CS162
Operating Systems and Systems Programming
Lecture 1

Overview

January 22, 2014
Anthony D. Joseph
http://inst.eecs.berkeley.edu/~cs162

Who Are We?

- Anthony D. Joseph
  - 465 Soda Hall (RAD Lab)
  - Web: http://www.cs.berkeley.edu/~adj/
  - Office hours: Mondays/Tuesdays 10-11AM in 449 Soda
- Research areas:
  - Current: Cloud computing (Mesos), Secure Machine Learning (SecML), DETER security testbed
  - Other: Peer-to-Peer networking (Tapestry), Mobile computing, Wireless/Cellular networking

Who Are We? (cont'd)

- TA: Nicholas Chang
  - E-mail: cs162-ta@cory.eecs.berkeley.edu
  - Sections: Th 2-3p and 3-4p
  - Office hours: Fr 1-3 in 751 Soda

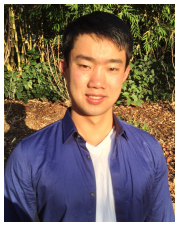
- TA: Kelvin Chou
  - E-mail: cs162-tb@cory.eecs.berkeley.edu
  - Sections: Th 11a-12p and 4-5p
  - Office hours: Tu 5-6 in 651 Soda,
    W 1-2 in 751 Soda

Who Are We? (cont'd)

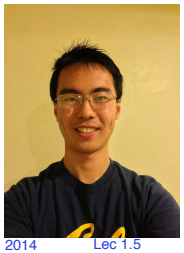
- TA: Riyaz Faizullabhoy
  - E-mail: cs162-td@cory.eecs.berkeley.edu
  - Sections: Th 10-11a and 1-2p
  - Office hours: M 1-2 in 611 Soda, M 2-3 in 411 Soda

- TA: Vaishaal Shankar
  - E-mail: cs162-tc@cory.eecs.berkeley.edu
  - Sections: Th 5-6p and 6-7p
  - Office hours: M 12-1 in TBD, Tu 4-5 in 751 Soda

Who Are We? (cont'd)



- TA: Isaac Tian
  - E-mail: cs162-te@cory.eecs.berkeley.edu
  - Sections: Fr 2-3p and 3-4p
  - Office hours: M 3-4 in 751 Soda,
    W 10-11 in 751 Soda



- George Yiu
  - E-mail: cs162-tf@cory.eecs.berkeley.edu
  - Sections: TBD
  - Office hours: TBD

Who Are We? (cont'd)

- Reader: Jiawen Xu

- Reader: William Liu

Goals of Today Lecture

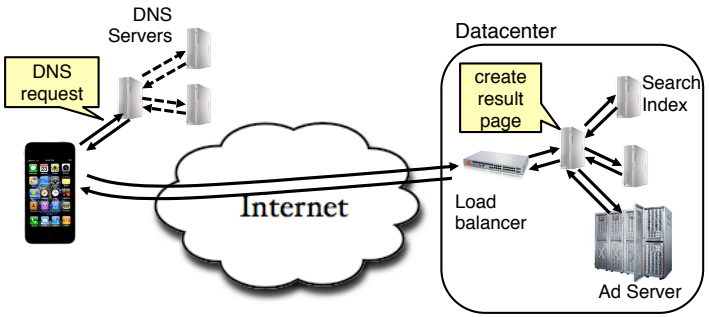
- What are we going to learn and why?
- What is an operating system?
- How does this class operate?
- Interactivity is important! Please ask questions!

Note: Some slides and/or pictures in the following are adapted from slides ©2005 Silberschatz, Galvin, and Gagne. Slides courtesy of Anthony D. Joseph, John Kubiatowicz, John Canny, AJ Shankar, George Necula, Alex Aiken, Eric Brewer, Ras Bodik, Ion Stoica, Doug Tygar, and David Wagner.

Goal of This Course

- Learn how "systems" work
- Main challenges in building systems
- Principles of system design, i.e., how to address these challenges
- Learn how to apply these principles to building systems

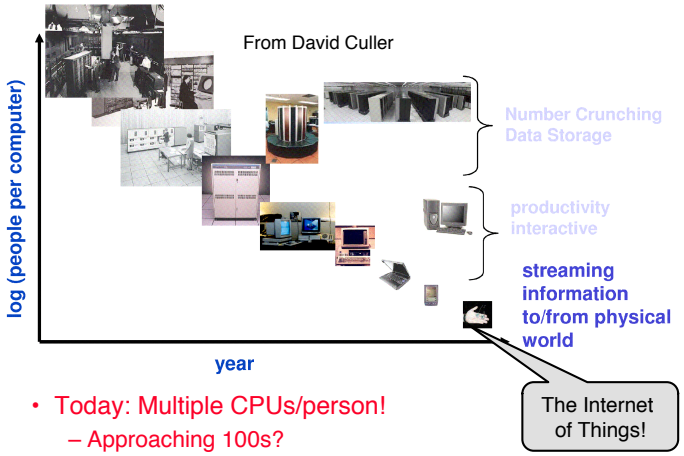
Example: What's in a Search Query?



- Complex interaction of multiple components in multiple administrative domains

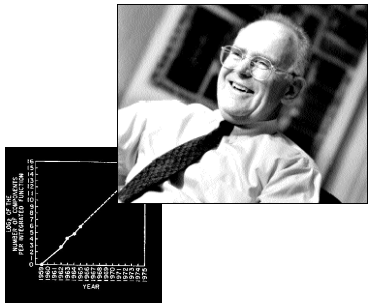
Computing Devices Everywhere

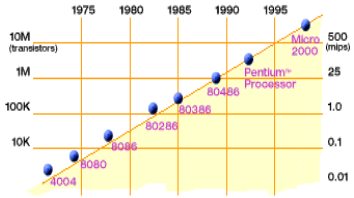
People-to-CPUs Ratio Over Time



- Today: Multiple CPUs/person!
  - Approaching 100s?

Technology Trends: Moore's Law



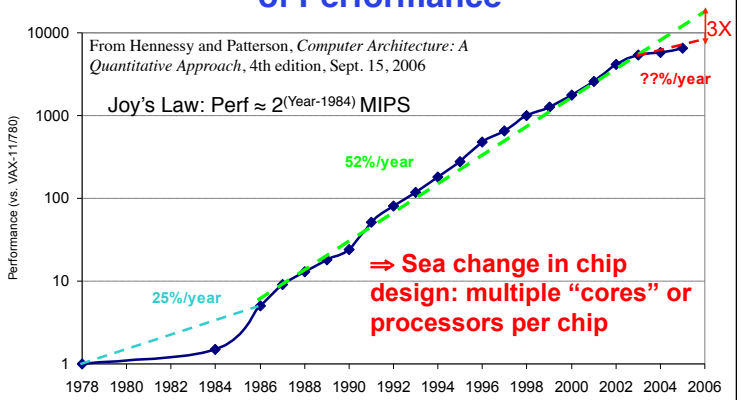


2X transistors/Chip Every 1.5 years
Called "Moore's Law"

Gordon Moore (co-founder of Intel) predicted in 1965 that the transistor density of semiconductor chips would double roughly every 18 months.

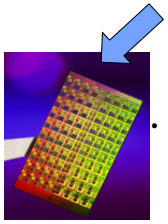
Microprocessors have become smaller, denser, and more powerful.

New Challenge: Slowdown in Joy's law of Performance

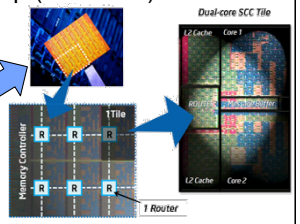


- VAX : 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2002
- RISC + x86: ??%/year 2002 to present

ManyCore Chips: The future is here



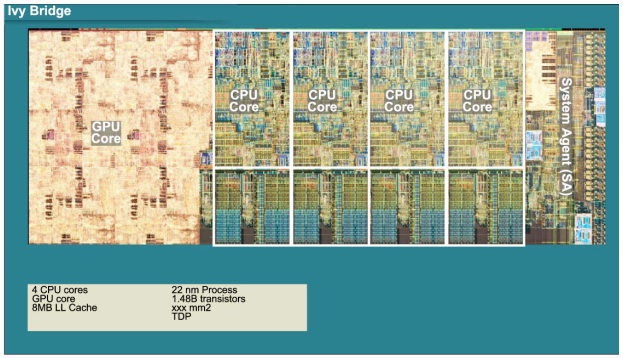
- Intel 80-core multicore chip (Feb 2007)
  - 80 simple cores
  - Two FP-engines / core
  - Mesh-like network
  - 100 million transistors
- Intel Single-Chip Cloud Computer (August 2010)
  - 24 "tiles" with two cores/tile
  - 24-router mesh network
  - 4 DDR3 memory controllers
  - Hardware support for message-passing



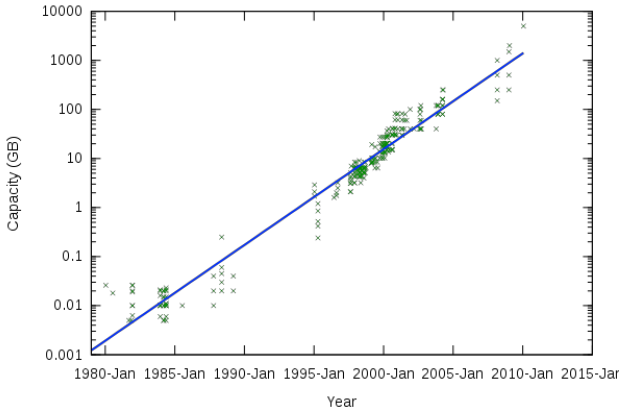
- "ManyCore" refers to many processors/chip
  - 64? 128? Hard to say exact boundary
- How to program these?
  - Use 2 CPUs for video/audio
  - Use 1 for word processor, 1 for browser
  - 76 for virus checking???
- Parallelism must be exploited at all levels

Ivy Bridge die photo: The world is parallel!

- Intel Xeon E3 – 4 cores (8 hardware threads)
  - 22 nm process, 160 mm die, 1.4 billion transistors

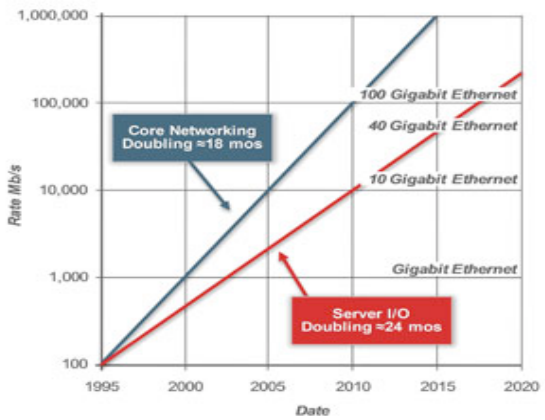


Storage Capacity



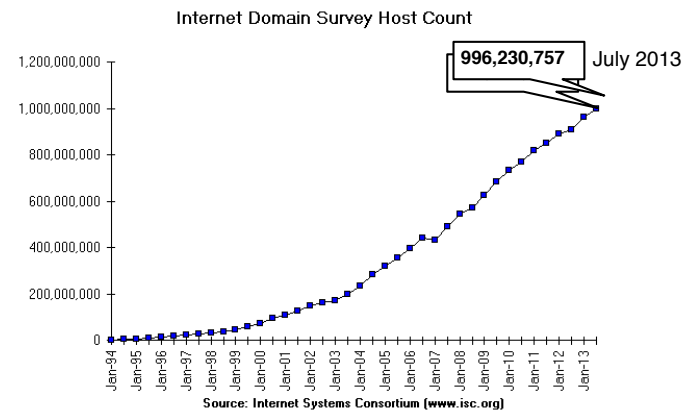
- *Retail* hard disk capacity in GB
  (source: http://www.digitaltonto.com/2011/our-emergent-digital-future/)

Network Capacity



(source: http://www.ospmag.com/issue/article/Time-Is-Not-Always-On-Our-Side)

Internet Scale: .96 Billion Hosts



https://www.isc.org/solutions/survey

Internet Scale: Almost 2.5 Billion Users!

WORLD INTERNET USAGE AND POPULATION STATISTICS June 30, 2012						
World Regions	Population (2012 Est.)	Internet Users Dec. 31, 2000	Internet Users Latest Data	Penetration (% Population)	Growth 2000-2012	Users % of Table
Africa	1,073,380,925	4,514,400	**167,335,676**	15.6 %	3,606.7 %	7.0 %
Asia	3,922,066,987	114,304,000	**1,076,681,059**	27.5 %	841.9 %	44.8 %
Europe	820,918,446	105,096,093	**518,512,109**	63.2 %	393.4 %	21.5 %
Middle East	223,608,203	3,284,800	**90,000,455**	40.2 %	2,639.9 %	3.7 %
North America	348,280,154	108,096,800	**273,785,413**	78.6 %	153.3 %	11.4 %
Latin America / Caribbean	593,688,638	18,068,919	**254,915,745**	42.9 %	1,310.8 %	10.6 %
Oceania / Australia	35,903,569	7,620,480	**24,287,919**	67.6 %	218.7 %	1.0 %
WORLD TOTAL	**7,017,846,922**	**360,985,492**	**2,405,518,376**	**34.3 %**	**566.4 %**	**100.0 %**

NOTES: (1) Internet Usage and World Population Statistics are for June 30, 2012. (2) CLICK on each world region name for detailed regional usage information. (3) Demographic (Population) numbers are based on data from the US Census Bureau and local census agencies. (4) Internet usage information comes from data published by Nielsen Online, by the International Telecommunications Union, by GfK, local ICT Regulators and other reliable sources. (5) For definitions, disclaimers, navigation help and methodology, please refer to the Site Surfing Guide. (6) Information in this site may be cited, giving the due credit to www.internetworldstats.com. Copyright © 2001 - 2013, Miniwatts Marketing Group. All rights reserved worldwide.

(source: http://www.internetworldstats.com/stats.htm)

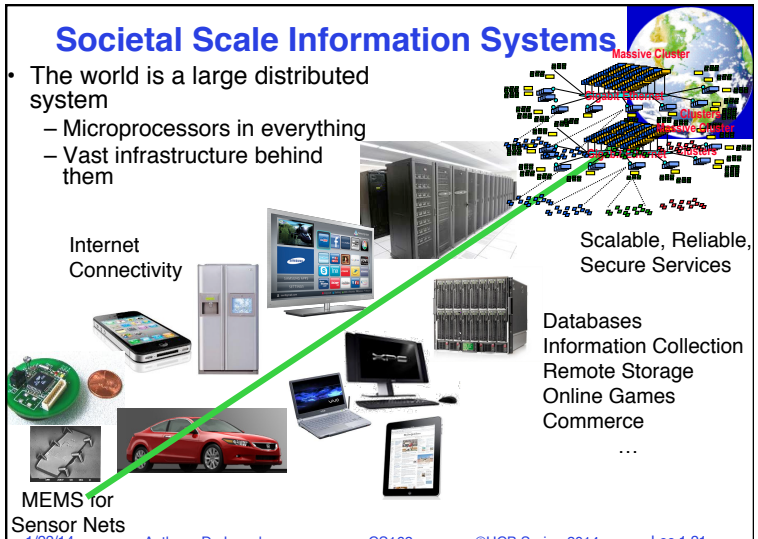
Not Only PCs connected to the Internet

- Smartphone shipments now exceed PC shipments!
- 2011 shipments:
  - 487M smartphones
  - 414M PC clients
    - » 210M notebooks
    - » 112M desktops
    - » 63M tablets
  - 25M smart TVs

- 4 billion phones in the world → smartphone over next decade

Societal Scale Information Systems

- The world is a large distributed system
  - Microprocessors in everything
  - Vast infrastructure behind them



Class Schedule & Info

- Class Time: MW 4-5:30pm, 245 Li Ka Shing
  - Please come to class; best part of class is interaction!
  - 5% of grade is participation (section, class, Piazza)
  - Waitlist processing is done by the department, not instructors
- Sections
  - Important information is in the sections
  - The sections assigned to you by Telebears are temporary!
  - Every member of a project group must be in same section
  - *Attend your preferred section this week*
- Website: http://www-inst.eecs.berkeley.edu/~cs162/
- Newsgroup: http://www.piazza.com/berkeley/spring2014/cs162/
- **TBD:** Time-delayed Webcast (audio podcast with slides)

Lecture Goal

Interactive!!!

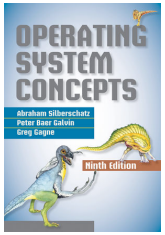
Syllabus

- Emphasize: not only single node OS, but also end-to-end system design
  - Some networking, database, and security concepts
  - Projects to reflect this emphasis
- Long term plan: make CS 162 a gateway course for
  - Database class (CS 186)
  - Security class (CS 161)
  - Software engineering class (CS 169)
  - Networking class (EE 122)
  - New advanced OS class (CS 194-24) – Advanced Operating Systems Structures and Implementation

Syllabus (cont'd)

- 14 lectures on core OS
- 3 lectures on Networking
- 2 lectures on Databases
- 2 lectures on Security
- 1 lecture on Software Engineering
- Capstone lecture on "putting everything together"

Textbook



- Textbook: Operating Systems Concepts, 9th Edition Silbershatz, Galvin, Gagne
  - 8th Edition is also acceptable
- Online supplements
  - See "Information" link on course website
  - Includes Appendices, sample problems, etc.
- Networking, Databases, Software Eng, Security
  - Limited to lecture notes

Grading

- Rough Grade Breakdown
  - Two Midterms: 40% (20% each)
  - Four Projects: 50% (i.e. 12.5% each)
  - Participation 5%
  - Quizzes: 5%
- Four Projects:
  - Project I: Threads (Nachos)
  - Project II: Multiprogramming (Nachos)
  - Project III: Single node key-value store
  - Project IV: Distributed key-value store
- Late Policy:
  - Four slip days for code (not design documents)!
  - 10% off per day after deadline

Computing Facilities

- Every student who is enrolled should get an account form at the end of lecture
  - Gives you an account of form cs162-xx@cory
  - This account is required
    - » Most of your debugging can be done on other EECS accounts or on your own computers, however…
    - » All of the final runs must be done on your cs162-xx account and must run on an instructional machine
- Make sure to log in into your new account this week

Group Project Simulates Industrial Environment

- Project teams have 4 or 5 members in same discussion section
  - Must work in groups in "the real world"
- Communicate with colleagues (team members)
  - Communication problems are natural
  - What have you done?
  - What answers you need from others?
  - You must document your work!!!
- Communicate with supervisor (TAs)
  - What is the team's plan?
  - What is each member's responsibility?
  - Short progress reports are required

Project Signup (1/2)

- Project Signup: Watch "Group/Section Signup" Link
- 4-5 members to a group
  - Everyone in group must be able to *actually* attend same section
  - The sections assigned to you by Telebears are temporary!
  - Use Piazza pinned teammate search thread (please close when done!)
- Only submit once per group! Due Thursday (1/30) by 11:59pm
  - Everyone in group must have logged into their cs162-xx accounts once before you register the group
  - Make sure that you select at least 3 potential sections

Project Signup (2/2)

- New section assignments emailed by following Monday (2/3)
  - Attend new sections starting 2/6

Sec.	Day	Time	Location	Instructor / TA
101	Th	10:00A-11:00A	6 Evans	Riyaz
102	Th	11:00A-12:00P	9 Evans	Kelvin
103	Th	6:00-7:00P	405 Soda	Vaishaal
104	Th	1:00-2:00P	7 Evans	Riyaz
105	Th	2:00-3:00P	121 Wheeler	Nicholas
106	Th	3:00-4:00P	81 Evans	Nicholas
107	Th	4:00-5:00P	81 Evans	Kelvin
108	Th	5:00-6:00P	81 Evans	Vaishaal
109	Fr	2:00-3:00P	310 Soda	Isaac
110	Fr	3:00-4:00P	310 Soda	Isaac

5 min Break

CS 162 Collaboration Policy

http://inst.eecs.berkeley.edu/~cs162/sp14/collaboration.html

Explaining a concept to someone in another group
Discussing algorithms/testing strategies with other groups
Helping debug someone else's code (in another group)
Searching online for generic algorithms (e.g., hash table)

Sharing code or test cases with another group
Copying OR reading another group's code or test cases
Copying OR reading online code or test cases from from prior years

We compare all project submissions against prior year submissions and online solutions and will take actions (described on the course overview page) against offenders

Challenge: Scale and Dynamic Range

- Enormous scale, heterogeneity, and dynamic range:
  - CPU: sensor motes → GPUs
    - » Cores: one → 100s [2-orders of magnitude variation]
    - » Clusters: few machines → 10,000s machines [4 orders of mag.]
  - Network: Inter-core networks → Internet
    - » Latency: nanosecs → secs (satellite) [9 orders of mag.]
    - » Bandwidth: Kbps → Gbps [6 orders of mag.]
    - » …
  - Storage: caches → disks
    - » Size: MB → TB [6 orders of mag.]
    - » Access time: few nanosecs → millisecs [6 orders of mag.]

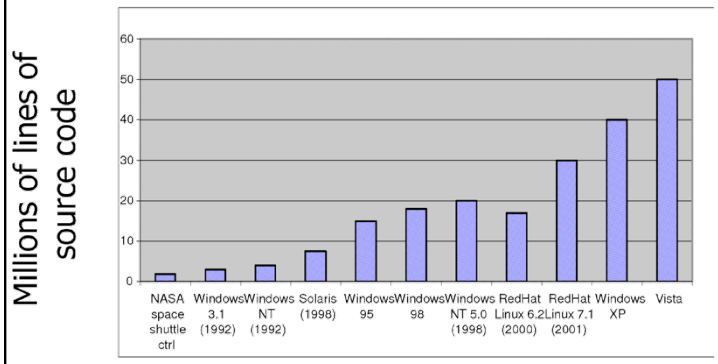
Compare with Cars…

- Horse Power: 50HP → 1000HP [20x times]
- Speed: 100 Km/H → 400 Km/H [4x times]
- Weight: 500 Kg → 20,000 Kg [40x times]
- Mileage: 80 MPG → 2 MPG [40x times]

Challenge: Complexity

- Applications consisting of…
  - … a variety of software modules that …
  - … run on a variety of devices (machines) that
    - » … implement different hardware architectures
    - » … run competing applications
    - » … fail in unexpected ways
    - » … can be under a variety of attacks
- Not feasible to test software for all possible environments and combinations of components and devices
  - The question is not whether there are bugs but how serious are the bugs!

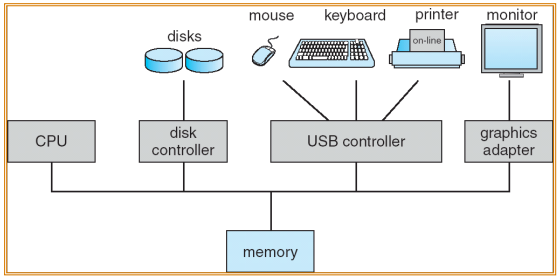
Increasing Software Complexity



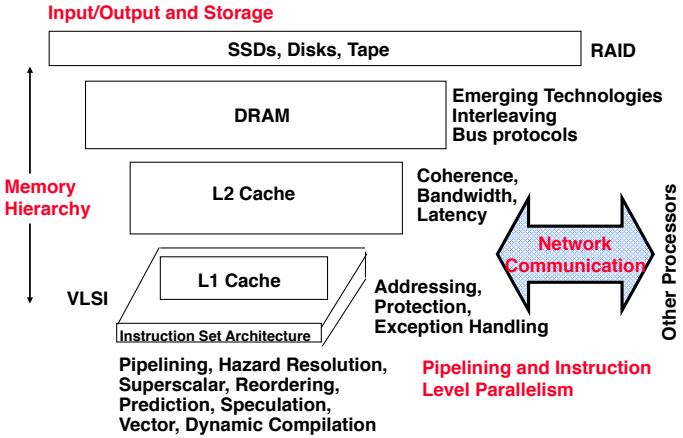
From MIT's 6.033 course

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Sample of Computer Architecture



Example: Some Mars Rover ("Pathfinder") Requirements – Dec 1996

- Pathfinder hardware limitations/complexity:
  - 20Mhz processor, 128MB of DRAM, VxWorks OS
  - Cameras, scientific instruments, batteries, solar panels, and locomotion equipment
  - Many independent processes work together
- Can't hit reset button very easily!
  - Must reboot itself if necessary
  - Must always be able to receive commands from Earth
- Individual Programs must not interfere
  - Suppose the MUT (Martian Universal Translator) module is buggy
  - Better not crash antenna positioning software!
- Further, all software may crash occasionally
  - Automatic restart with diagnostics sent to Earth
  - Periodic checkpoint of results saved?
- Certain functions time critical:
  - Need to stop before hitting something
  - Must track orbit of Earth for communication

How do We Tame Complexity?

- Every piece of computer hardware different
  - Different CPU
    - » Pentium, ARM, PowerPC, ColdFire
  - Different amounts of memory, disk, …
  - Different types of devices
    - » Mice, keyboards, sensors, cameras, fingerprint readers, touch screen
  - Different networking environment
    - » Cable, DSL, Wireless, …
- Questions:
  - Does the programmer need to write a single program that performs many independent activities?
  - Does every program have to be altered for every piece of hardware?
  - Does a faulty program crash everything?

Virtual Machines

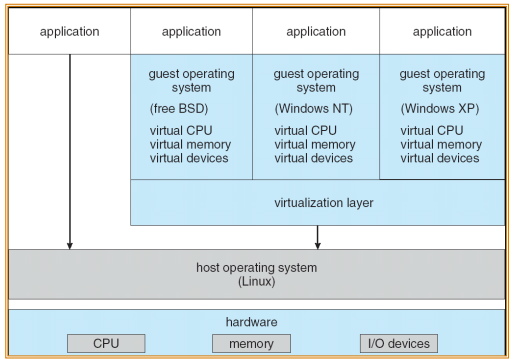
- Software emulation of an abstract machine
  - Give programs illusion they own the machine
  - Make it look like hardware has features you want
- Two types of VMs
  - System VM: supports the execution of an entire OS and its applications (e.g., VMWare Fusion, Parallels Desktop, Xen)
  - Process VM: supports the execution of a single program; this functionality typically provided by OS

Process VMs

- Programming simplicity
  - Each process thinks it has all memory/CPU time
  - Each process thinks it owns all devices
  - Different devices appear to have same interface
  - Device interfaces more powerful than raw hardware
    - » Bitmapped display ⇒ windowing system
    - » Ethernet card ⇒ reliable, ordered, networking (TCP/IP)
- Fault Isolation
  - Processes unable to directly impact other processes
  - Bugs cannot crash whole machine
- Protection and Portability
  - Java interface safe and stable across many platforms

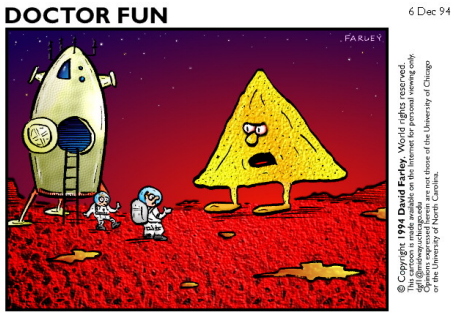
System Virtual Machines: Layers of OSs

- Useful for OS development
  - When OS crashes, restricted to one VM
  - Can aid testing programs on other OSs



Nachos: Virtual OS Environment

- You will be working with Nachos
  - Simulation environment: Hardware, interrupts, I/O
  - Execution of User Programs running on this platform
  - See the "Projects and Nachos" link off the course home page



"This is the planet where nachos rule."

What does an Operating System do?

- Silberschatz and Gavin: "An OS is Similar to a government"
  - Begs the question: does a government do anything useful by itself?
- OS as a Traffic Cop:
  - Manages all resources
  - Settles conflicting requests for resources
  - Prevent errors and improper use of the computer
- OS as a facilitator ("useful" abstractions):
  - Provides facilities/services that everyone needs
  - Standard libraries, windowing systems
  - Make application programming easier, faster, less error-prone
- Some features reflect both tasks:
  - File system is needed by everyone (Facilitator) …
  - … but File system must be protected (Traffic Cop)

What is an Operating System,… Really?

- Most Likely:
  - Memory Management
  - I/O Management
  - CPU Scheduling
  - Synchronization / Mutual exclusion primitives
  - Communications? (Does Email belong in OS?)
  - Multitasking/multiprogramming?
- What about?
  - File System?
  - Multimedia Support?
  - User Interface?
  - Internet Browser? ☺
- Is this only interesting to Academics??

Operating System Definition (Cont'd)

- No universally accepted definition
- "*Everything a vendor ships when you order an operating system*" is good approximation
  - But varies wildly
- "*The one program running at all times on the computer*" is the kernel
  - Everything else is either a system program (ships with the operating system) or an application program

Summary

- Operating systems provide a virtual machine abstraction to handle diverse hardware
- Operating systems coordinate resources and protect users from each other
- Operating systems simplify application development by providing standard services and abstractions
- Operating systems can provide fault containment, fault tolerance, and fault recovery
- CS162 combines things from many other areas of computer science:
  - Languages, data structures, hardware, SW engineering, databases, security, networking, and algorithms