Today's topics

- **Performance**
  - Notes from David A. Patterson and John L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, Morgan Kaufmann, 1997.
- **Operating Systems**
  - Brookshear: Chapter 3
  - Notes from Kevin Wayne's COS 126 course at Princeton
- **Upcoming**
  - Security
    - GI, 11 & Brookshear 3.7

Performance

- *Performance* = 1/*Time*
- The goal for all software and hardware developers is to increase performance
- Metrics for measuring performance (pros/cons?)
  - Elapsed time
  - CPU time
    - Instruction count (RISC vx. CISC)
    - Clock cycles per instruction
    - Clock cycle time
  - MIPS vs. MFLOPS
  - Throughput (tasks/time)
  - Other more subjective metrics?
- What kind of workload to be used?
  - Applications, kernels and benchmarks (toy or synthetic)

What is an Operating System?

Quotes from Microsoft trial.

Dave Farber [DOJ witness]:
- "An Operating System is software that controls the execution of programs on computer systems and may provide low-level services such as resource allocation, scheduling and input-output control in a form which is sufficiently simple and general so that these services are broadly useful to software developers."

Ed Felten [DOJ witness]:
- "An operating system is software that provides services relating to booting the machine and starting programs, interfacing with the hardware, managing and scheduling use of hardware resources, and basic security services."

What is an Operating System?

Jim Allchin [Microsoft witness] :
- "A computer operating system, like any other type of software, is a set of instructions that causes a computer to carry out specified functions. Although no clear line of demarcation exists between the functions performed by operating system software and other types of software, operating systems generally serve, at a bare minimum, as the computer's 'central nervous system,' scheduling the execution of tasks by the central processing unit and controlling the flow of information within the computer and between the computer and any peripheral devices that may be attached to it. It is important to bear in mind that all software is a series of instructions to a computer, and terms that have evolved to categorize such software are merely descriptive of general categories of functionality. There is no widely accepted definition of an operating system; it is a concept that has evolved over time based on what is technically possible and what customer[s] have said they want."
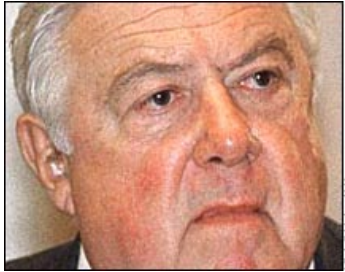
What is an Operating System?

Judge Jackson's ruling:

- "An 'operating system' is a software program that controls the allocation and use of computer resources (such as central processing unit time, main memory space, disk space, and input/output channels). The operating system supports the functions of software programs, called 'applications,' that perform specific user-oriented tasks."

What is an Operating System?

Modern operating systems support:

- Software tools for creating programs
  - libraries, compilers
- Running multiple programs.
  - multiprogramming
- Saving/accessing data.
  - files, virtual memory
- User interaction.
  - window system
- Interaction with other systems.
  - networking
- Core applications programs.
  - client-server

What is an Operating System?

Execution Control.

- OS keeps track of state of CPU, devices.

External Devices.

- Display, keyboard, mouse, disks, CD, network.

Virtual Machines.

- Pretend machines that each person/program can use.
- OS implements abstract devices.

Multiprogramming

Operating system "kernel" keeps track of several programs.

- CPU does 1 thing at a time.
- Goal: illusion of multiple machines.

INTERRUPT:

- Part of hardware of real machines
  - stop
  - save Program Counter (PC) somewhere "special"
  - change PC
- Necessary to manage input-output devices.
  - mouse click, keyboard
- OS allows several programs/processes to "share" CPU by keeping table of "current" PC's for programs setting clock to interrupt periodically.
  - Context switching
  - round-robin or user priorities

Virtual Memory

Problem 1: several programs need to share same memory.
- **Direct solution: apportion up the memory.**

Problem 2: program needs more memory than machine has.
- **Direct solution: "overlays."**
  - **program shuffles its own data in and out of memory to disk**

It's all just memory, why should file system look more complicated?

"Better" solution: VIRTUAL MEMORY (1960's).
- **All programs assume access to all memory.**
- **Each program actually uses a small portion.**

Size of Virtual Memory

How many bits is enough?
- **16 bits is not enough.**
- **32 bits is not enough.**
- **64 bits?**
  - 2^{64} **= 18,446,744,073,709,551,616 > 10^{19} addresses**
- **512 certainly enough.**

Some big numbers.
- 2^{70}**: number of grains of sand on beach at Coney Island.**
- 2^{93}**: number of oxygen atoms in a thimble.**
- 2^{256}**: number of electrons in the universe.**

More sophisticated paging strategies needed.

Paging

Each page brought in has to REPLACE another.
- **Page replacement strategies.**
  - **Ex. least recently used**
- **Still being studied, invented.**

Basic principles.
- **MEMORY HIERARCHY**
  - **local: fast, small, expensive**
  - **remote: slow, huge, cheap**
- **Tradeoff speed for cost.**
- **CACHE recently accessed information.**

Window Manager

Virtual Terminals.
- **Each program has its own virtual display.**
- **Ex. X-terminal: complex, customizable, virtual!**
- **Just another simulation program.**
- **Commonplace today, rare in 1985**
- **Ingenious design meets accelerating technology.**

History.
- **Xerox PARC (Alto), Macintosh, Windows NT, X-terminal, Netscape.**

Problem or opportunity?
- **Truly "virtual."**
- **Moving away from grounding in reality.**
  - **harder for programmers to understand what is happening**
- **Flexibility vs. standardization.**
- **Other ways of interacting with computer?**

Client-Server Model

System divided into two distinct parts.
- **Ex: display server (implement virtual display).**
  - – draw stuff on screen
  - – monitor keyboard and mouse input
- **Ex: Client (use virtual display).**
  - – applications programs

Server is interface between client program and display hardware.

Model generalizes beyond display management.
- **Client: request service.**
- **Server: do the work.**

Advantages.
- **Single server can handle multiple clients.**
- **Keeps kernel simple, adaptable.**
- **Smooth transition to DISTRIBUTED SYSTEM.**

The Network

"Ultimate" distributed system.

INTERNET
- **"All the cooperating networks."**

Circuit switched network
- **Phone system.**

Packet switched network
- **Network system.**

IP: Internet protocol.
- **Packet.**
  - – 1-1500 bytes
  - – from address
  - – to address
- **Address.**
  - – Ex. 152.3.140.129

ROUTERS
- **Move packets across network.**

TCP: Transmission control protocol.
- **Break big messages into packets.**
- **Collect received packets into messages.**
- **Check for errors.**

Domain Name System.
- **Distribute authority/responsibility for name service.**
- **Can use "www.duke.edu" instead of 152.3.233.7.**

(many details omitted!)

Operating System / Network Issues

Network applications.
- **Communication (mail, news).**
- **Remote login (telnet).**
- **File transfer (ftp, Napster, Gnutella).**
- **Publishing (html).**
- **Browsing (Netscape, IE).**
- **E-commerce.**

Modern rendition of ancient tradeoffs.
- **Personal computer or Network computer.**
- **ONE huge virtual machine?!?**

Compare/contrast.
- **Computer center, phone system, Post office (snail mail), Libraries.**

Current network ethics:
- **Honor and foster individualism.**
- **Network is good and must be preserved.**

Should hackers or the government "run" the net?
- **Can commercial apps trust an "open net"?**
- **Does a "closed net" violate individual rights?**

Security/Privacy/Copyright.

Who owns? Who pays?

Unix File System Layout

Goal: provide simple abstraction (sequence of bytes) for user programs.

Each disk has:
- **I-nodes (one per file).**
  - – indexing information
  - – pointers to disk blocks
- **Data blocks.**
  - – just data

Superblock (block 1).
- **Catalog of disk layout.**
- **Size and number of data blocks.**
- **Size and number of i-nodes.**
- **Free list of data blocks.**

File.
- **List of data blocks.**

Directory.
- **List of file names.**
- **i-node addresses**

Forms a TREE structure.
- **Traverse the tree for sequential access.**