Operating System Concepts

Lecture 1: Course Logistics and Introduction

Omid Ardakanian
oardakan@ualberta.ca
University of Alberta

MWF 12:00-12:50 VVC 2 215

Today's class

- Course organization and outline
  - topics
  - learning outcomes
- Assignments and exams
- Policies
- Introduction to Operating System (OS)

General information

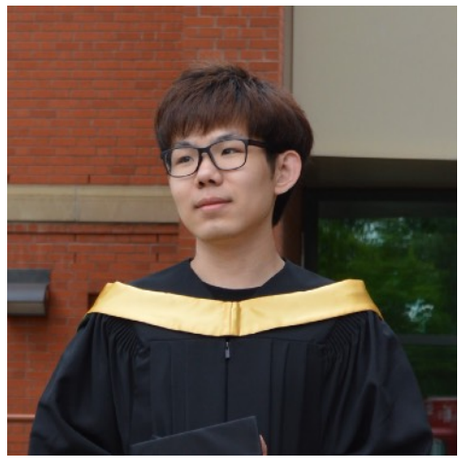
- Course web page: https://eclass.srv.ualberta.ca/course/view.php?id=53653
  - assignment submission, supplemental readings and slides, discussion forum, etc.
- Contact information
  - Google group: cmput-379-f19@googlegroups.com
  - instructor: oardakan@ualberta.ca

General information

- Course web page: https://eclass.srv.ualberta.ca/course/view.php?id=53653
  - assignment submission, supplemental readings and slides, discussion forum, etc.
- Contact information
  - Google group: cmput-379-f19@googlegroups.com
  - instructor: oardakan@ualberta.ca
- TAs

Max Ellis

Peiran Yao



Tianyu Zhang

Aidan Bush

Course objectives

- Enhance your knowledge and skills in developing programs that
  - utilize advanced Operating System (OS) services
  - support concurrent operations properly
  - perform inter-process communications via shared memory, signals, pipes, and sockets
  - interact with the Internet (client-server programs)
- Introduce fundamental ideas underlying the design and implementation of modern operating systems

How does this course fit in the CS/CE Curriculum?

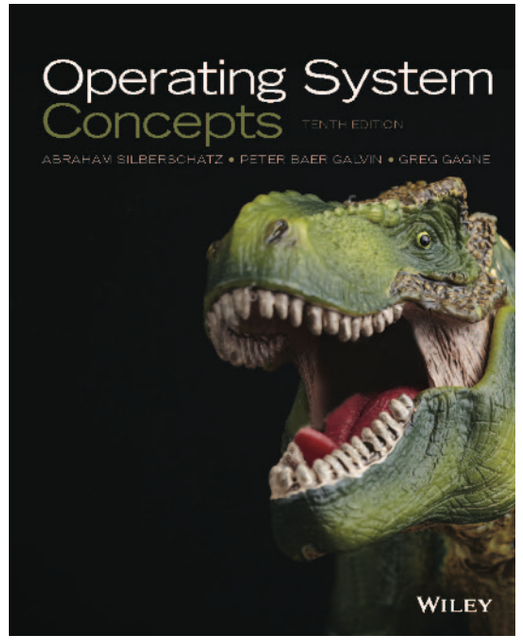
- CMPUT 229: Computer Organization and Architecture I
  - organization of the hardware architecture
  - fundamentals behind program execution
- CMPUT 379: Operating System Concepts
  - operating system structure and services
  - programming in the UNIX environment
- CMPUT 313: Computer Networks
  - network architecture, protocols, and applications

Prerequisites

- CMPUT 201 and 204 or 275; one of CMPUT 229, EE 380 or ECE 212.
- To understand this course you must have
  - a good grasp of computer organization and hardware (CPU instruction sets, memory hierarchy, I/O systems, etc.)
  - solid programming skills (in particular C/C++)
  - a working knowledge of data structures and algorithms

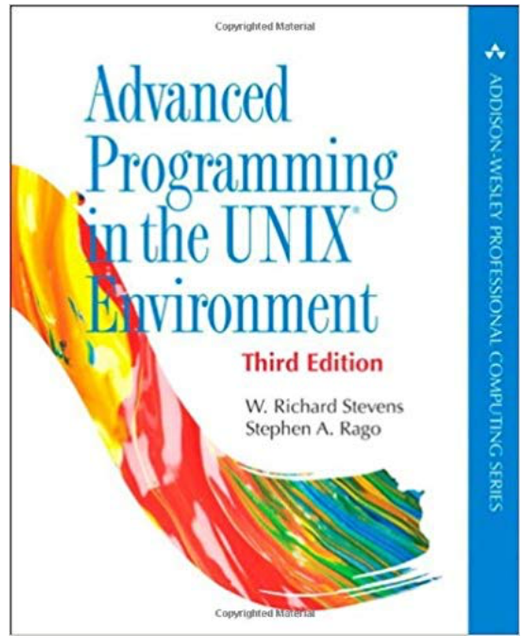
Textbook

1. A. Silberschatz, P. Galvin, and G. Gagne, Operating System Concepts, 10th Edition, John Wiley, 2018 (**required**).



2. W. Stevens, and S. Rago, Advanced Programming in the Unix Environment, 3rd Edition, Addison-Wesley, 2013 (**highly recommended**)

3. R. H. Arpaci-Dusseau, and A. C. Arpaci-Dusseau, Operating Systems: Three Easy Pieces, 1st Edition, ArpaciDusseau Books, 2018 (**highly recommended**) available for free online: http://pages.cs.wisc.edu/ ~remzi/OSTEP/

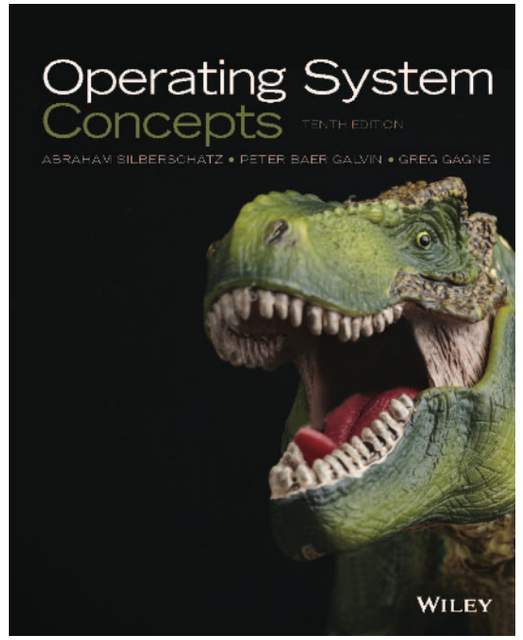


Also keep your architecture book on hand.

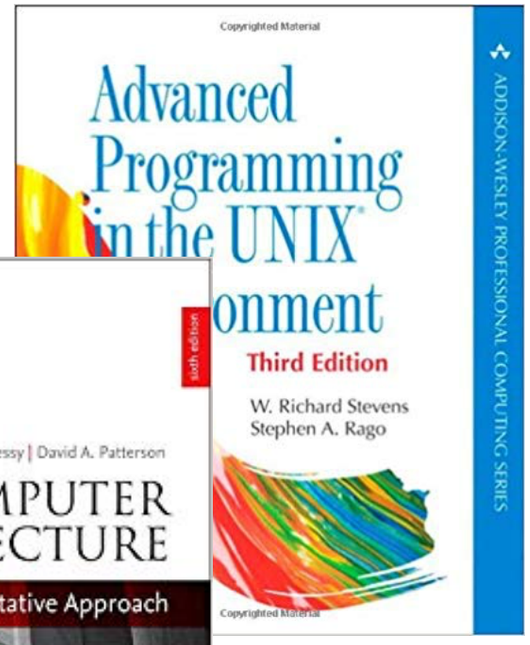
Textbook

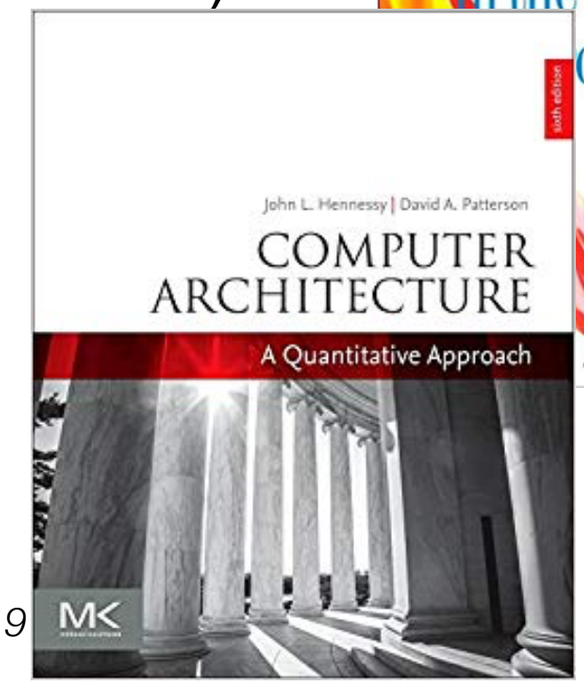
1. A. Silberschatz, P. Galvin, and G. Gagne, Operating System Concepts, 10th Edition, John Wiley, 2018 (**required**).

2. W. Stevens, and S. Rago, Advanced Programming in the Unix Environment, 3rd Edition, Addison-Wesley, 2013 (**highly recommended**)



3. R. H. Arpaci-Dusseau, and A. C. Arpaci-Dusseau, Operating Systems: Three Easy Pieces, 1st Edition, ArpaciDusseau Books, 2018 (**highly recommended**) available for free online: http://pages.cs.wisc.edu/ ~remzi/OSTEP/





Also keep your architecture book on hand.

Evaluation

- CMPUT 379 has two substantial components:
  - conceptual component: covered in class and tested in the exams
  - hands-on programming component: covered in lab sessions and tested in the assignments
- 45% Programming assignments
  - C/C++ programming in the UNIX environment
  - (a) small-scale development and (b) simulating part of an operating system
  - rubrics will be posted online
  - submissions within 24 hours after the deadline are subject to a 20% penalty
- 20% Midterm exam (Monday 4 Nov., during the class time)
- 35% Final exam

Assignments

- 3 to 4 programming assignments spread over the term
  - must be completed **individually**
    - you cannot **borrow** code from someone, **hire** someone to write your code, or even **look at** someone else's code
    - you cannot use code downloaded from the Internet
    - you need to cite all resources you used
    - you might consult the TAs and instructor, but not other students
  - can be done in C or C++
  - can develop code on your own machine but must make sure that it runs on a lab machine
  - strict late policies and policies on cheating (plagiarism detection software will be used)
- See eClass for assignment release and due dates

OS labs

- Two lab sessions per week starting from the third week of classes
  - ETLC E1003: Tuesday, 5:00 – 7:50 PM
    CAB 311: Thursday, 2:00 – 4:50 PM
  - please go to the session you signed up for
- Lab attendance is not mandatory (but it is highly recommended)
  - we won't have enough time to implement some of the concepts in class
  - TAs will help you with programming assignments and course topics

Other policies

- Cell phones must be off or on silent during lectures and lab sessions
- Slides will be posted on eClass after each lecture
- Always compile and run your code on Linux lab machines before submitting it

Overview of 379 topics

- Introduction
  - OS Roles, Services, Structure
- Process management
  - creation, suspension, resumption, and termination of user & system processes
  - scheduling processes and threads on the CPUs
  - mechanisms for process communication and synchronization
- Main memory and virtual memory management
  - keeping track of which parts of memory are currently being used
  - allocating and deallocating memory space as needed
  - deciding on which processes and data to move into and out of memory

Overview of 379 topics

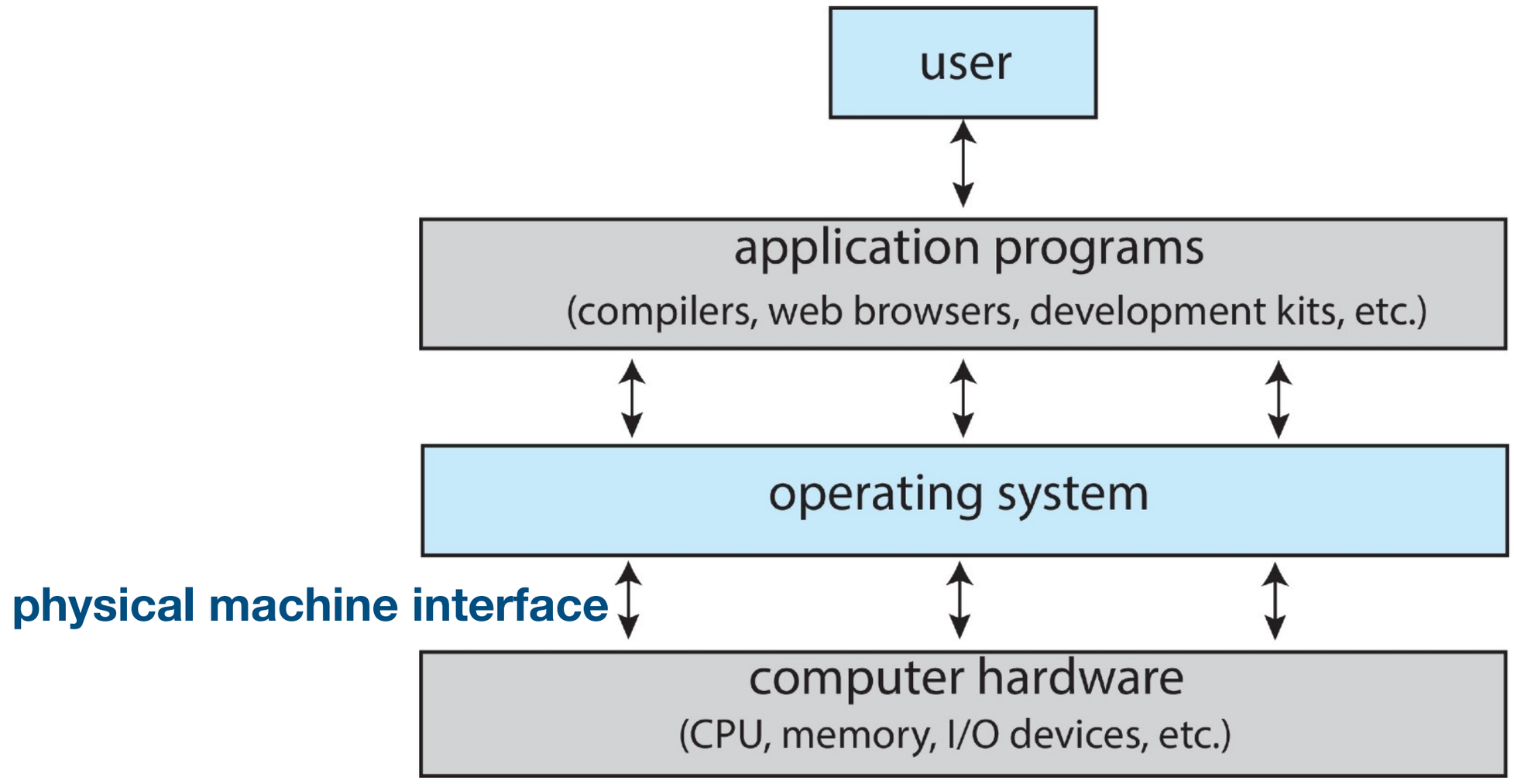
- Storage management
  - mounting and unmounting
  - free space management
  - disk scheduling
  - partitioning and protection
- I/O system management
  - buffering, caching, and spooling
  - device-driver interface
- File system management
  - creating and deleting files and directories
  - mapping files onto mass storage
  - backing up files on nonvolatile storage media

What's an Operating System?

- software that manages a computer's hardware and coordinates its use among various application programs

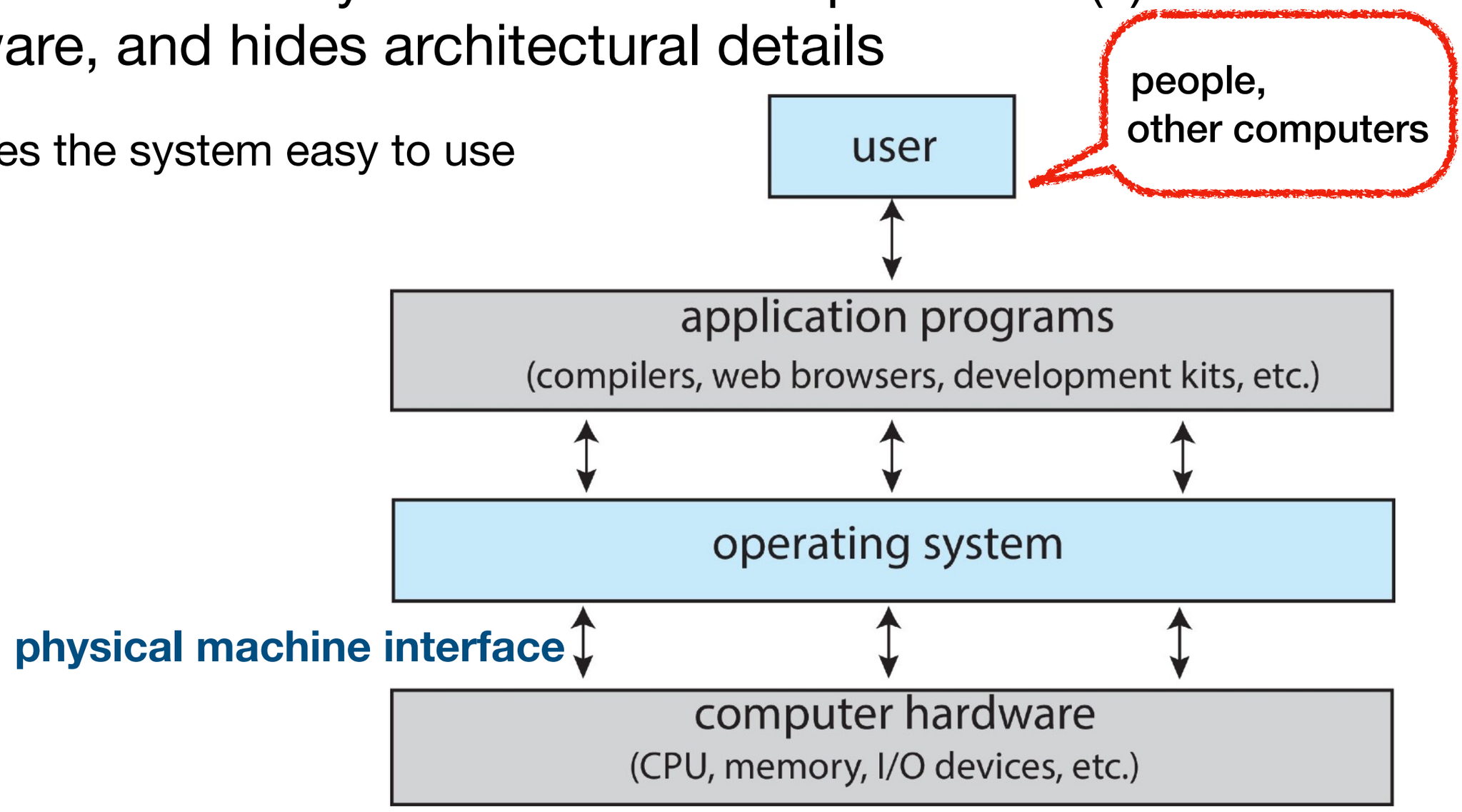
What's an Operating System?

- software that manages a computer's hardware and coordinates its use among various application programs



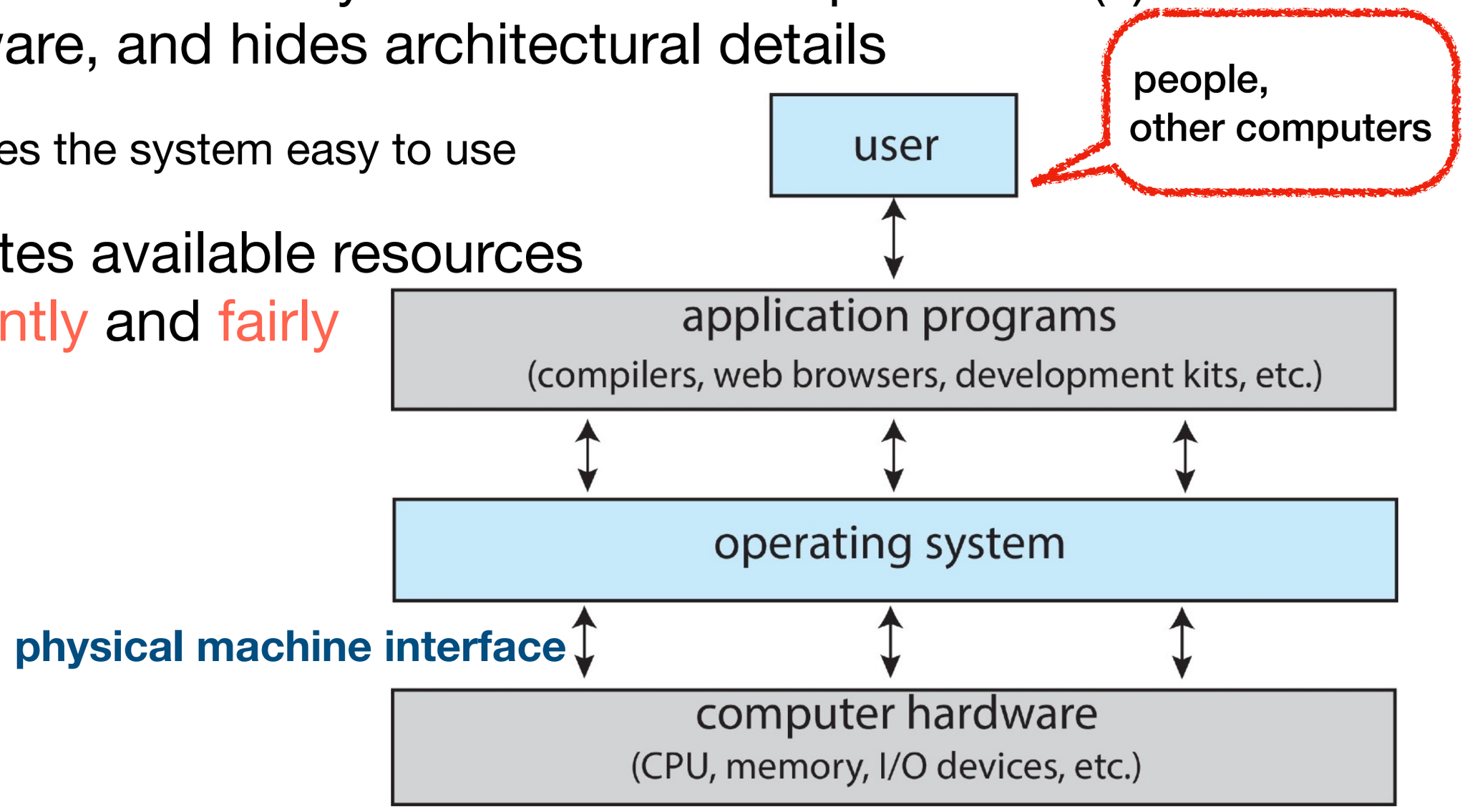
What's an Operating System?

- software that manages a computer's hardware and coordinates its use among various application programs
- acts as intermediary between the computer user(s) and hardware, and hides architectural details
  - makes the system easy to use



What's an Operating System?

- software that manages a computer's hardware and coordinates its use among various application programs
- acts as intermediary between the computer user(s) and hardware, and hides architectural details
  - makes the system easy to use
- allocates available resources efficiently and fairly



OS virtualizes hardware resources

- Definition: transforming a physical resource into a more general and powerful virtual form of itself

OS virtualizes hardware resources

- Definition: transforming a physical resource into a more general and powerful virtual form of itself
- Virtualizing CPU
  - allowing many programs to **seemingly** run at once on many **virtual** CPUs

OS virtualizes hardware resources

- Definition: transforming a physical resource into a more general and powerful virtual form of itself
- Virtualizing CPU
  - allowing many programs to **seemingly** run at once on many **virtual** CPUs
- Virtualizing Memory
  - each running program (i.e., **process**) accesses its own private address space; hence, a memory reference by one does not affect other ones
  - two running programs may perform an update at address 0x200000 independently!
  - OS maps these address spaces onto the physical memory

OS Roles

- OS as a referee
  - Allocates resource among users/applications
  - Isolates different users, applications from each other
  - Coordinates communication between users/applications

OS Roles

- OS as a referee
  - Allocates resource among users/applications
  - Isolates different users, applications from each other
  - Coordinates communication between users/applications
- OS as an illusionist
  - Gives the illusion of having infinite resources (virtual CPUs, memory, etc.) and reliable storage and network transport. Each application appears to a dedicated machine.
  - Masks physical limitations and details

OS Roles

- OS as a referee
  - Allocates resource among users/applications
  - Isolates different users, applications from each other
  - Coordinates communication between users/applications
- OS as an illusionist
  - Gives the illusion of having infinite resources (virtual CPUs, memory, etc.) and reliable storage and network transport. Each application appears to a dedicated machine.
  - Masks physical limitations and details
- OS as a glue
  - Provides an execution environment with a standard library to applications, …

What constitutes an Operating System?

- a core kernel
  - a program that runs at all times in the kernel model, aka supervisor mode or privileged mode
  - located and loaded into memory by the bootstrap program at system boot time

What constitutes an Operating System?

- a core kernel
  - a program that runs at all times in the kernel model, aka supervisor mode or privileged mode
  - located and loaded into memory by the bootstrap program at system boot time
- system programs/daemons
  - programs that are shipped with the operating system, but are not part of the kernel
  - In Linux the first system daemon is **`systemd`** which starts many other daemons

What constitutes an Operating System?

- a core kernel
  - a program that runs at all times in the kernel model, aka supervisor mode or privileged mode
  - located and loaded into memory by the bootstrap program at system boot time
- system programs/daemons
  - programs that are shipped with the operating system, but are not part of the kernel
  - In Linux the first system daemon is `systemd` which starts many other daemons
- middleware
  - a set of software frameworks that provide additional services to application developers such as databases, multimedia, graphics

Homework?

- Watch "A Narrative History of BSD" on YouTube