Operating Systems

Introduction

IIIS & 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
  - Reference Textbooks
  - Course Overview
  - Course Scheduling
  - Grading & Prerequisites
  - Words to Remember
- What is an Operating System?
- Evolution of Operating Systems
- Operating-System Structures

Contact Information

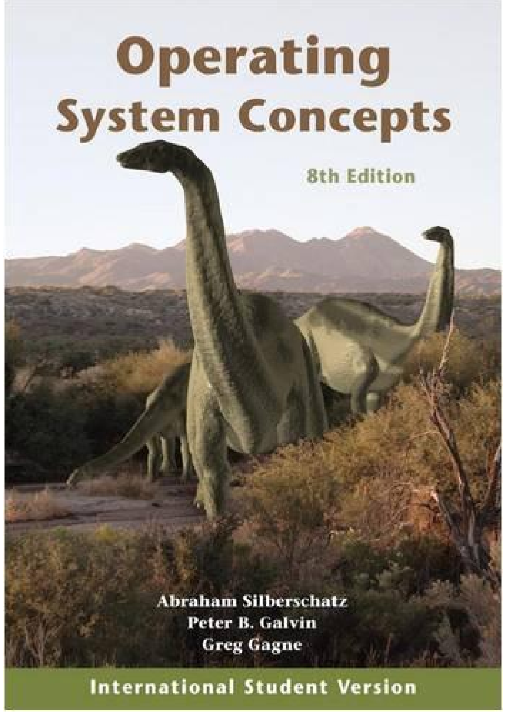
Instructor:	**CHEN, Yu 陈渝**
Office:	Room 3-124/106, FIT Building
Email:	yuchen@tsinghua.edu.cn， chyyuu@gmail.com
Wiki :	http://os.cs.tsinghua.edu.cn/oscourse/OS2013
TA:	戚啸 xinhaoyuan@gmail.com 茅俊杰 eternal.n08@gmail.com
maillist:	OS-YAO-002 in piazza.com https://groups.google.com/group/oscourse?hl=en

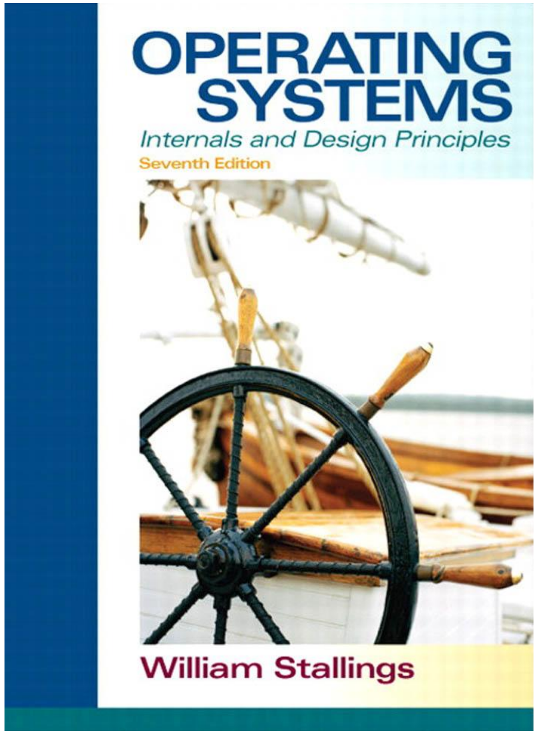
Purpose of This Course

- 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

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



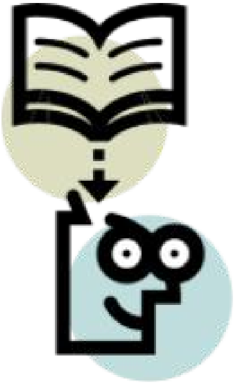


Reference Textbooks(cont.)

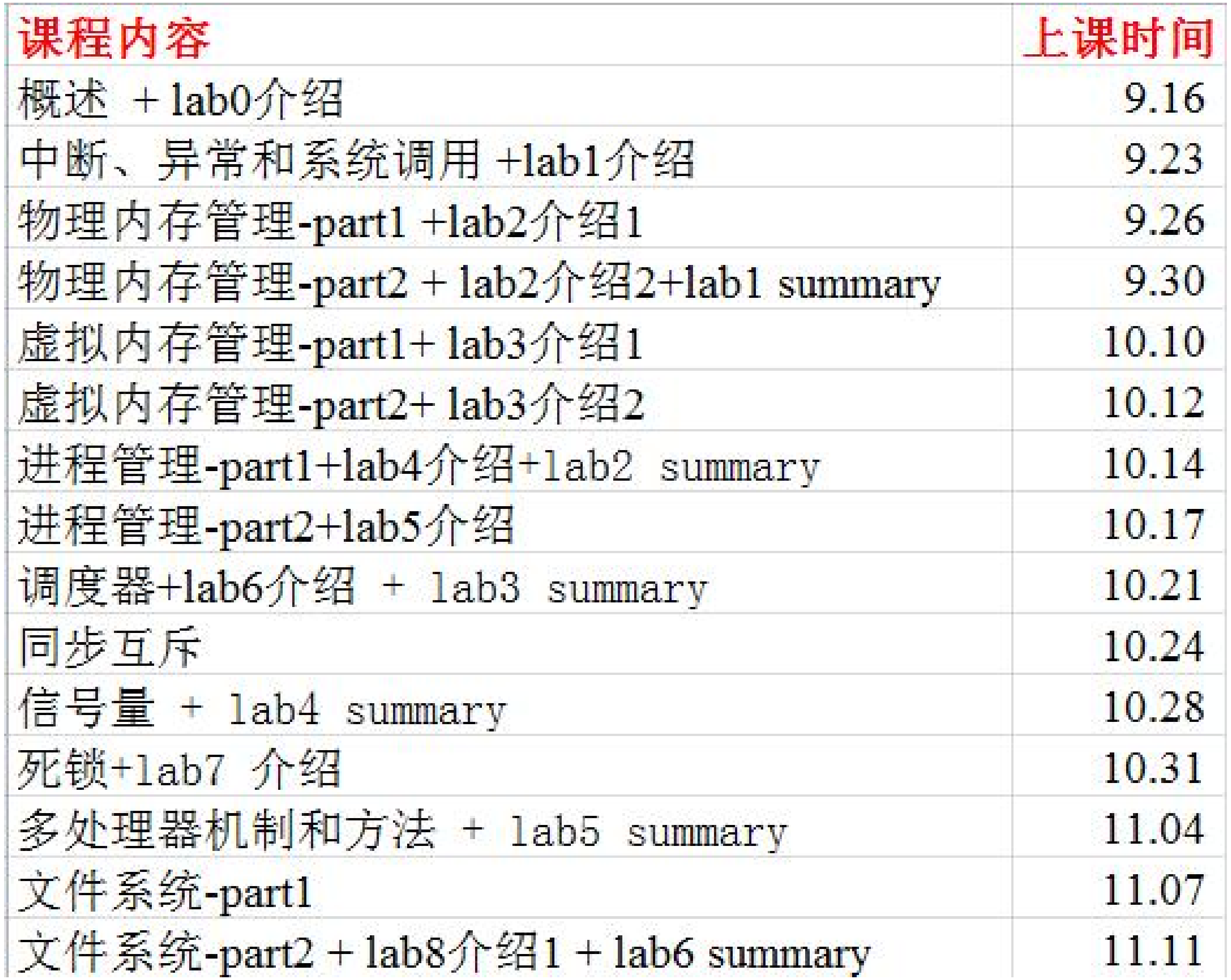
- Solaris Internals:Solaris 10 and OpenSolaris Kernel Architecture, 2nd Edition, Richard McDougall, Jim Mauro, Prentice Hall, July 10, 2006, ISBN 0-13-148209-2
- Microsoft Windows Internals, 4th Edition, Mark E. Russinovich, David A. Solomon, Microsoft Press, 2005， ISBN 0-7356-1917-4
- Understanding 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
- Practice
  - Do Labs/Projects
- Extension
  - Reading & Discuss Hot Topics

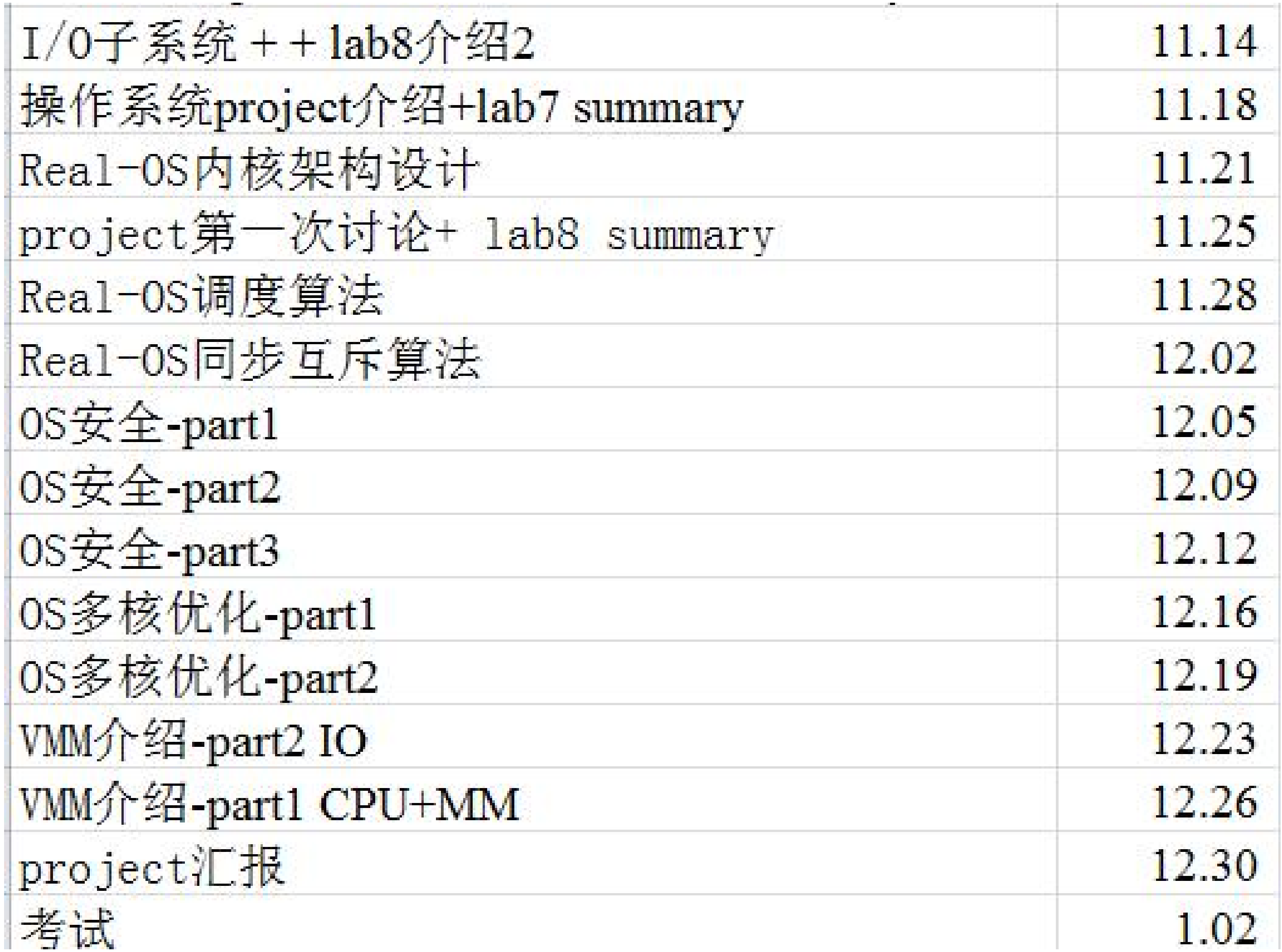


OS Course Scheduling



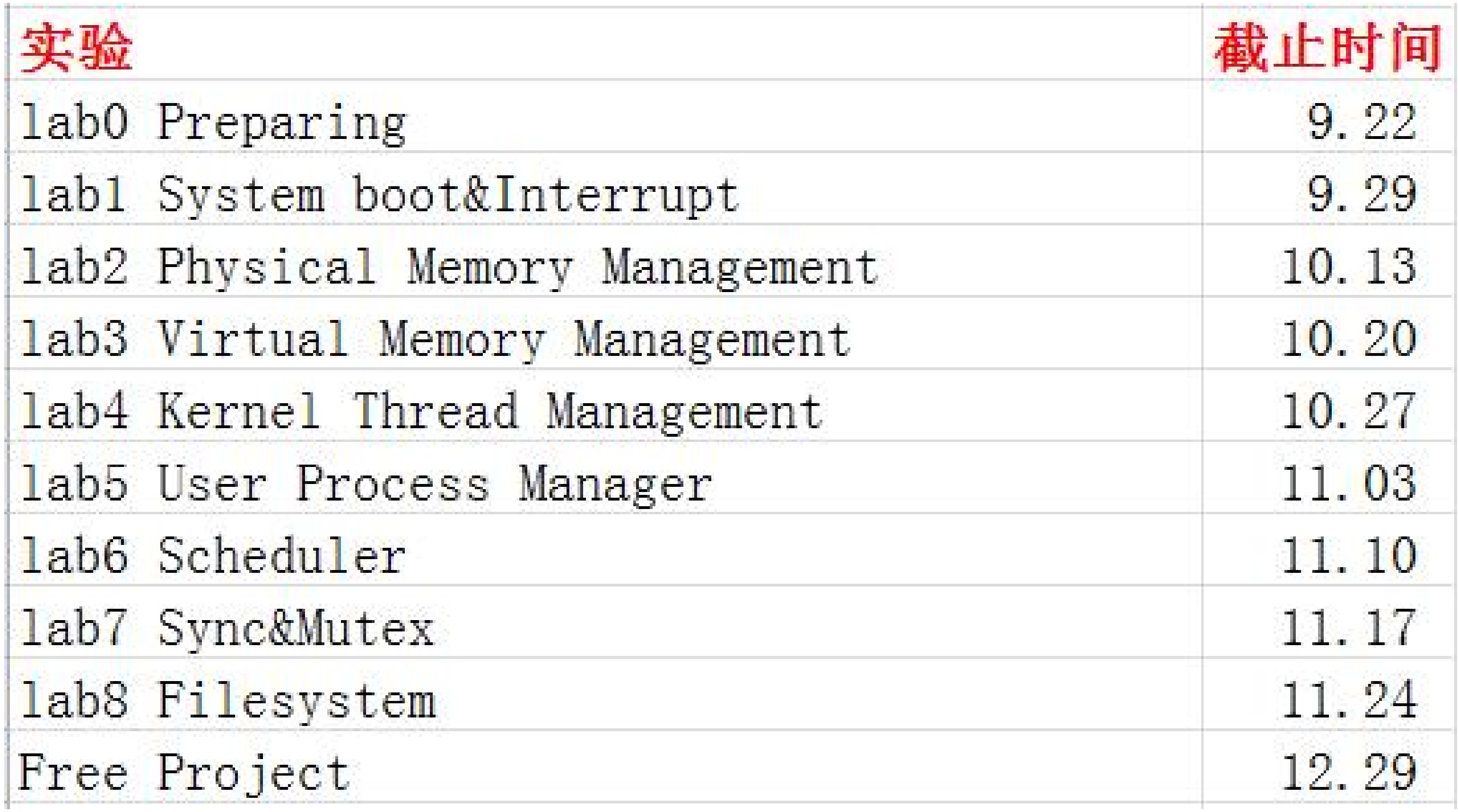
课程内容	上课时间
概述 ＋lab0介绍	9.16
中断、异常和系统调用 +lab1介绍	9.23
物理内存管理-part1 +lab2介绍1	9.26
物理内存管理-part2 + lab2介绍2+lab1 summary	9.30
虚拟内存管理-part1+ lab3介绍1	10.10
虚拟内存管理-part2+ lab3介绍2	10.12
进程管理-part1+lab4介绍+lab2 summary	10.14
进程管理-part2+lab5介绍	10.17
调度器+lab6介绍 + lab3 summary	10.21
同步互斥	10.24
信号量 + lab4 summary	10.28
死锁+lab7 介绍	10.31
多处理器机制和方法 + lab5 summary	11.04
文件系统-part1	11.07
文件系统-part2 + lab8介绍1 + lab6 summary	11.11

Course Scheduling



I/O子系统＋＋lab8介绍2	11.14
操作系统project介绍+lab7 summary	11.18
Real-OS内核架构设计	11.21
project第一次讨论+ lab8 summary	11.25
Real-OS调度算法	11.28
Real-OS同步互斥算法	12.02
OS安全-part1	12.05
OS安全-part2	12.09
OS安全-part3	12.12
OS多核优化-part1	12.16
OS多核优化-part2	12.19
VMM介绍-part2 IO	12.23
VMM介绍-part1 CPU+MM	12.26
project汇报	12.30
考试	1.02

Course Scheduling



实验	截止时间
lab0 Preparing	9.22
lab1 System boot&Interrupt	9.29
lab2 Physical Memory Management	10.13
lab3 Virtual Memory Management	10.20
lab4 Kernel Thread Management	10.27
lab5 User Process Manager	11.03
lab6 Scheduler	11.10
lab7 Sync&Mutex	11.17
lab8 Filesystem	11.24
Free Project	12.29

OS Grading & Prerequisites

- Grading
  - Labs+Homeworks: 40%
  - Middle&Final Exam: 60% (midterm 20%+Final 40%) 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

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 useful!

OS is challenging!

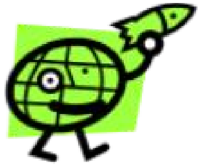
I want to be involved!

Outline

- Course Introduction
- What is an Operating System?
  - Some Operating Systems
  - Why Study OS?
  - OS is Challenging
  - Purpose of This Course
  - Computer System Structure
  - Operating System Definition
  - Functions of OS
- Evolution of Operating Systems
- Operating-System Structures

Words to Remember for this Course

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

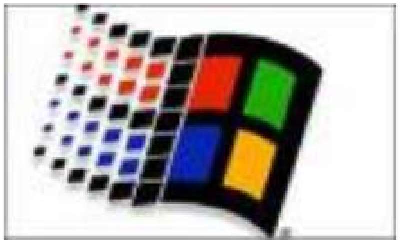


"Genius is 1% inspiration and **99%** perspiration"
-- Thomas Edison

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

Some Operating Systems

Some Operating Systems



Why Study OS?

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

Why Study OS?

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

Why Study OS?

Top Conferences on Operating System Research

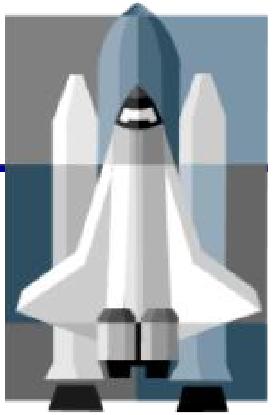
- ACM Symposium on Operating Systems Principles (SOSP)
  - ACM SIGOPS
  - Every two years (odd number: 1967-)
  - ~20 papers
- USENIX Symposium on Operating Systems Design and Implementation (OSDI)
  - USENIX
  - Every two years (even number: 1994-)
  - ~20 papers

Why Study OS?

Most Influential Operating Systems Papers

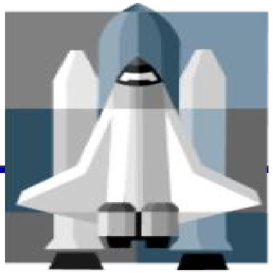
- SIGOPS Hall-of-Fame Awards
  - Paper must be published in peer-reviewed literature at least ten years in the past
  - 10 awards so far
  - 1 award every year starting next year
- If you want to do OS research
  - Must read and understand these 10 papers

OS is Challenging



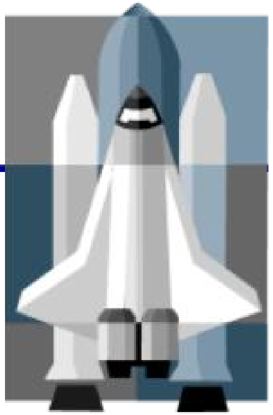
- The OS is really large
  - Windows XP is 45 million lines
- The OS manages concurrency
  - Concurrency leads to interesting programming challenges
- OS code manages raw hardware
  - Timing dependent behavior, undocumented behavior, HW bugs
- OS code must be efficient, low CPU, memory, disk use
- OS 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
    - monitors and philosophers don’t live in OS kernels
    - the locking problems there are in kernels require too much background
  - disk scheduling is mostly irrelevant (SCSI does it for you)
  - process scheduling is a small topic (worth 1h max)

OS is Challenging



- OS is about:
  - tradeoffs
    - time vs space
    - performance vs predictability
    - fairness vs performance (which design will work and why?)
  - Hardware
    - 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!

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

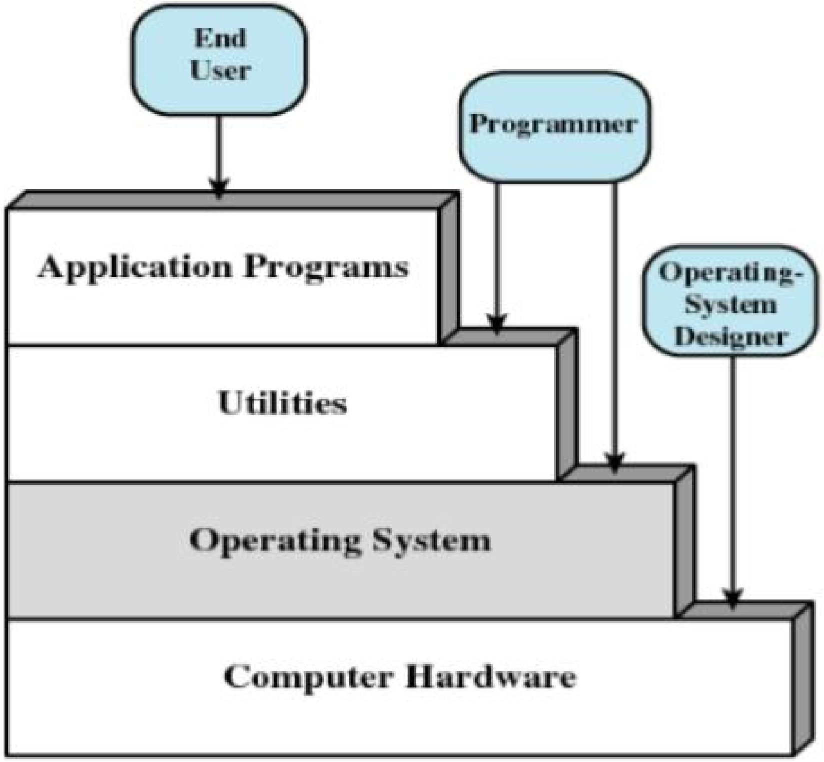
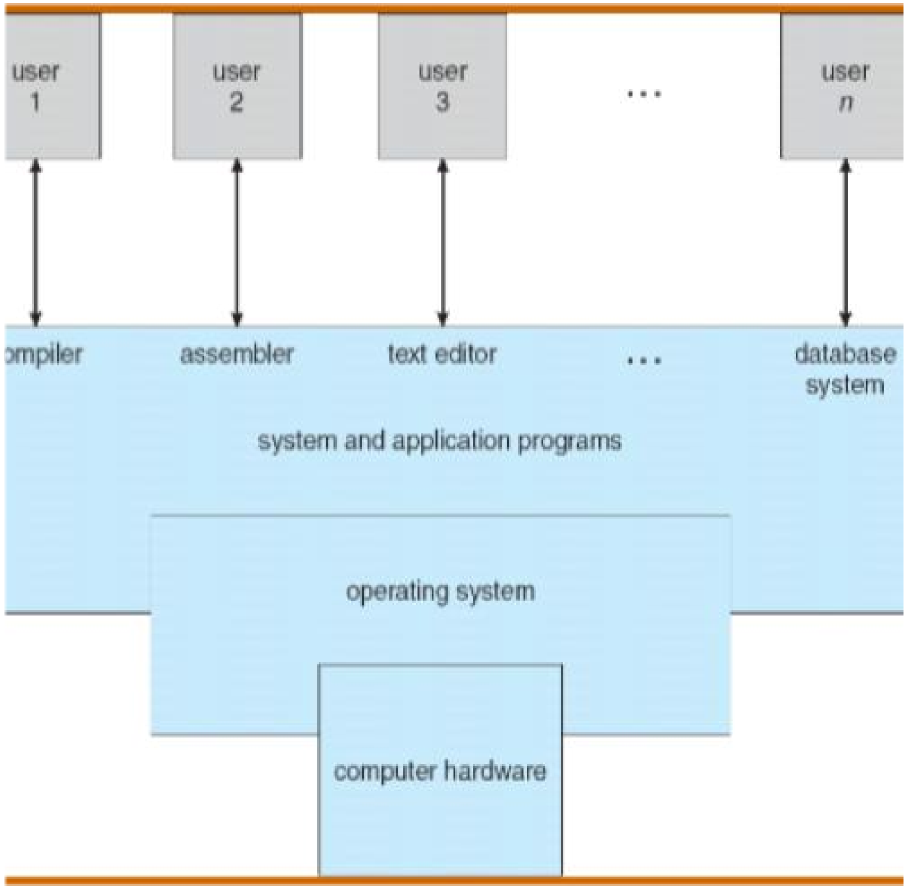
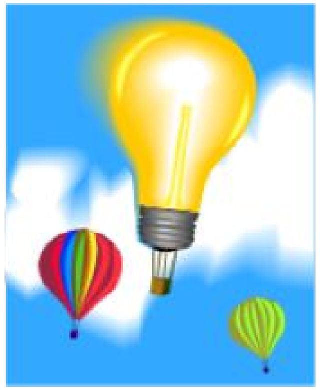


Figure 2.1 Layers and Views of a Computer System



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



OS 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 20123

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

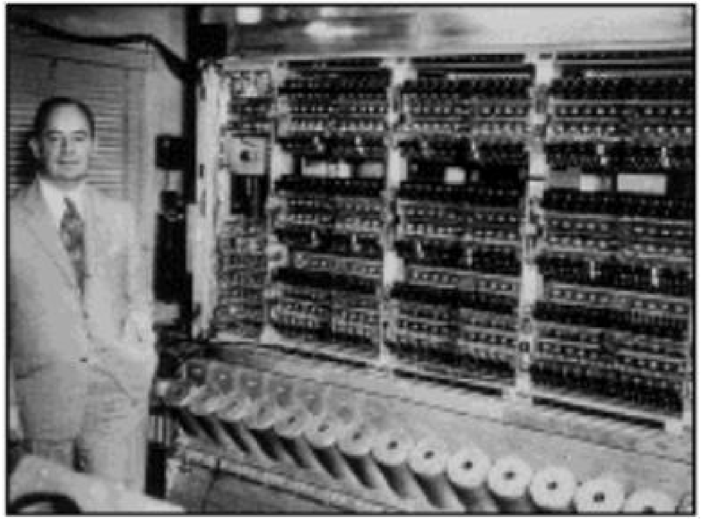
Evolution of an Operating System

- 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 ('45-'55)

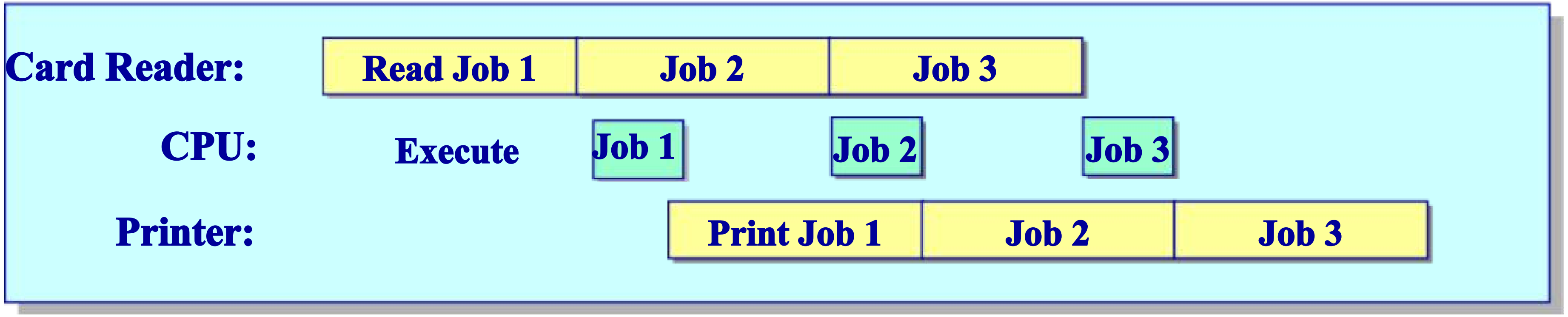
- OS = *loader* + *libraries of common subroutines*
- Problem: low *utilization* of expensive components

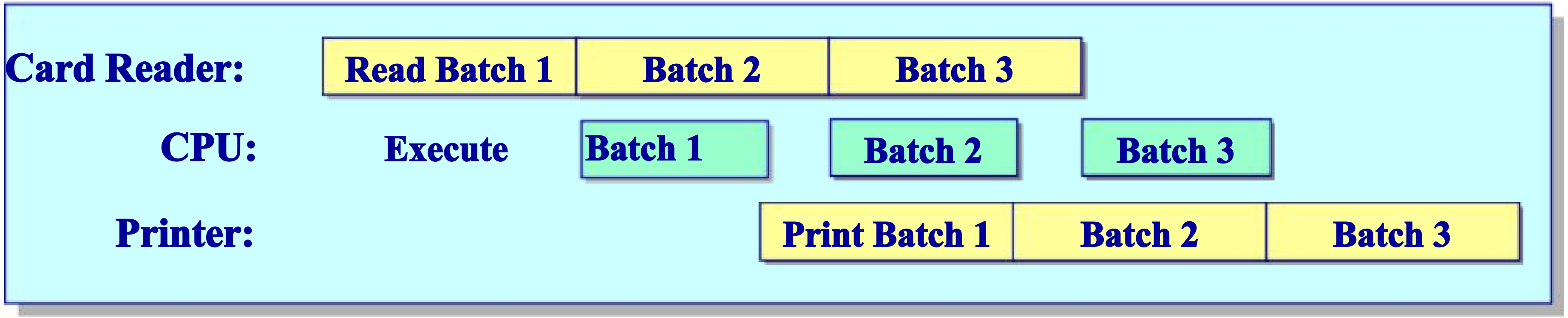
$$\frac{\textit{Execution time}}{\textit{Execution time} + \textit{Card reader time}} = \%\ \textit{utilization}$$



Batch/Off-line Processing (‘55-‘65)

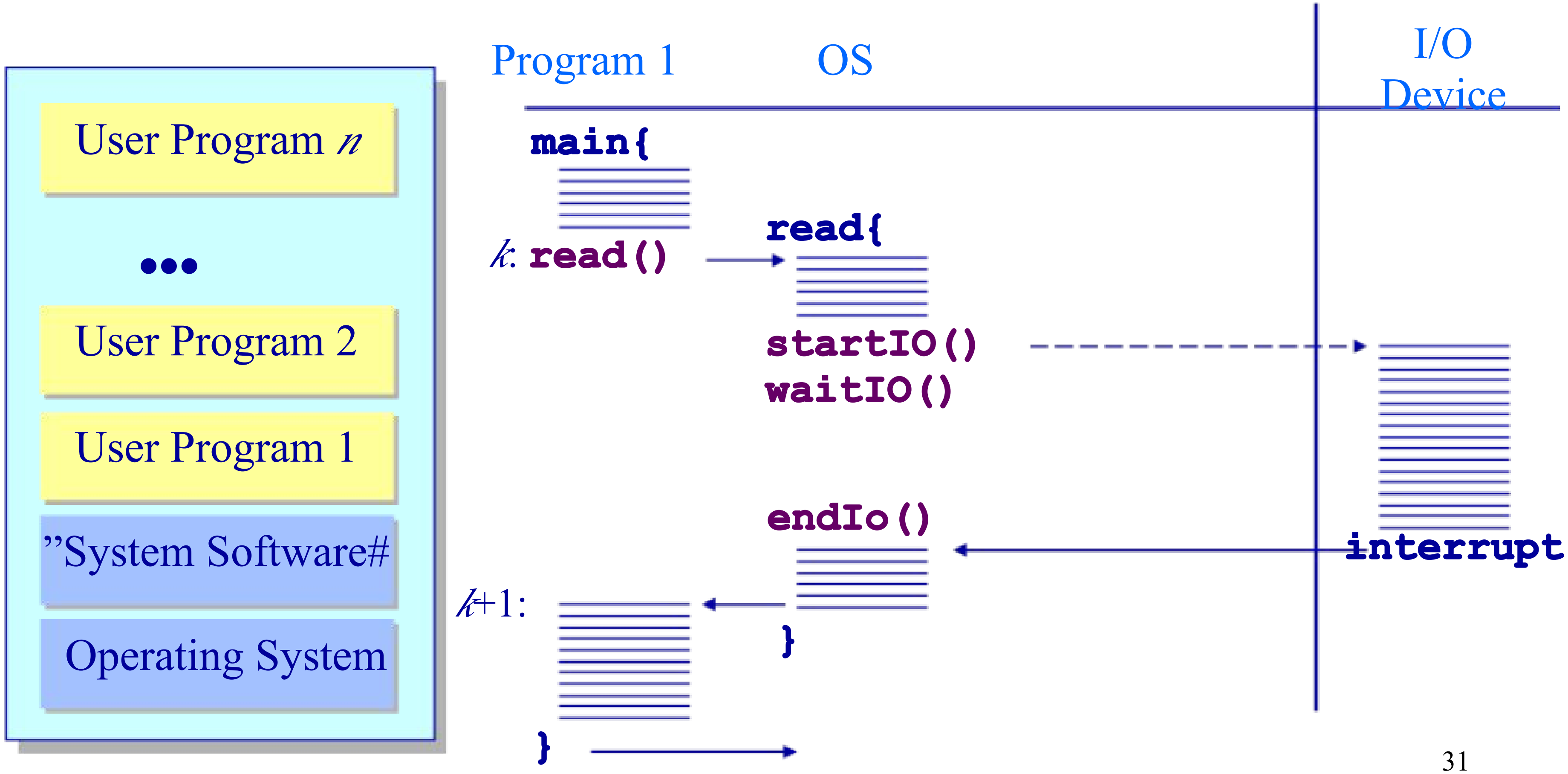
- sequential *vs.* Batching execution of jobs





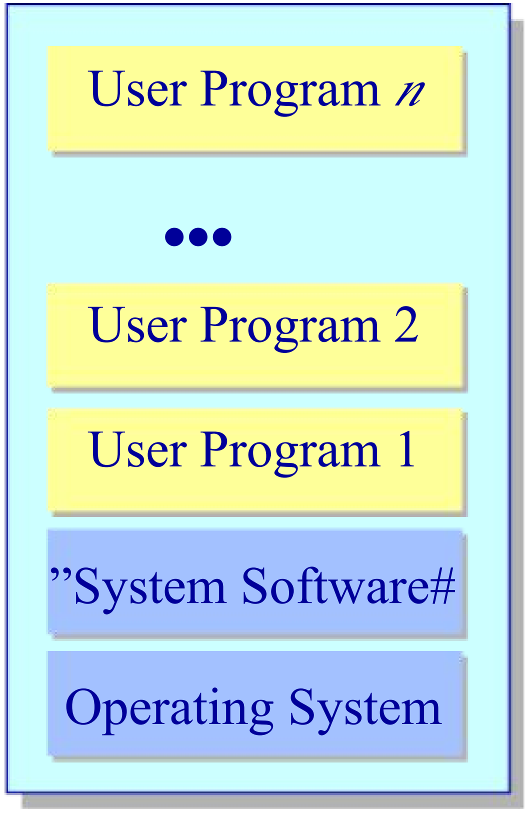
Multiprogramming (‘65-‘80)

- Keep several jobs in memory and multiplex CPU between jobs



Multiprogramming ('65-'80)

- Keep several jobs in memory and multiplex CPU between jobs



User Program *n*

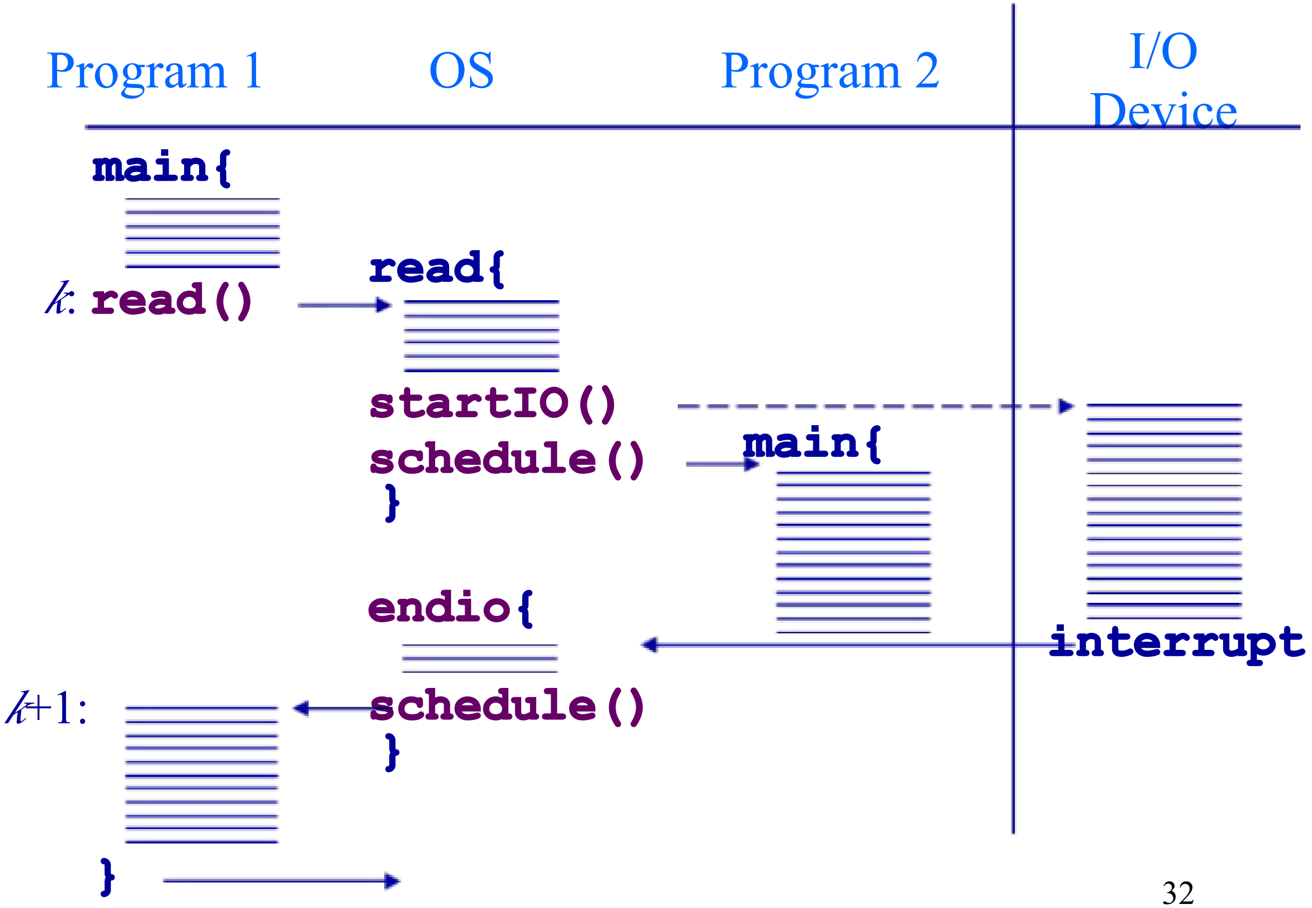
•••

User Program 2

User Program 1

"System Software#

Operating System



Program 1

OS

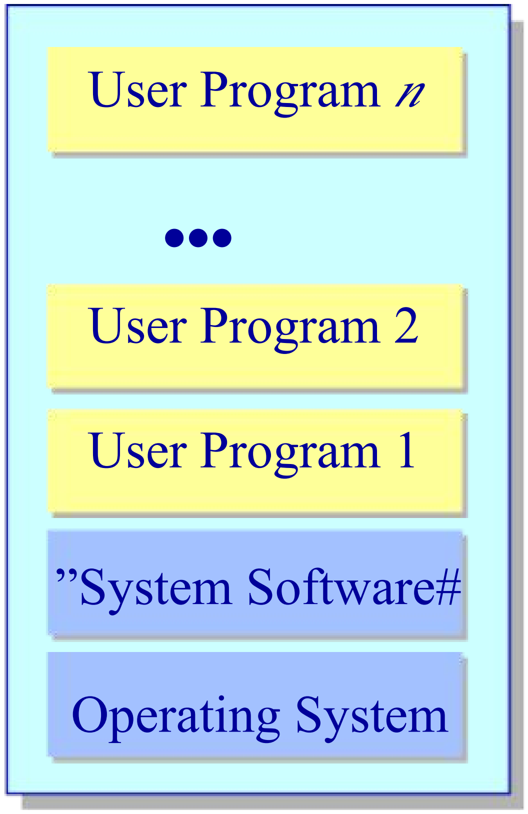
Program 2

I/O Device

```
main{
k: read()
read{
startIO()
schedule()
}
main{
endio{
interrupt
schedule()
}
k+1:
}
```

Timesharing (‘70-)

- A timer interrupt is used to multiplex CPU among jobs


User Program *n*

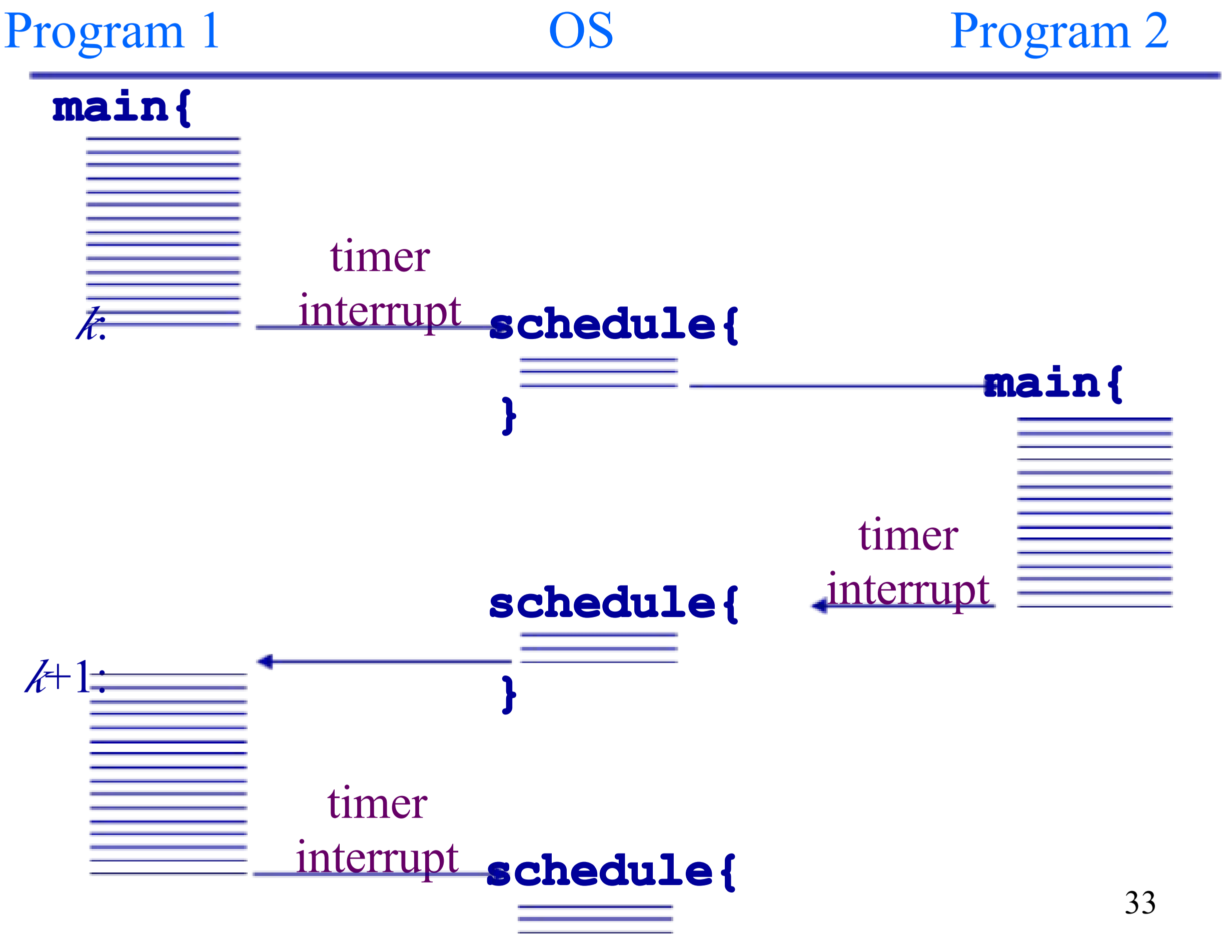
•••

User Program 2

User Program 1

”System Software#

Operating System


Program 1

OS

Program 2

main{

k:

timer interrupt

schedule{

}

main{

timer interrupt

schedule{

}

k+1:

timer interrupt

schedule{

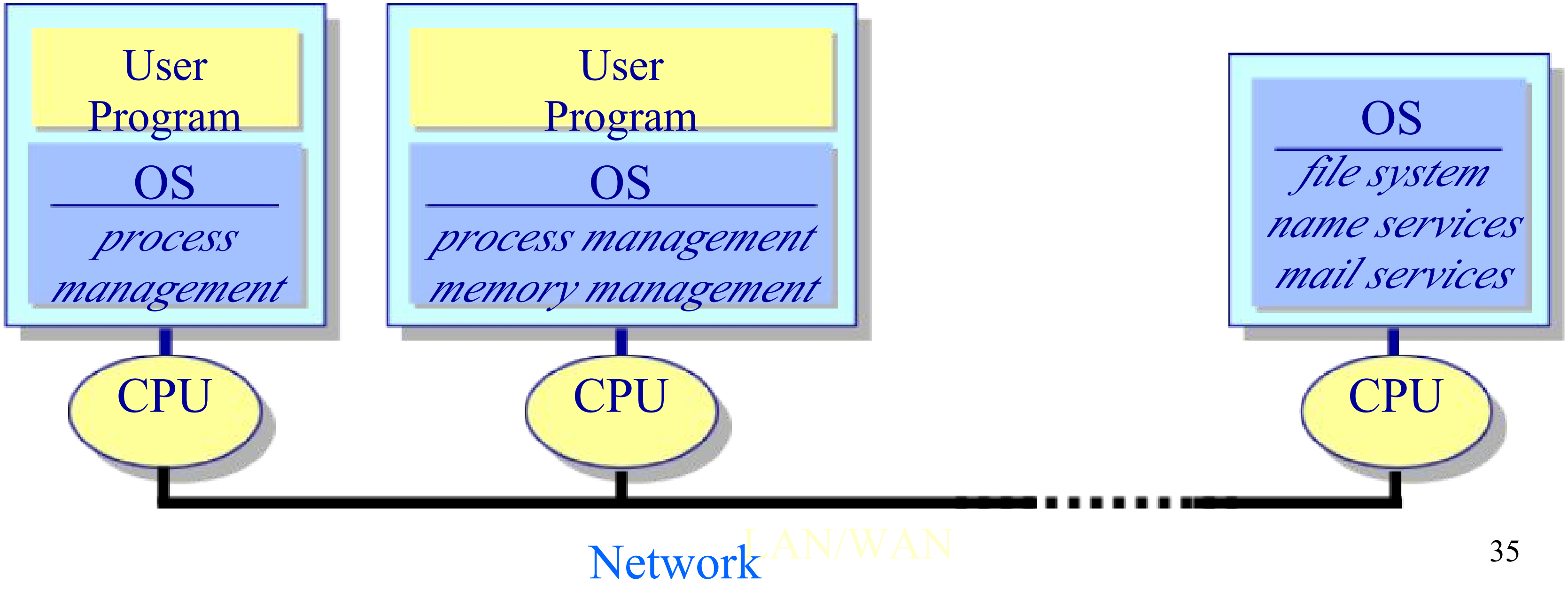
Operating Systems for PCs

- 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



User Program

OS

process management

CPU

User Program

OS

process management

memory management

CPU

OS

file system

name services

mail services

CPU

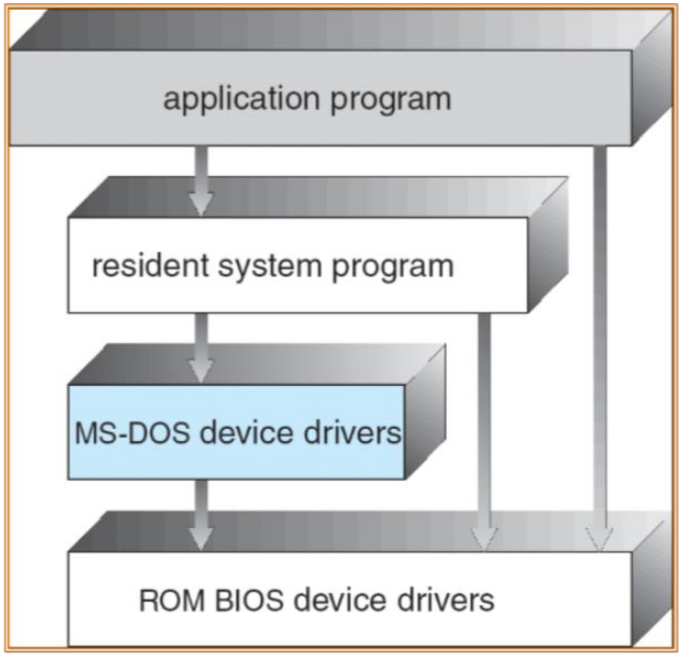
Network LAN/WAN

Outline

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

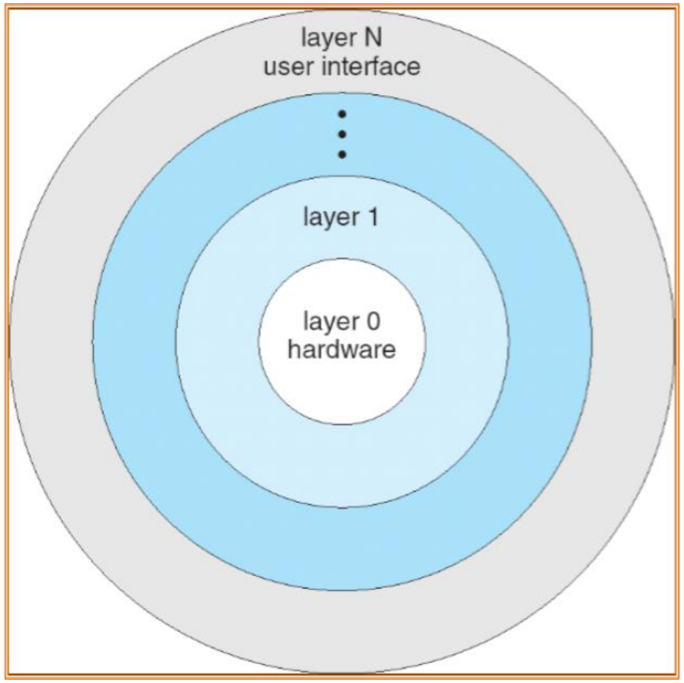
Simple Structure

- 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



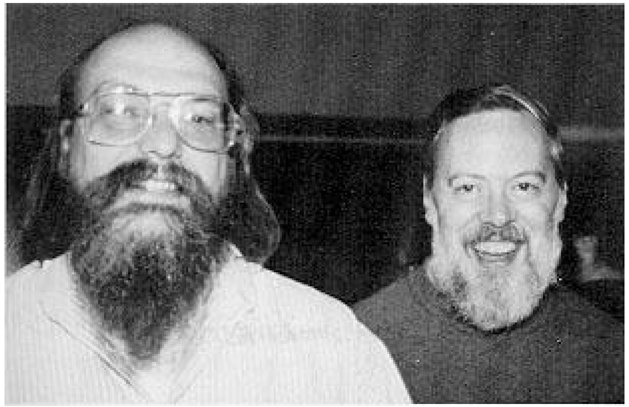
OS Layered Approach

- 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

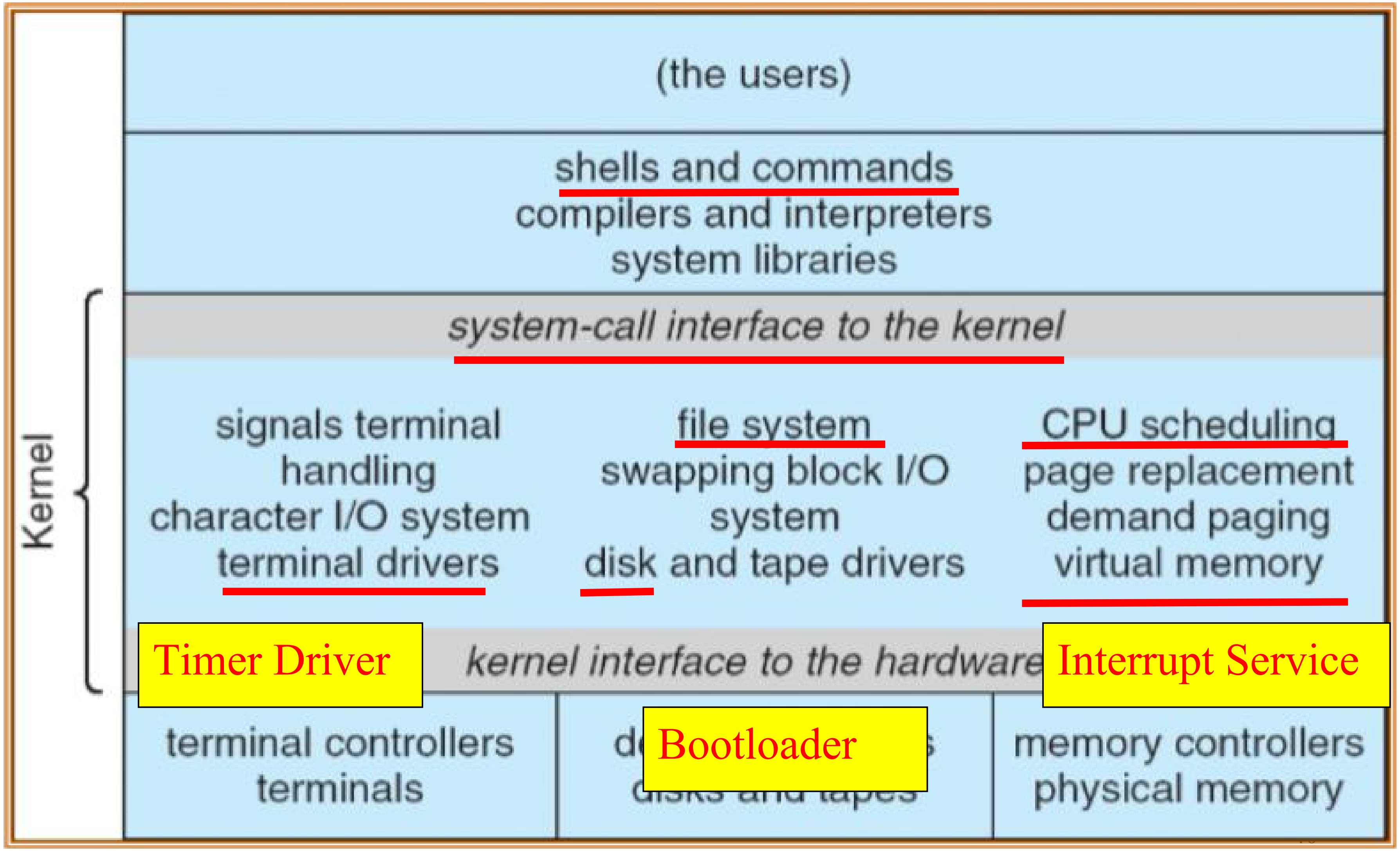
- Designed by Kenneth Thompson and Dennis Ritchie at Bell Labs in 1972.
- Designed for coding the routines of the UNIX operating system.
- ”High level# systems programming language which created the notion of a portable operating system



K. Thompson and D. Ritchie

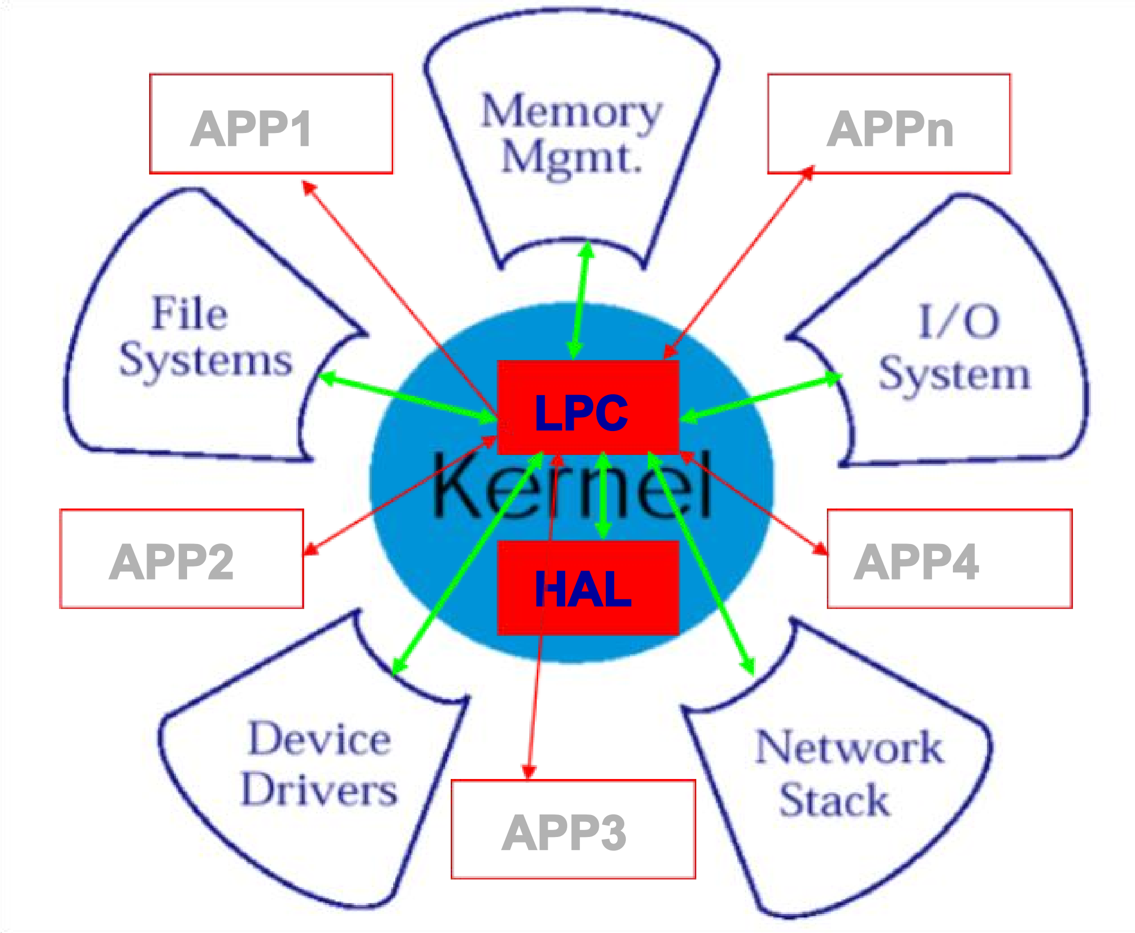
UNIX System Structure

Our Labs



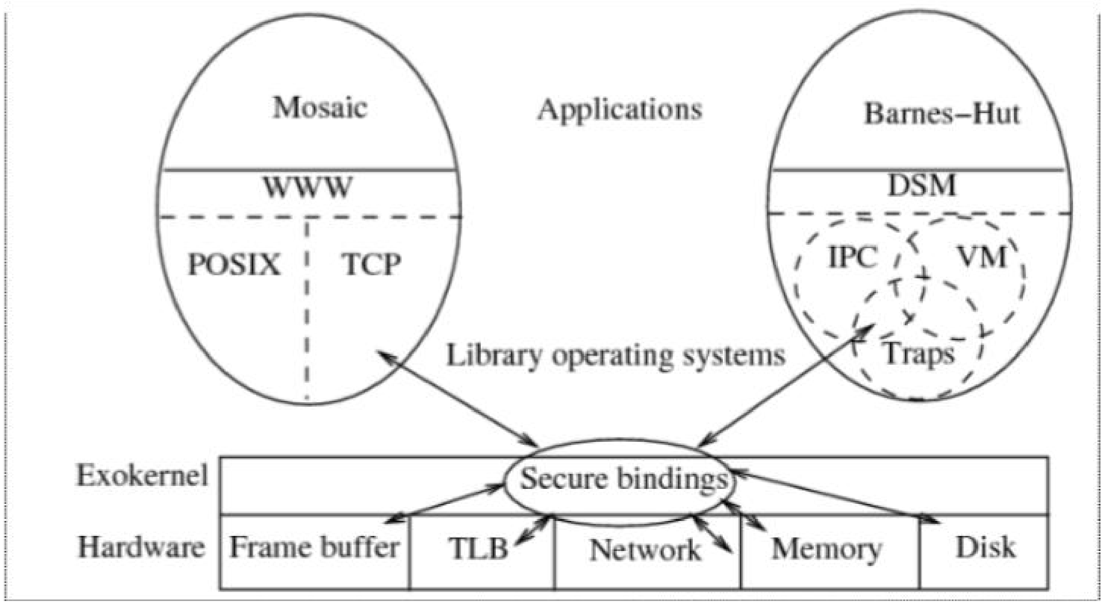
Microkernel System Structure

- 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



OS 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