My Introduction

- Full name: Apostolos Lerios. Call me Tô Li (bowl and glass).
- Grew up in Greece, immigrated to US in 1989.
- BS, MS Computer Science from Stanford, CA. Not a Doctor after 5 years in PhD (computer graphics).
- Taught at Stanford, gifted high-school programs.
- Worked for NASA, Sun Microsystems and more. Joined four startups (founded one): one bankrupt, one struggling, one sold, one public.
- My wife, Christine, is Vietnamese-American.
- Enjoy being interrupted with questions.

Administrivia

- Academic honesty: collaborate to understand material, solve problems on your own.
- Disability services: see web.
- Course graded on curve. Final 20%, assignments 80%:
  - Four assignments total. Each 20%.
  - All include programming part in Java.
  - Last assignment includes group presentations: Windows 2000, Linux. Invite peers, friends, teachers.
  - Handed out M, due F when class starts.
  - Grace policy: 2 grace days (where *day* means MWF only).
- March 12 lecture is optional:
  - Life cycle of a US company. Or: From Ideas to Money.
  - Life in Texas. Or: From Bún to BBQ.
- Lecture slides on-line

Chapter 1: Introduction

- What is an Operating System?
- System types
- Computing Environments

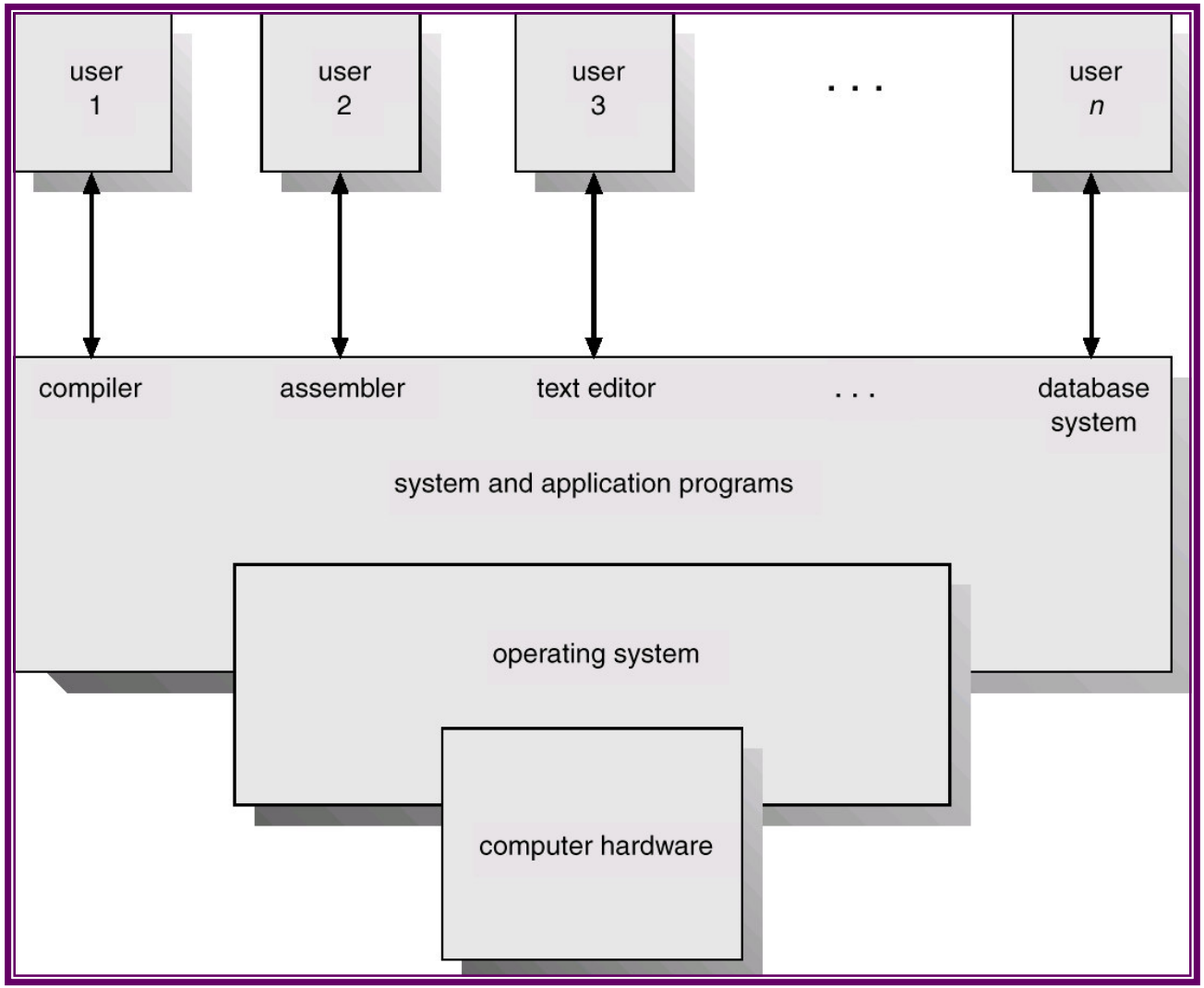
What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
- Use the computer hardware in an efficient manner.

Computer System Components

1. Hardware – provides basic computing resources (CPU, memory, I/O devices).
2. Operating system – controls and coordinates the use of the hardware among the various application programs for the various users.
3. Applications programs – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).
4. Users (people, machines, other computers).

Abstract View of System Components



Operating System Definitions

- Resource allocator – manages and allocates resources.
- Control program – controls the execution of user programs and operations of I/O devices .
- Kernel – the one program running at all times (all else being application programs).

What is not an Operating System?

- The commands/applications that query system status: ls (UNIX), Task Manager (Windows). These are tools and do not run all the time. Also, can emulate: Cygwin/MKS are not OS; they are UNIX-like toolsets for Windows.
- The drivers that come with peripherals: those enable new hardware to be used by OS, but are OS extensions, not the OS itself. Like browser plugin vs. browser itself.
- Software that enables access to hardware alone:
  - VMWare is a virtual PC (not hardware). Can install Windows "on" it.
  - Java VM is a *virtual* machine (not hardware), its API is a generalized OS. (Careful: JavaOS is a separate project.)
- Not free (Total Cost of Ownership), not perfect, not static.

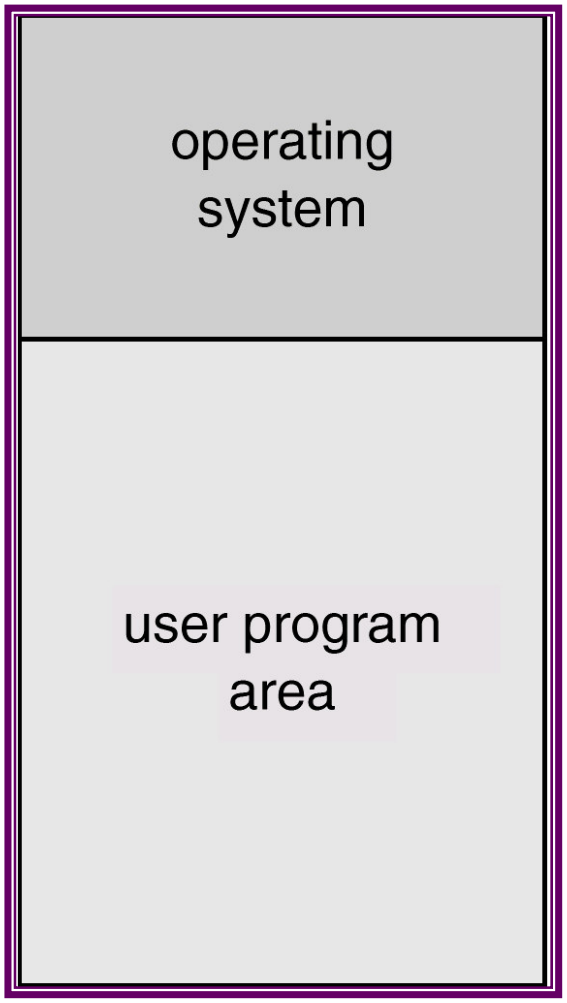
Why bother?

- OS design is excellent case study of software engineering design.
- The better you know the OS, the better apps you write, the better you understand its bugs and work around them.
- OS uses complex algorithms, many can be reused in non-OS software, e.g. deadlock detection.
- OS needs drive many hardware improvements.
- UHCL says so.

Mainframe Systems

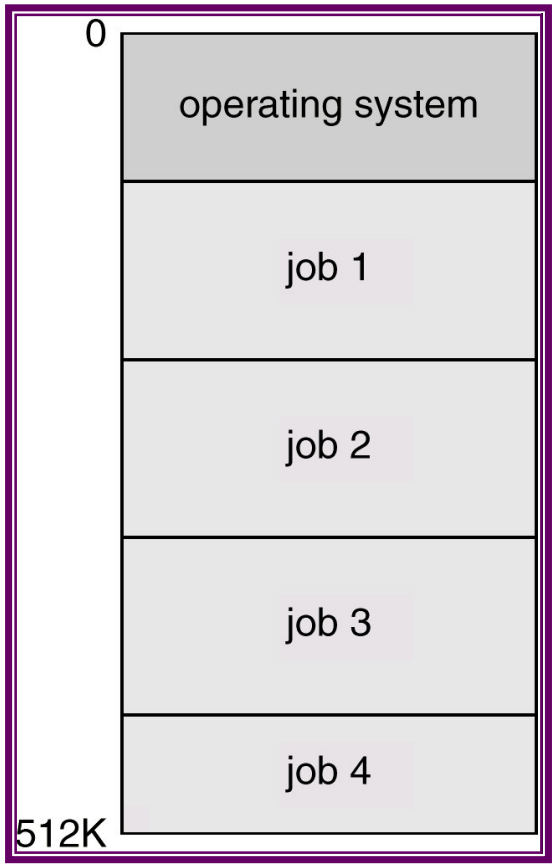
- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another. First rudimentary operating system.
- Resident monitor
  - initial control in monitor
  - control transfers to job
  - when job completes control transfers back to monitor

Memory Layout for a Simple Batch System



Multiprogrammed Batch Systems

Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.



OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

Time-Sharing Systems–Interactive Computing

- The CPU is multiplexed among several jobs (*processes*) that are kept in memory and on disk (the CPU is allocated to a job only if the job is in memory).
- A job swapped in and out of memory to the disk (*virtual memory*).
- On-line communication between the user and the system is provided; when the operating system finishes the execution of one command, it seeks the next “control statement” from the user’s keyboard.

Desktop Systems

- *Personal computers* – computer system dedicated to a single user.
- I/O devices – keyboards, mice, display screens, small printers.
- User convenience and responsiveness.
- Can adopt technology developed for larger operating system. But often individuals have sole use of computer so less emphasis on advanced CPU utilization, protection features.
- May run several different types of operating systems (Windows, MacOS, Linux, Solaris x86, BeOS)

Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of parallel system:
  - Increased *throughput* (goal is linear: N processors, N speedup)
  - Economical (due to shared resources)
  - Increased reliability (graceful degradation), a.k.a. fault tolerant system

Parallel Systems (Cont.)

- *Symmetric multiprocessing (SMP)*
  - Each processor runs and identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP.
- *Asymmetric multiprocessing*
  - Each processor is assigned a specific task; master processor schedules and allocated work to slave processors.
  - More common in extremely large systems or special-purpose systems (e.g. rendering).

Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses, telephone lines, Internet.
- Advantages of distributed systems.
  - Same as parallel systems but also…
  - … cheaper: off-the-shelf parts (Genetic Programming Inc. http://www.genetic-programming.com/)…
  - … duplication of data is implicit backup (reliability).
- But harder to manage than parallel systems: network failures, node failures, large cost of data transfer, etc.

Distributed Systems (cont)

- Requires networking infrastructure:
  - Local area networks (LAN)
  - Wide area networks (WAN)
- May be either
  - client-server (file/compute) or
  - peer-to-peer (workgroup)

Clustered Systems

- Clustering allows two or more systems to share storage.
- Provides high reliability.
- *Asymmetric clustering*: one server runs the application while other servers standby.
- *Symmetric clustering*: all N hosts are running the application.

Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well-defined fixed-time constraints.
- Real-Time systems may be either *hard* or *soft* real-time.

Real-Time Systems (Cont.)

- Hard real-time:
  - Secondary storage limited or absent, data stored in short term memory, or read-only memory (ROM).
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- Soft real-time:
  - Limited utility in industrial control of robotics.
  - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features.

Handheld Systems

- Personal Digital Assistants (PDAs)
- Cellular telephones
- Issues:
  - Limited memory
  - Slow processors
  - Small display screens.

Gaming systems

- Limited input: joystick, few buttons.
- Limited computer hardware.
- Emphasis on graphics, sound.
- User convenience and responsiveness.
- Networking needed for multi-user games.
- OS was simple; has evolved to desktop complexity.

Computing Environments

- Traditional computing: mainframes, desktops
- Web-Based Computing: “the network is the computer”
- Embedded Computing: usually special-purpose