CSc 360
Operating Systems
OS Structures

Jianping Pan

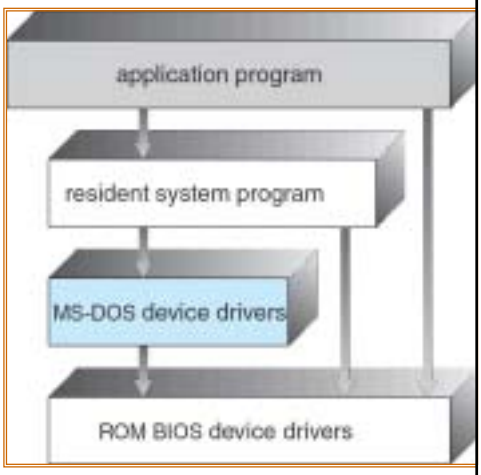
Summer 2006

OS design and implementation

- An art of balance
  - hardware vs software
    - efficiency vs flexibility
  - user vs system
    - convenience vs effectiveness
- General design guidelines
  - separation of mechanisms and policies
- Best current practices

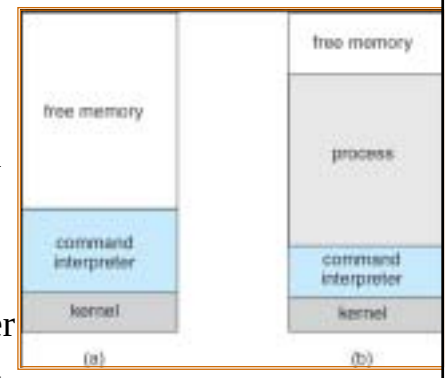
Simple structure

- E.g., MS-DOS
  - single user
  - almost single process
    - direct access
  - almost flat memory
    - MZ linked list
  - executables
    - .COM: segment limit
    - .EXE: MZ file magic



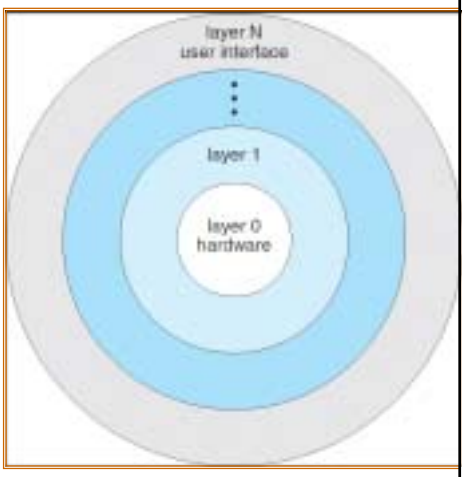
MS-DOS

- Load program
  - "shrink" interpreter
  - make room for program
- Execute program
  - access to everywhere
  - even "kernel"/interpreter
- Reload interpreter back
  - otherwise, "cannot find command.com…"



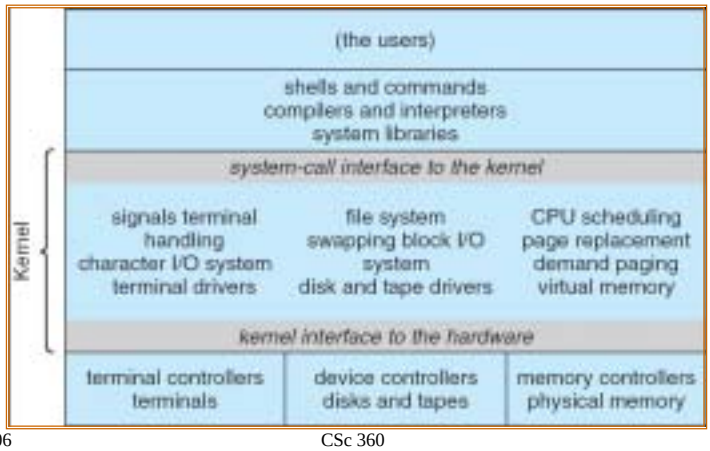
Layered structure

- Layers
  - L_0: hardware
  - L_N: user interface
  - L_i: anything in btw
    - use L_{i-1} service
    - offer service to L_{i+1}
- Divide & conquer
- Cross-layer issues



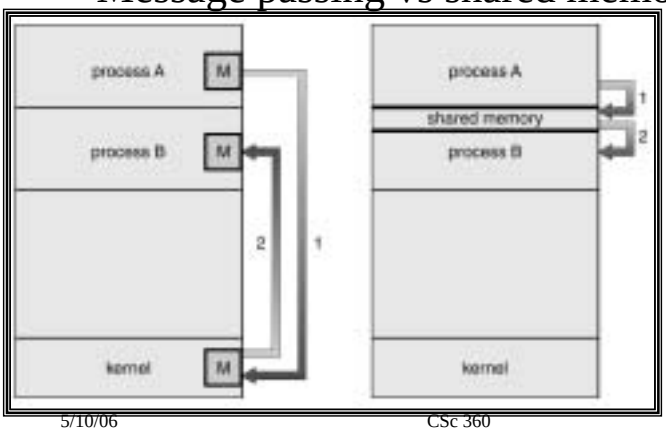
Unix

- Hybrid structure



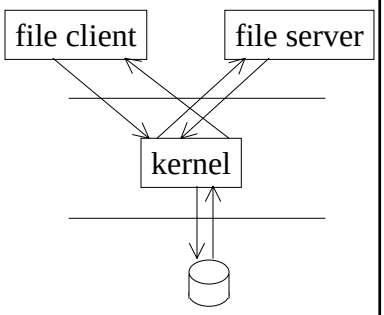
Multiprogramming

- Message passing vs shared memory



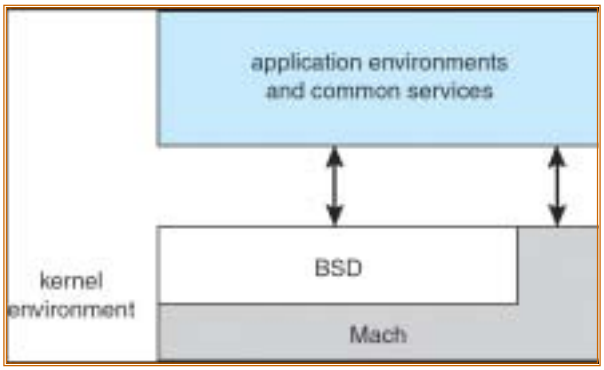
Micro-kernel structure

- E.g.
  - Mach
- Smaller kernel
  - only those “essentials”
  - e.g., handle hardware
- More by system/application programs
  - message passing
- Overhead between kernel and user spaces



Mac OS X

- Mach (CPU,memory) + BSD (file,network)

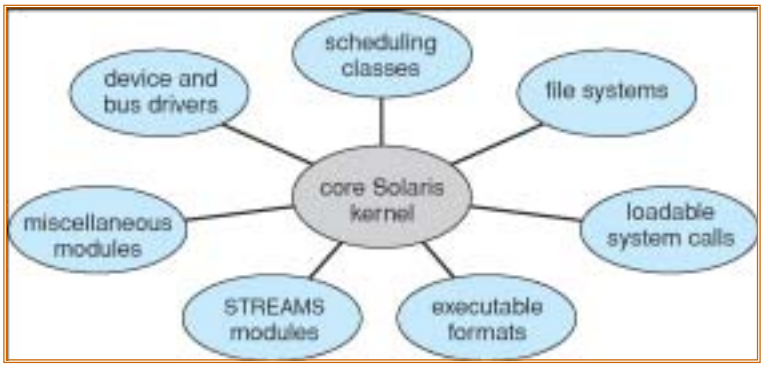


Modular structure

- Object-oriented methodology
  - not necessary implemented in OO languages
  - popular choices for modern OS, e.g., Linux
  - e.g., insmod fat|vfat|msdos
- On-demand, loadable kernel modules
  - each module is a separate function/support
  - communicate through know kernel interface
  - module dependency

SunOS Solaris

- Modular design (high-level diagram)



This lecture

- OS structures
  - design and implementation tradeoffs
    - user requirement
    - hardware support
  - layered, micro-kernel, modular
    - pros and cons
- Explore further
  - which OS structures are good for embedded system, I/O or computation-intensive system?

Next lecture

- Virtual machines vs real machines
  - machine virtualization
  - from power-on to login
  - read OSC7 Chapter 2 (or OSC6 Chapter 3)