Introduction

CS 256/456

Dept. of Computer Science, University of Rochester

General Course Information

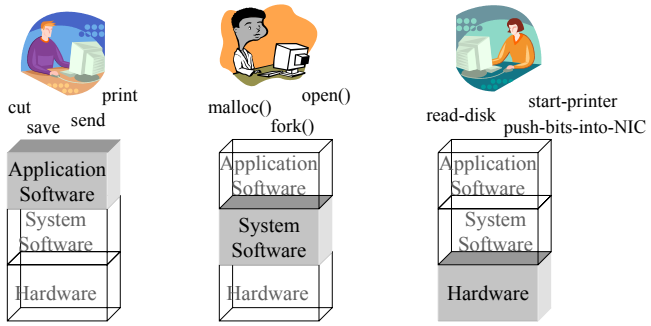
- Course Web page:
  - www.cs.rochester.edu/~kshen/csc256-spring2007
- Course-related announcement/correspondence:
  - Broadcast email: cs256@cs.rochester.edu
- Texts
  - Tanenbaum, "Modern Operating Systems"
  - Silberschatz et al, "Operating System Concepts"

General Course Information (cont.)

- Assignments and grading
  - five/six programming assignments (total 60%)
  - midterm (15%) and final (25%)
  - extra 5% for participation in class discussions
  - maybe two written assignments (given to you to help preparing for exams, not graded)
- "CSC456 Part" in assignments
- C programming
- End-of-term survey paper for CSC456 students

What is an Operating System?

Perspectives of the Computer



(a) End User View

(b) Application Programmer View

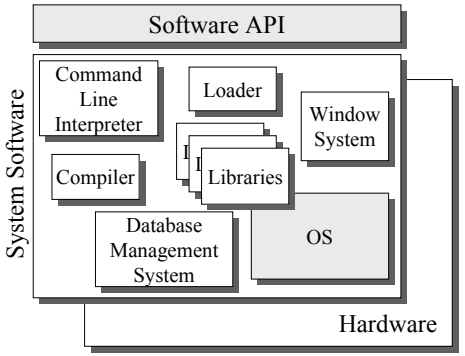
(c) System Programmer View

System Software

- A general piece of software with common functionalities that support many applications
- Examples
  - C compiler and library functions
  - Shell – command line interpreter
  - A window system
  - A database management system
  - The operating system
    - A thin layer of software that operates directly on the raw hardware

The Structure of Computer System

Application Programmer



What does an Operating System do?

- It is an extended machine
  - Hides the messy details about hardware resources
  - Presents users with a resource abstraction that is easy to use
- It is a resource manager
  - Allows multiple users/programs to share resources and coordinate the sharing

Resource Abstraction

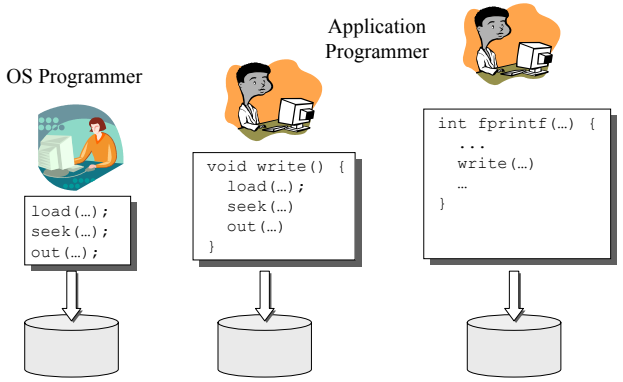
```
load(block, length, device);
seek(device, track);
out(device, sector)
```

```
write(char *block, int len, int device,
                   int track, int sector) {
   ...
  load(block, length, device);
  seek(device, track);
  out(device, sector);
   ...
}
```

```
write(char *block, int len, int device,int addr);
```

```
fprintf(fileID, "%d", datum);
```

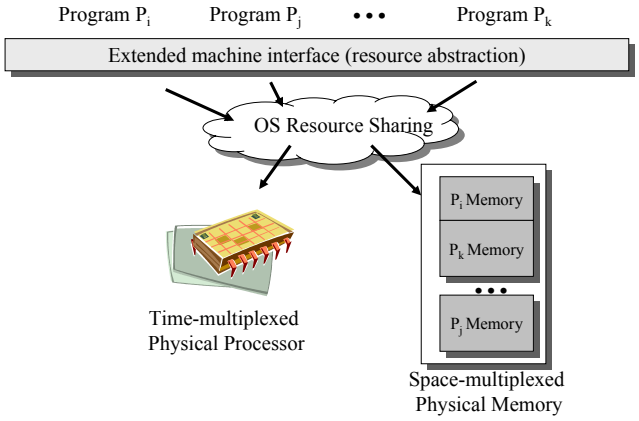
Disk Abstractions



Under the Abstraction

- functional complexity
- a single abstraction over multiple devices
- replication → reliability

Resource Sharing



Objectives of Resource Sharing

- efficiency
- fairness
- security

Specific Types of OS

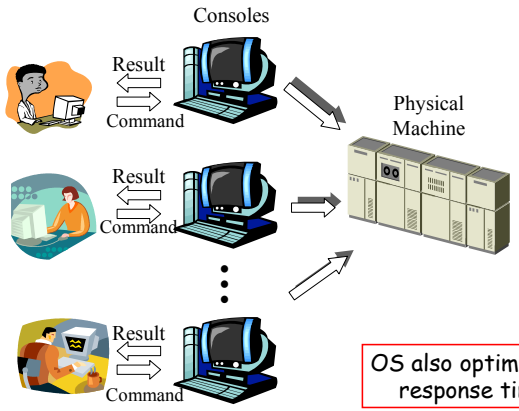
- Batch processing
- Timesharing OS
- Real-time OS
- Distributed OS
- Small computers

Batch Processing

- *Job* (sequence of OS commands) prepared offline
- Batch of jobs given to OS at one time
- OS processes jobs one-after-the-other
- No human-computer interaction
- Batch processing still used today (e.g., for running compute-intensive simulations)

OS optimizes for high processing throughput.

Timesharing Systems

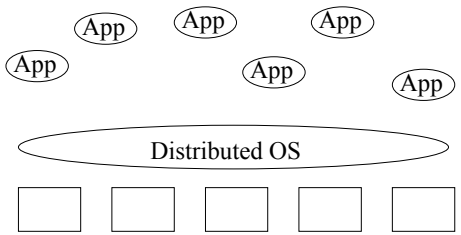


OS also optimizes for short response time.

Real-Time Systems

- Must respond to external events in fixed time (e.g., industrial process control) – hard real-time
- Continuous media popularizing real-time techniques – soft real-time

Distributed OS



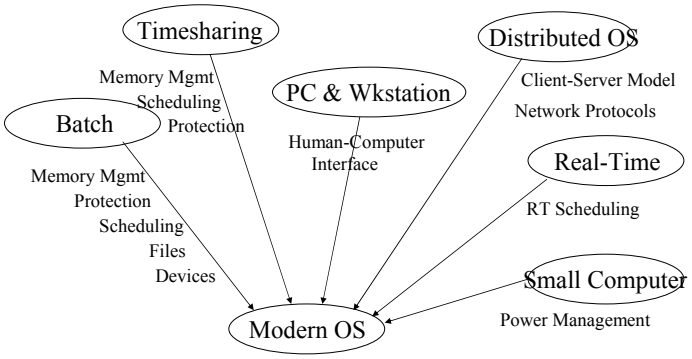
Multiple Computers connected by a Network

More issues: distributed coordination, fault tolerance, security, …

Small Computers

- PDAs, embedded systems, computer sensors became more significant
- Have an OS, but
  - Not general purpose
  - Limited hardware resources (an OS in 4KB memory!) and power
  - PalmOS, Pocket PC (WinCE), VxWorks, TinyOS, …

Evolution of Modern OS



Examples of Modern OS

- UNIX variants (e.g., Solaris, Linux) -- have evolved since 1970
- Windows NT/2K -- has evolved since 1989 (much more modern than UNIX
- Research OSes –
  - microkernel
  - extensible OS
  - virtual machines
  - sensor OS
  - special-purpose OS – for highly concurrent Internet servers
  - still evolving …

Why Study Operating Systems?

- Learn to design an OS or other computer systems
- Understand an OS
  - Understand the inner working of an OS
  - Enable you to write efficient/correct application code

Assignment #1

- Exclusively outside of the OS
- Part I: observing the OS through the /proc virtual file system
- Part II: building a shell (command-line interpreter)
  - Support foreground/background executions
  - (CSC456 Part) Support pipes

Disclaimer

- Parts of the lecture slides contain original work of Gary Nutt and Andrew S. Tanenbaum. The slides are intended for the sole purpose of instruction of operating systems at the University of Rochester. All copyrighted materials belong to their original owner(s).