What is an Operating System? (Review)

- An *operating system* (OS) is the interface between the user and the hardware
  - It implements a virtual machine that is easier to program than bare hardware
- An OS provides standard **services** (functionality) which are implemented on the hardware, including:
  - Processes, CPU scheduling, memory management, file system, networking
- The OS **coordinates** multiple applications and users (multiple processes) in a fair and efficient manner

↪The goal in OS development is to make the machine both **convenient** to use (a software engineering problem) as well as **efficient** (a system and engineering problem)

History of Operating Systems

- Phase 0 — hardware is a very expensive experiment; no operating systems exist
  1. One user at console
     - One function at a time (computation, I/O, user think/response)
     - Program loaded via card deck
       - Libraries of device drivers (for I/O)
     - User debugs at console
- Phase 1 — hardware is expensive, humans are cheap
  2. Simple batch processing: load program, run, print results, dump, repeat
     - User gives program (cards or tape) to the operator, who schedules the jobs
     - *Resident monitor* automatically loads, runs, dumps user jobs
     - Requires memory management (relocation) and protection
     - More efficient use of hardware, but debugging is more difficult (from dumps)

History of Operating Systems (cont.)

- Phase 1 — hardware is expensive, humans are cheap
  3. Overlapped CPU & I/O operations
     - First: buffer slow I/O onto fast tape drives connected to CPU, replicate I/O devices
     - Later: *spool* data to disk
  4. Multiprogrammed batch systems
     - Multiple jobs are on the disk,waiting to run
     - *Multiprogramming* — run **several** programs at the "same" time
       - Pick some jobs to run (*scheduling*), and put them in memory (*memory management*)
       - Run one job; when it waits on something (tape to be mounted, key to be pressed), switch to another job in memory
     - First big failures:
       - MULTICS announced in 1963, not released until 1969
       - IBM's OS/360 released with 1000 known bugs
     - OS design should be a science, not an art

History of Operating Systems (cont.)

- Phase 2 — hardware is less expensive than before, humans are expensive
  5. Interactive *timesharing*
     - Lots of cheap terminals, one computer
       - All users interact with system at once
       - Debugging is much easier
     - Disks are cheap, so put programs and data online
       - 1 punch card = 100 bytes
       - 1MB = 10K cards
       - OS/360 was several feet of cards
     - New problems:
       - Need *preemptive scheduling* to maintain adequate *response time*
       - Need to avoid *thrashing* (swapping programs in and out of memory too often)
       - Need to provide adequate security measures
     - Success: UNIX developed at Bell Labs so a couple of computer nerds (Thompson, Ritchie) could play Star Trek on an unused PDP-7 minicomputer

History of Operating Systems (cont.)

- Phase 3 — hardware is cheap, humans are expensive
  6. Personal computing
     - CPUs are cheap enough to put one in each terminal, yet powerful enough to be useful
       - Computers for the masses!
     - Return to simplicity; make OS simpler by getting rid of support for multiprogramming, concurrency, and protection
  7. Parallel systems
     - User multiple CPUs with a shared memory in close communication
       - Increased throughput
     - Mostly MIMD hardware, some SIMD
     - Symmetric multiprocessing (SMP)
       - Each processor runs an identical copy of the OS, multiple processes running at once

History of Operating Systems (cont.)

- Phase 4 — hardware is cheap, ubiquitous, and pervasive
  6. Distributed systems
     - Distribute the computation among several (possibly different) physical processors, each with its own memory, in loose communication
       - Resource sharing
       - Increased throughput
       - Reliability
     - Client-server computing
       - Server provides specified services (e.g., file service, directory service, print service) to a set of clients
  7. Embedded / handheld systems
     - PDAs, cell phones, CD players, etc.
     - Small OS "footprint" (limited memory)
     - Slow processors
     - Small displays
     - Power consumption is a primary consideration

Modern OS Functionality (Review)

- Textbook talks about OS as a:
  - Control program — manages the execution of user programs, prevents errors and improper use of the computer
  - Resource allocator — CPU time, memory space, file space, I/O devices
- OS must provide:
  - Processes & CPU scheduling
    - Multiple processes active concurrently
    - Processes may need to communicate
    - Processes may require mutually-exclusive access to some resource
  - Memory management — must allocate memory to processes, move processes between disk and memory
  - File system — must allocate space for storage of programs and data on disk