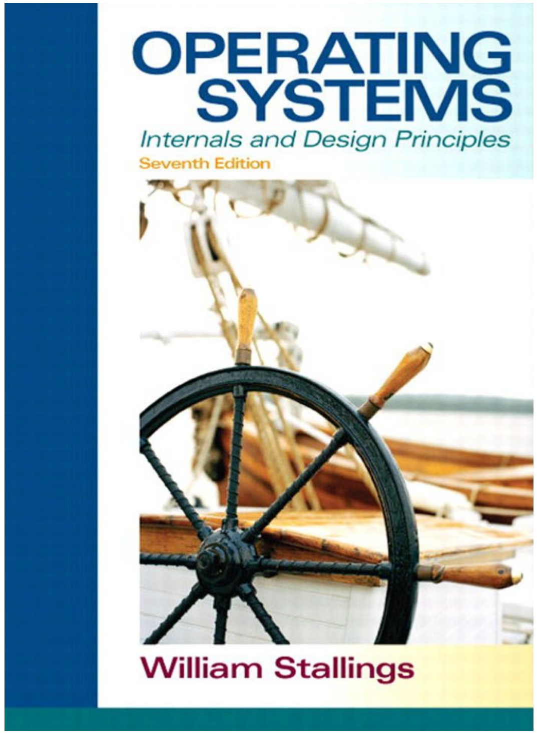
"Genius is 1% inspiration and **99%** perspiration"
-- Thomas Edison

"Hardest, best and **most fun** 3rd year course!"

Reference Textbooks

- Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, Operating system concepts (8th Edition), John Wiley & Sons, 2008
- William Stallings, Operating Systems-Internals and Design Principles(7th Edition), Prentice Hall, 2011



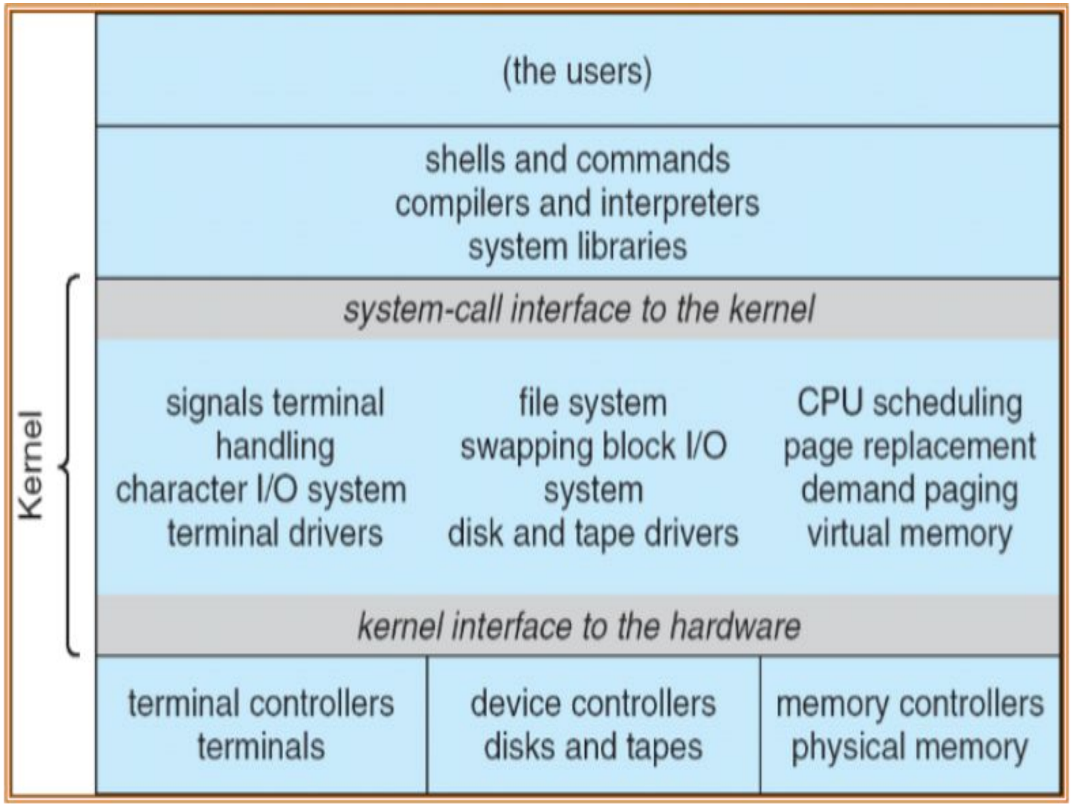


Reference Textbooks(cont.)

- Solaris Internals:Solaris 10 and OpenSolaris Kernel Architecture, 2nd Edition, Richard McDougall, Jim Mauro, Prentice Hall, July 10, 2006, ISBN 0-13-148209-2
- Microsoft Windows Internals, 4th Edition, Mark E. Russinovich, David A. Solomon, Microsoft Press, 2005， ISBN 0-7356-1917-4
- Understanding the Linux Kernel, 3rd Edition, Daniel P. Bovet, 2005, ISBN 0-5960-0565-2

Course Overview

- Basic
  - Interrupt&Syscall
  - Memory management
  - Process&Thread
  - Scheduling
  - Synchronization
  - File system
  - I/O subsystem
  - Kernel Security
- Practice
  - Do Labs/Projects
- Extension
  - Reading & Discuss Hot Topics



Course Scheduling

No.	Content
Lecture 1	Introduction+lab0
Lecture 2	Interrupt and System call+lab1
Lecture 3	physical Memory Management: partition+lab2: segmentation
Lecture 4	physical Memory Management: paging + lab2: paging
Lecture 5	Virtual Memory: page fault + lab3: page fault
Lecture 6	Virtual Memory: replacement algorithm + lab3: swap
Lecture 7	Memory virtualization
Lecture 8	Process&thread: process&thread states + lab4: kernel thread
Lecture 9	Process&thread: process&thread control + lab5: fork & exec
Lecture 10	CPU Scheduling: concept + lab6
Lecture 11	CPU Scheduling: implementation
Lecture 12	Virtualization & CPU virtualization
April 8, 2014	Mid-exam

Course Scheduling

No.	Content
Lecture 13	Synchronization
Lecture 14	Semaphore&Monitor
Lecture 15	IPC & Deadlocks + lab7
Lecture 16	File System: concept
Lecture 17	File System: implementation + lab8
Lecture 18	I/O Subsystem: concept & disk I/O
Lecture 19	I/O Subsystem: device driver (usb)
Lecture 20	I/O virtualization
Lecture 21	Kernel Security: stack overflow
Lecture 22	Kernel Security: symbol execution
May 20, 2014	Final-exam
June 10, 2014	Project report

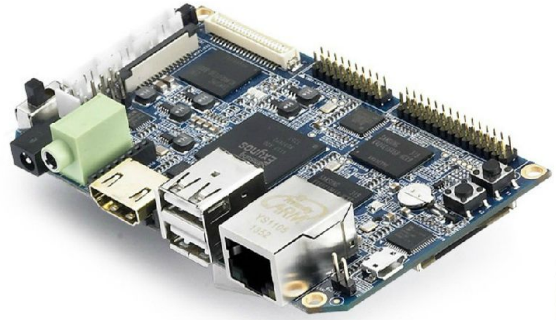
OS Course Scheduling

No.	Content
Lab 0	Preparing
Lab 1	System boot
Lab 2	Physical memory management
Lab 3	Virtual memory management
Lab 4	Process management
Lab 5	User Process Management
Lab 6	CPU Scheduling
Lab 7	Synchronization
Lab 8	File System

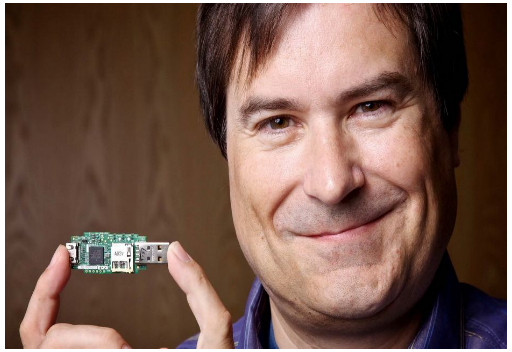
Grading & Prerequisites

- Grading
  - Labs+Homeworks: 20%
  - Middle&Final Exam: 80% (midterm 30%+Final 50%) or Course Projects
    - Principle ,Labs
- Prerequisites:
  - Computer constitution principle (Intel 80386+)
  - Data structure
  - C & ASM programming
- Course requirement
  - Keep your mobile phone in silent/vibrating alert
  - no chat during the class

Course Projects



- ucore+ modularization
  - Analysis the interfaces between kernel modules
  - Loadable kernel module
  - Loadable FAT file system
  - Loadable scheduler
  - Loadable memory manager
  - Device Driver Environment: DDEKit



- Extented projects
  - ucore+ on real board: Raspberry PI, etc.
- Other
  - Optional project related to OS

- Reference course projects

Outline

- Course Introduction
- What is an Operating System?
  - Some Operating Systems
  - Operating System Definition
- Evolution of Operating Systems
- Operating-System Structures

OS

Some Operating Systems

redhat
fedora
ubuntu
openSUSE
Mandriva
debian
gentoo linux
slackware linux
红旗
Hiweed
KNOPPIX
CentOS

Some Operating Systems

What is an Operating System?

Operating System Definition

- OS is a **control program**
  - A piece of system software
  - Controls execution of programs to prevent errors and improper use of the computer
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
- OS is a **resource allocator**
  - An interface between applications and hardware
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
  - Use the computer hardware in an efficient manner
- **No universally accepted definition**

Layers of Computer System

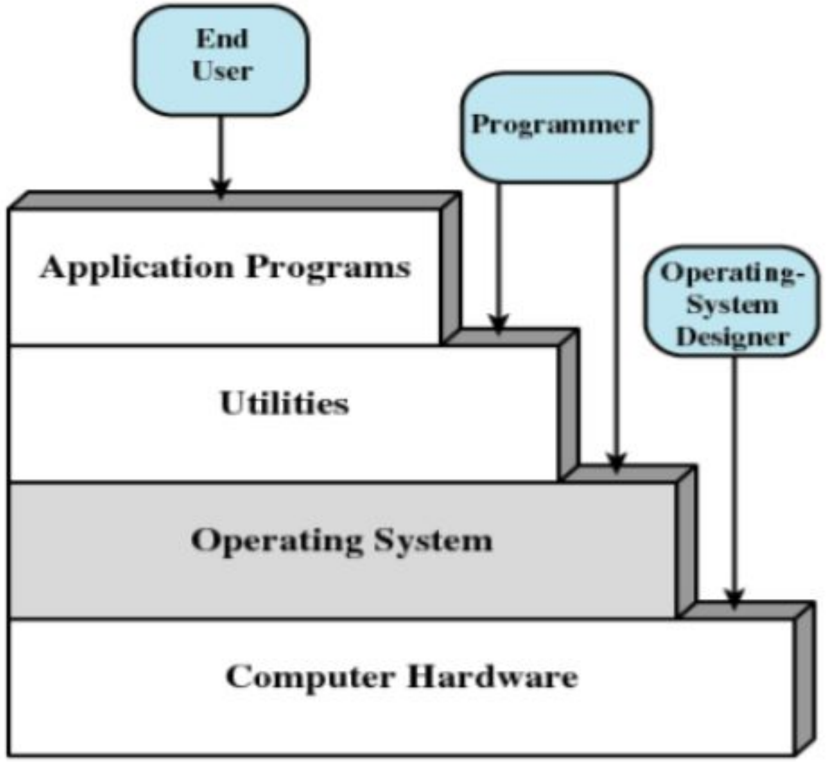
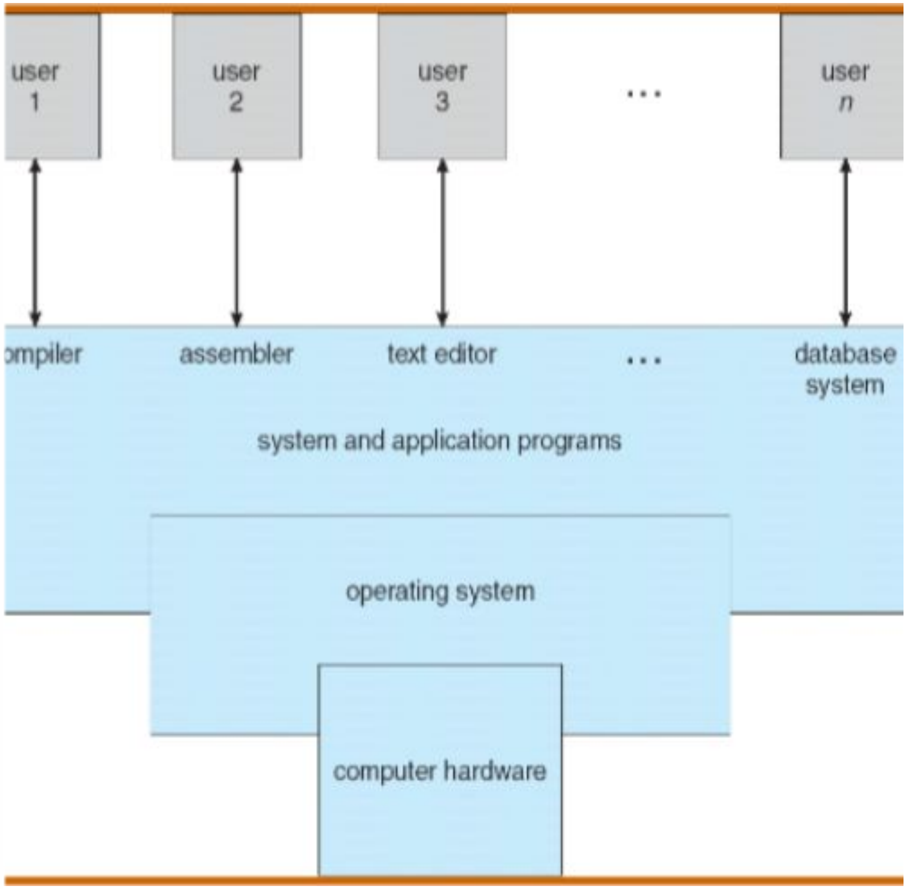


Figure 2.1 Layers and Views of a Computer System



Outline

- Course Introduction
- What is an Operating System?
- Evolution of Operating Systems
  - Single User System
  - Batch processing
  - Multiprogramming
  - Timesharing
  - OS for PCs
  - Distributed Operating Systems
- Operating-System Structures

Evolution of an Operating System

- Why do operating systems change?
  - Key functions: hardware abstraction and coordination
  - Principle: Design tradeoffs change as technology changes
  - Underlying technology has changed immensely over the past two decades !!
- Comparing computing systems from 1981 and 2014

Vital statistic	1981 IBM personal computer	2001 Dell OptiPlex GX150	2012 Dell XPS 8300
Price	$3045	$1447	$1090
CPU	4.77-MHz 8088	933-MHz Pentium III	3.4GHz Intel Core i7-2600
MIPS	0.33–1 MIPS	1,354 MIPS at 500 MHz	76,383 MIPS at 3.2 GHz
RAM	64KB	128MB	8GB DDR3 SDRAM at 1333MHz
Storage	160KB floppy drive	20GB hard drive, CD-RW and 1.44MB floppy drives	1TB - 7200RPM, SATA 3.0Gb/s

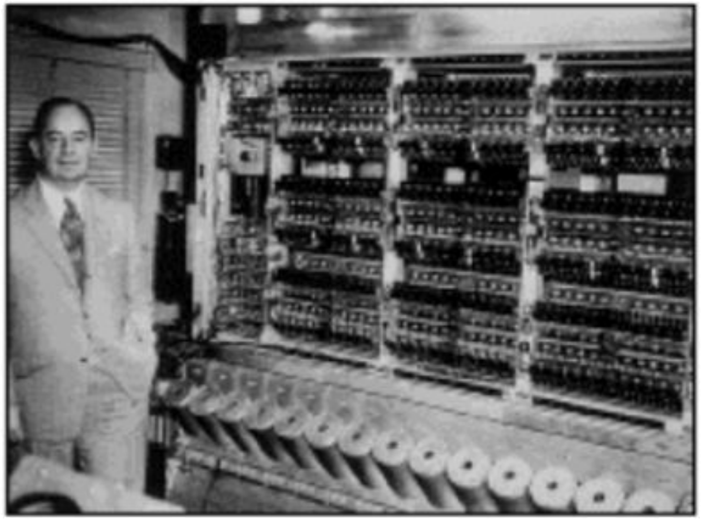
Evolution of an Operating System

- Single-user systems
- Batching systems
- Multi-programming systems
- Time sharing
- Personal computing: One system per user
- Distributed computing: lots of systems per user

Single User System (1945-1955)

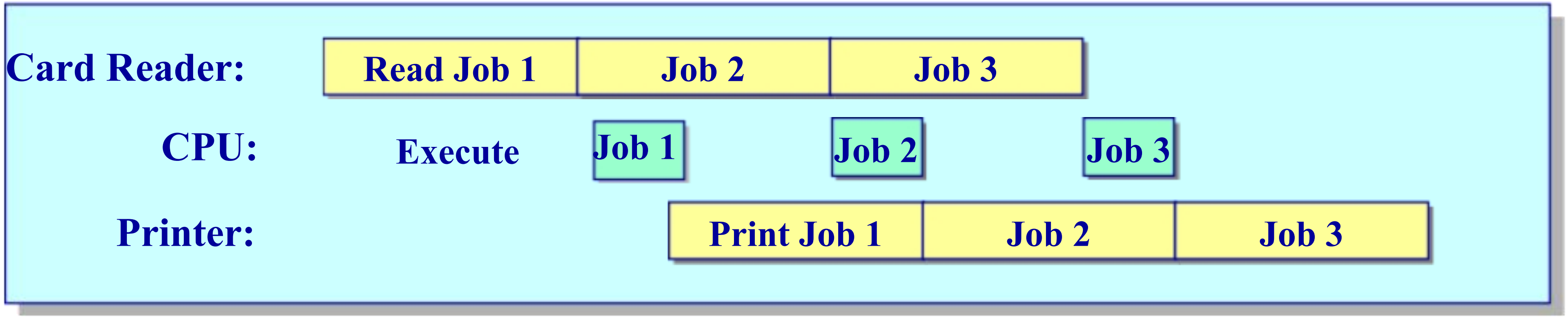
OS = *loader + libraries of common subroutines*
Problem: low *utilization* of expensive components

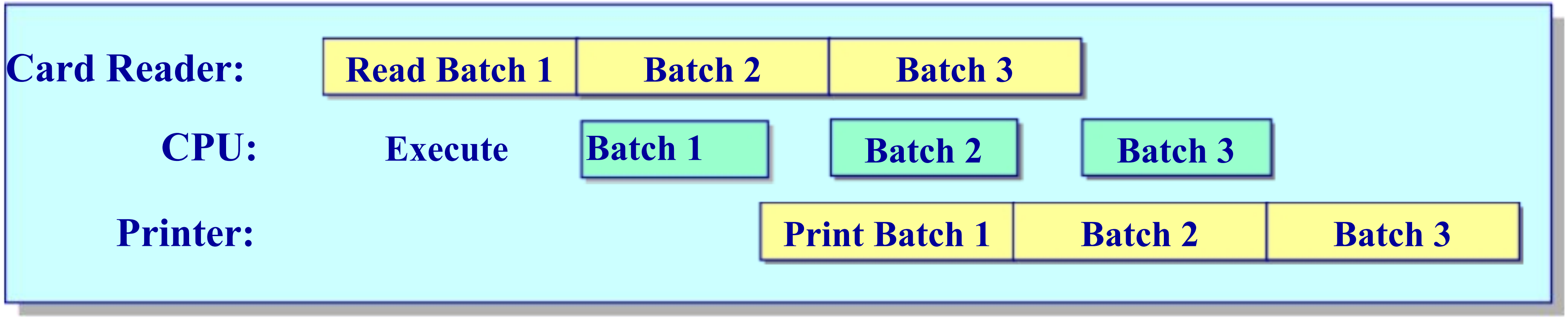
$$\frac{\text{Execution time}}{\text{Execution time + Card reader time}} = \text{\% utilization}$$



Batch/Off-line Processing (1955-1965)

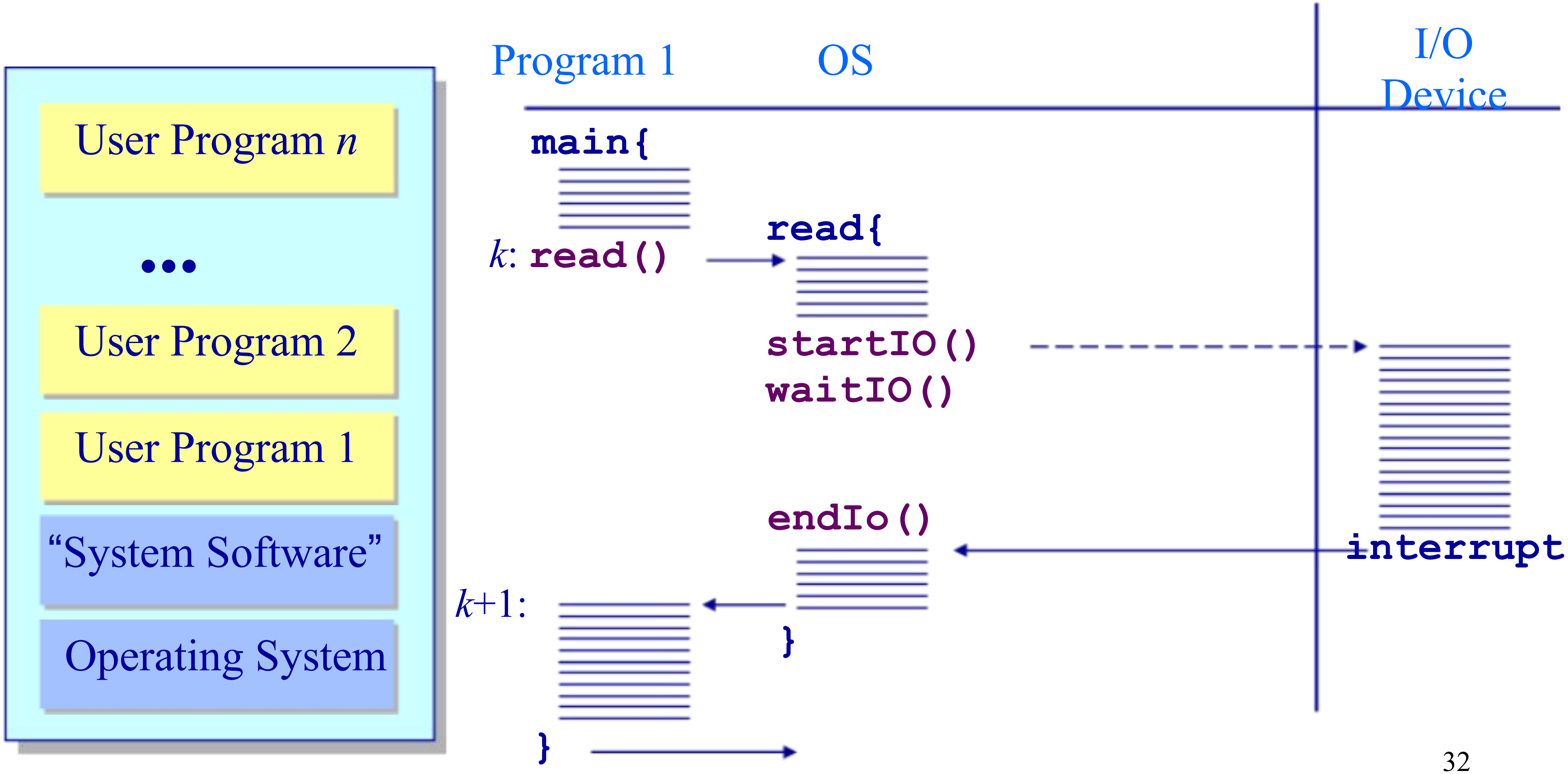
sequential *vs.* Batching execution of jobs





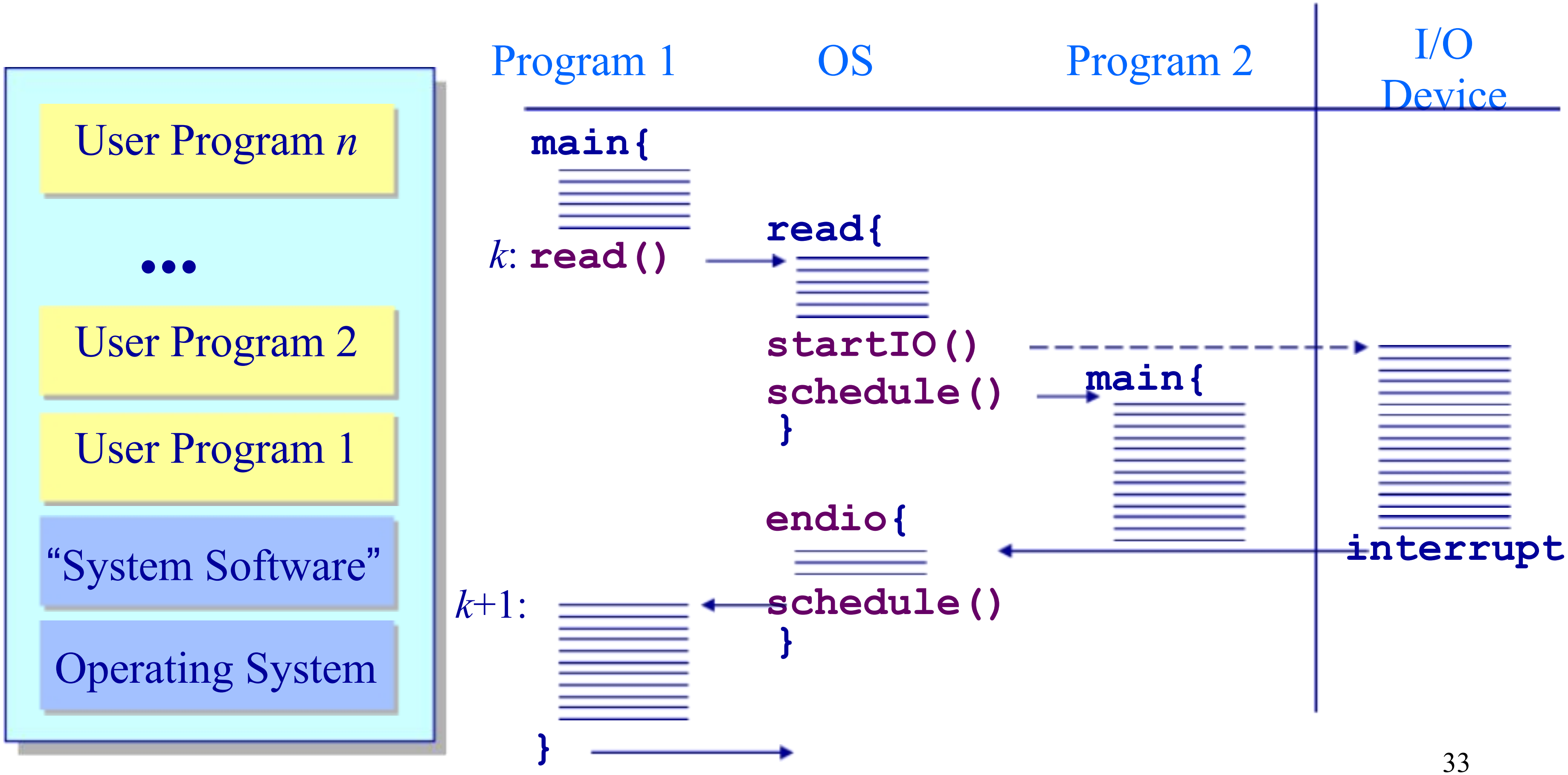
Multiprogramming (1965-1980)

Keep several jobs in memory and multiplex CPU between jobs



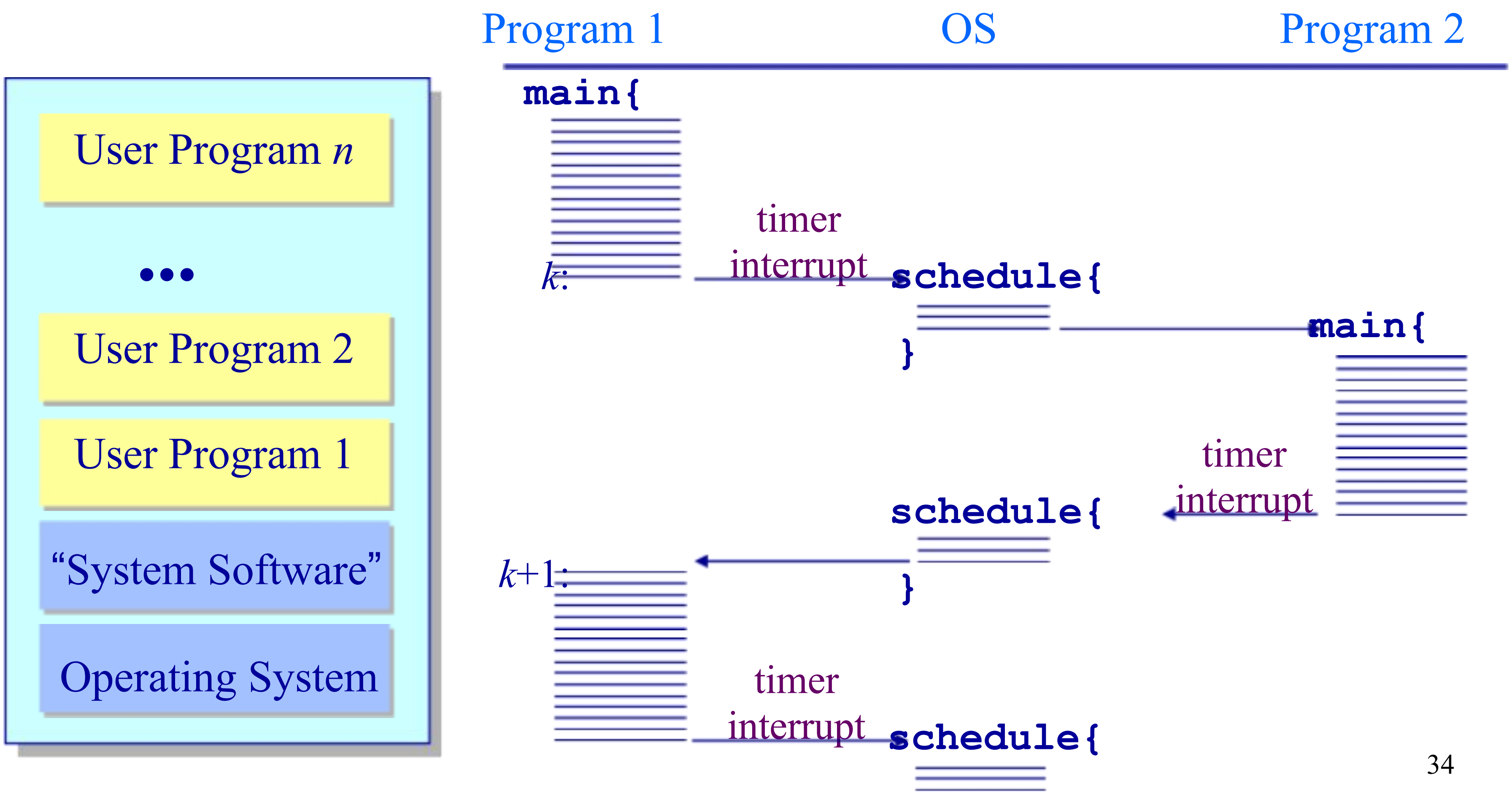
Multiprogramming (1965-1980)

Keep several jobs in memory and multiplex CPU between jobs



Timesharing (1970-)

A timer interrupt is used to multiplex CPU among jobs



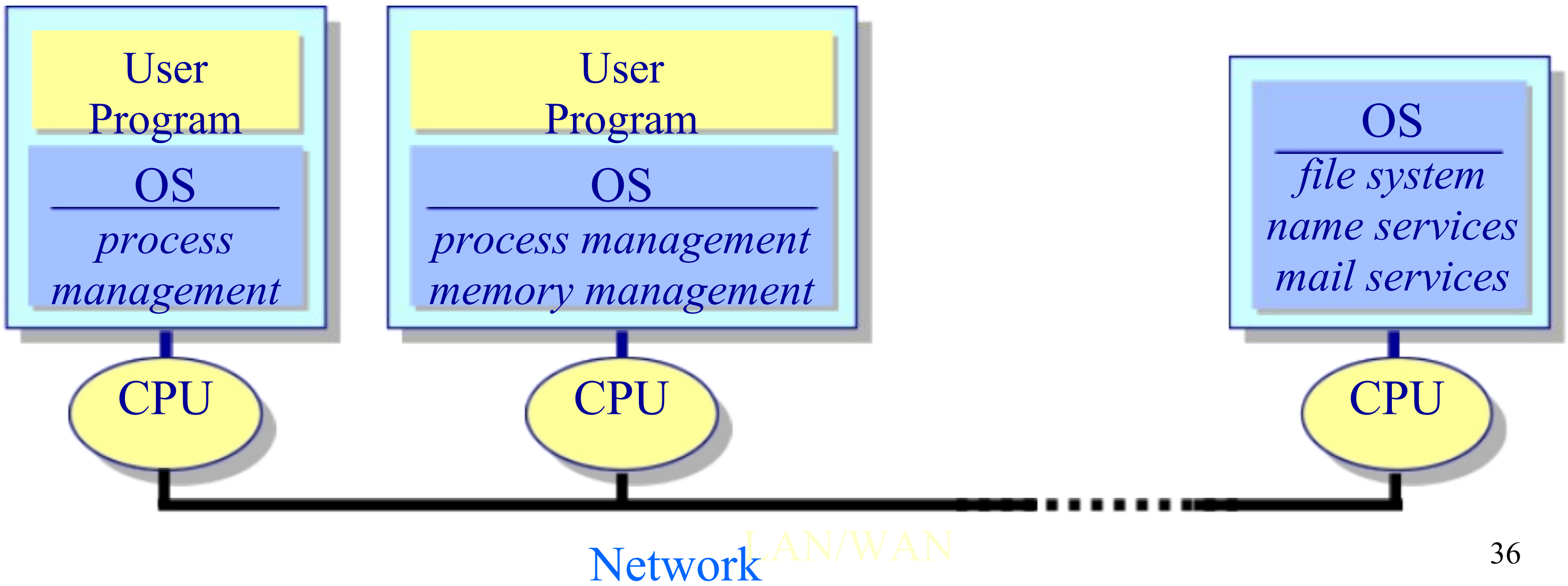
Operating Systems for PCs

- Personal computing systems
  - Single user
  - Utilization is no longer a concern
  - Emphasis is on user interface and API
  - Many services & features not present

- Evolution
  - Initially: OS as a simple service provider (simple libraries)
  - Now: Multi-application systems with support for coordination and communication
  - Growing security issues (e.g., online commerce, medical records)

Distributed Operating Systems

- Typically support distributed services
  - Sharing of data and coordination across multiple systems
- Possibly employ multiple processors
  - Loosely coupled *v.* tightly coupled systems
- High availability & reliability requirements

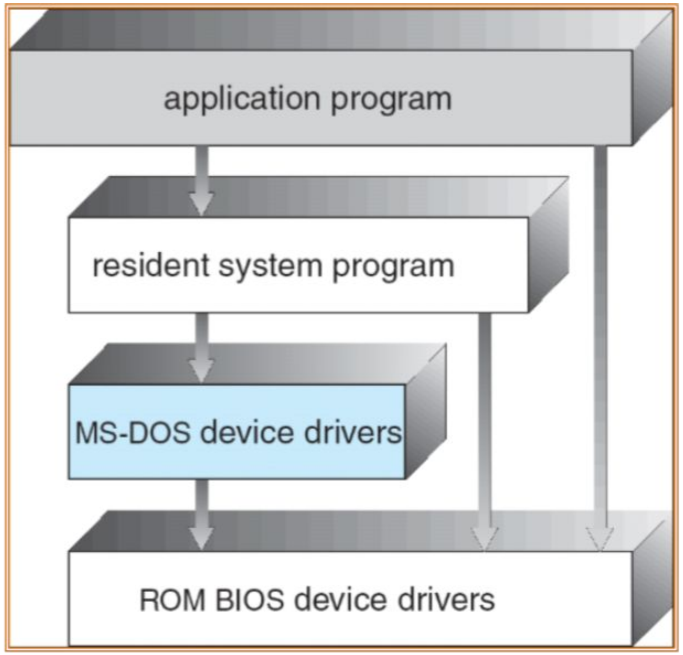


Outline

- Course Introduction
- What is an Operating System?
- Evolution of Operating Systems
- Operating-System Structures
    - Simple Structure
    - Layered Structure
    - Microkernel System Structure
    - Exokernel Structure

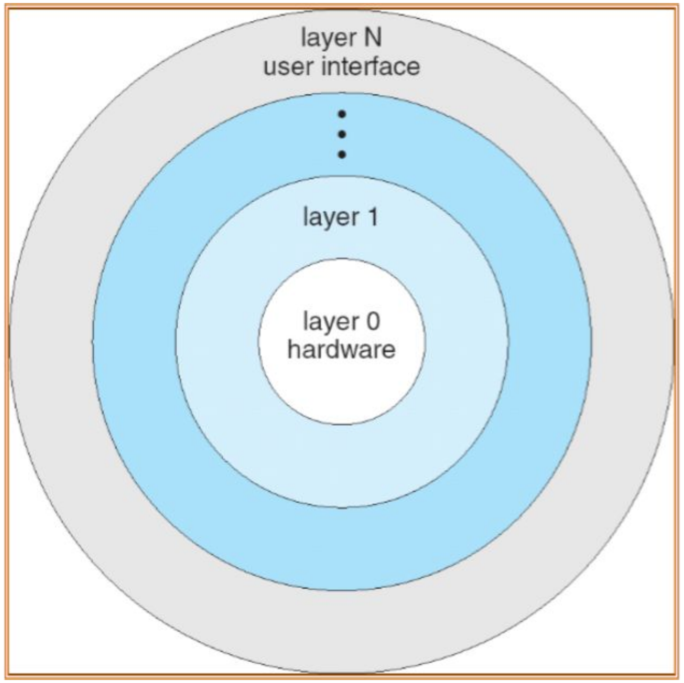
Simple Structure

- MS-DOS – written to provide the most functionality in the least space (1981~1994)
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated



Layered Approach

- Operating system is divided into a number of layers (levels)
  - Each built on top of lower layers
  - Bottom layer (layer 0), is the hardware
  - Highest (layer N) is the user interface
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers



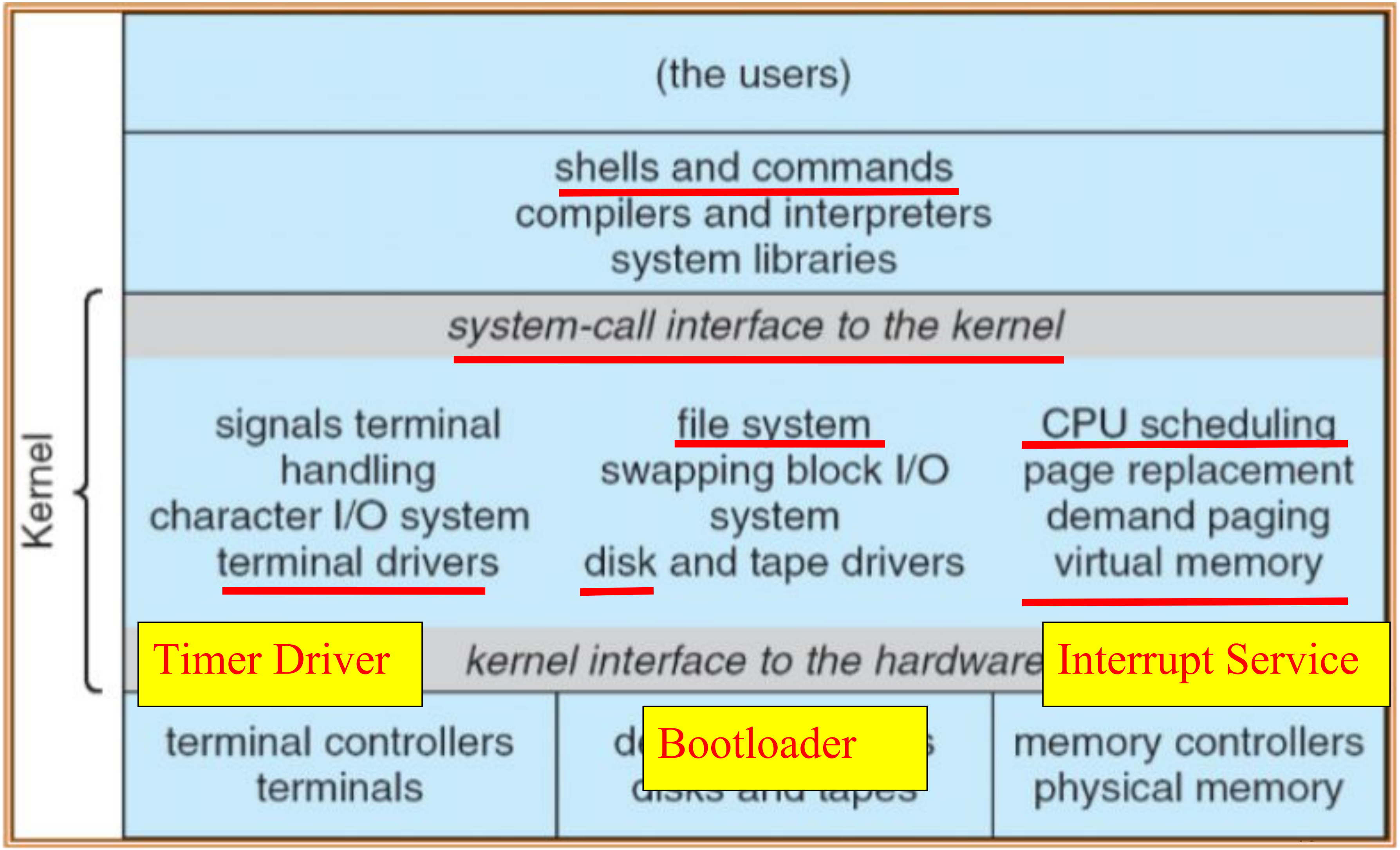
UNIX

Designed by Kenneth Thompson and Dennis Ritchie at Bell Labs in 1972.
Designed for coding the routines of the UNIX operating system.
“High level” systems programming language which created the notion of a portable operating system

K. Thompson and D. Ritchie

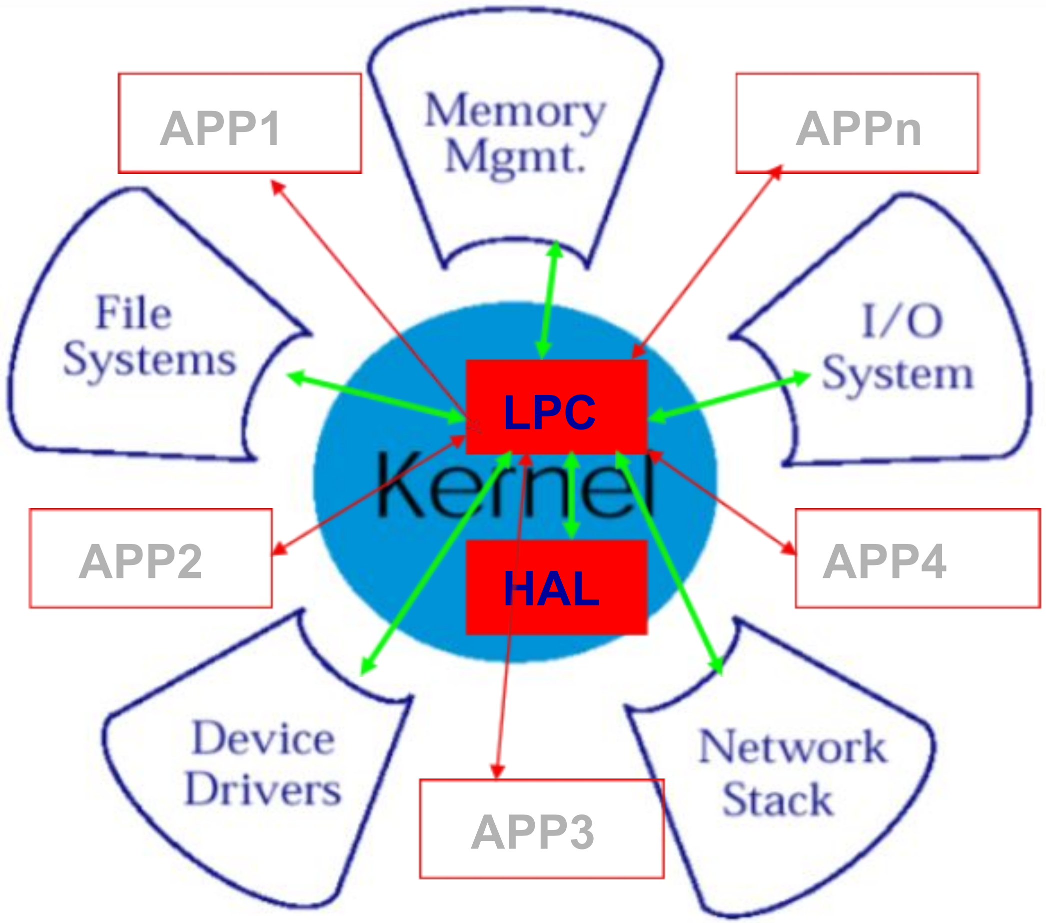
UNIX System Structure

Our Labs



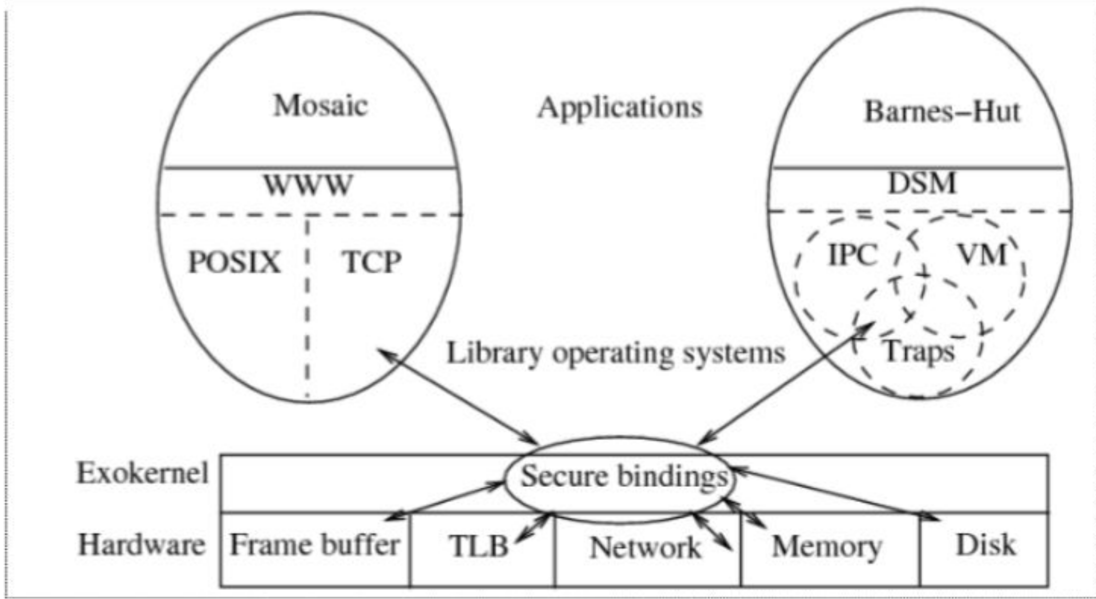
Microkernel System Structure

- Moves as much from the kernel into "*user*" space
- Communication takes place between user modules using message passing
- Benefits: flexible/security...
- Detriments: Performance



Exokernel Structure

- Overview
  - let the kernel allocate the physical resources of the machine to multiple application programs, and let each program decide what to do with these resources.
  - The program can link to an operating system library (libOS) that implements OS abstractions.
  - protected control transfer, PCT



Conclusions

OS/System is in trouble, but …

- It's unjustified
  - there are plenty of challenges and opportunities
- It's dangerous
  - the country will lose big time if we give up
- we can do it!
  - ... at least at Tsinghua

Today's Work

Start lab #0