A Short History of Computing Systems and Their Underlying Technology - I

Tiziano Villa (U. Verona)

Adapted by Tiziano Villa from lecture notes by John Kubiatowics (UC Berkeley)
Technology Trends: 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.

2X transistors/Chip Every 1.5 years
Called “Moore’s Law”
Societal Scale Information Systems

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

 MEMS for Sensor Nets

Internet Connectivity

Scalable, Reliable, Secure Services

- Databases
- Information Collection
- Remote Storage
- Online Games
- Commerce

...
People-to-Computer Ratio Over Time

From David Culler

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

⇒ Sea change in chip design: multiple “cores” or processors per chip
ManyCore Chips: The future is here

- Intel 80-core multicore chip (Feb 2007)
  - 80 simple cores
  - Two floating point engines /core
  - Mesh-like "network-on-a-chip"
  - 100 million transistors
  - 65nm feature size

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Voltage</th>
<th>Power</th>
<th>Bandwidth</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.16 GHz</td>
<td>0.95 V</td>
<td>62W</td>
<td>1.62 Terabits/s</td>
<td>1.01 Teraflops</td>
</tr>
<tr>
<td>5.1 GHz</td>
<td>1.2 V</td>
<td>175W</td>
<td>2.61 Terabits/s</td>
<td>1.63 Teraflops</td>
</tr>
<tr>
<td>5.7 GHz</td>
<td>1.35 V</td>
<td>265W</td>
<td>2.92 Terabits/s</td>
<td>1.81 Teraflops</td>
</tr>
</tbody>
</table>

- "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
Another Challenge: Power Density

- **Moore’s Law Extrapolation**
  - Potential power density reaching amazing levels!
- **Flip side: Battery life very important**
  - Moore’s law can yield more functionality at equivalent (or less) total energy consumption
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
Functionality comes with great complexity!

Pentium IV Chip

Proc

Caches

Memory

Busses

adapters

Controllers

I/O Devices:
- Disks
- Displays
- Keyboards

Networks

Proc

Caches

Memory

Busses

adapters

Controllers

I/O Devices:
- Disks
- Displays
- Keyboards

Networks
Sample of Computer Architecture Topics

Input/Output and Storage

- Disks, WORM, Tape
- RAID
- Emerging Technologies
  - Interleaving
  - Bus protocols

Memory Hierarchy

- DRAM
- L2 Cache
- L1 Cache

VLSI

Instruction Set Architecture

- Pipelining, Hazard Resolution
- Superscalar, Reordering
- Prediction, Speculation
- Vector, Dynamic Compilation

Network Communication

- Coherence, Bandwidth, Latency
- Addressing, Protection, Exception Handling
- Pipelining and Instruction Level Parallelism

Other Processors
Increasing Software Complexity

From MIT’s 6.033 course
Example: Some Mars Rover ("Pathfinder") Requirements

- 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) 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, PowerPC, ColdFire, ARM, MIPS
  - Different amounts of memory, disk, …
  - Different types of devices
    » Mice, Keyboards, Sensors, Cameras, Fingerprint readers
  - Different networking environment
    » Cable, DSL, Wireless, Firewalls,…

• 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?
  - Does every program have access to all hardware?
OS Tool: Virtual Machine Abstraction

Application

Operating System

Hardware

• Software Engineering Problem:
  - Turn hardware/software quirks ⇒ what programmers want/need
  - Optimize for convenience, utilization, security, reliability, etc...

• For Any OS area (e.g. file systems, virtual memory, networking, scheduling):
  - What’s the hardware interface? (physical reality)
  - What’s the application interface? (nicer abstraction)
• Why do interfaces look the way that they do?
  - History, Functionality, Stupidity, Bugs, Management
• Should responsibilities be pushed across boundaries?
  - RISC architectures, Graphical Pipeline Architectures
Virtual Machines

• Software emulation of an abstract machine
  - Make it look like hardware has features you want
  - Programs from one hardware & OS on another one

• 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
**Virtual Machines (con't): Layers of OSs**

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

![Diagram of OS layers]

- Application
- Guest operating system
  - (free BSD)
  - virtual CPU
  - virtual memory
  - virtual devices
- Virtualization layer
- Host operating system
  - (Linux)
- Hardware
  - CPU
  - Memory
  - I/O devices
What does an Operating System do?

- Silerschatz and Gavin:
  
  “An OS is Similar to a government”
  - Begs the question: does a government do anything useful by itself?

- Coordinator and Traffic Cop:
  - Manages all resources
  - Settles conflicting requests for resources
  - Prevent errors and improper use of the computer

- Facilitator:
  - Provides facilities that everyone needs
  - Standard Libraries, Windowing systems
  - Make application programming easier, faster, less error-prone

- Some features reflect both tasks:
  - E.g. 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
  - 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.)

- 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
What if we didn’t have an Operating System?

- Source Code $\Rightarrow$ Compiler $\Rightarrow$ Object Code $\Rightarrow$ Hardware
- How do you get object code onto the hardware?
- How do you print out the answer?
- Once upon a time, had to Toggle in program in binary and read out answer from LED’s!

Altair 8080
Simple OS: What if only one application?

- **Examples:**
  - Very early computers
  - Early PCs
  - Embedded controllers (elevators, cars, etc)

- **OS becomes just a library of standard services**
  - Standard device drivers
  - Interrupt handlers
  - Math libraries
MS-DOS Layer Structure

application program

resident system program

MS-DOS device drivers

ROM BIOS device drivers
More thoughts on Simple OS

• What about Cell-phones, XBoxes, etc?
  - Is this organization enough?
• Can OS be encoded in ROM/Flash ROM?
• Does OS have to be software?
  - Can it be Hardware?
  - Custom Chip with predefined behavior
  - Are these even OSs?
More complex OS: Multiple Apps

• **Full Coordination and Protection**
  - Manage interactions between different users
  - Multiple programs running simultaneously
  - Multiplex and protect Hardware Resources
    » CPU, Memory, I/O devices like disks, printers, etc

• **Facilitator**
  - Still provides Standard libraries, facilities

• Would this complexity make sense if there were only one application that you cared about?
## UNIX System Structure

<table>
<thead>
<tr>
<th>User Mode</th>
<th>Kernel Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td>Standard Libs</td>
</tr>
<tr>
<td>(the users)</td>
<td>shells and commands</td>
</tr>
<tr>
<td></td>
<td>compilers and interpreters</td>
</tr>
<tr>
<td></td>
<td>system libraries</td>
</tr>
<tr>
<td></td>
<td><strong>system-call interface to the kernel</strong></td>
</tr>
<tr>
<td></td>
<td>signals terminal handling</td>
</tr>
<tr>
<td></td>
<td>file system</td>
</tr>
<tr>
<td></td>
<td>character I/O system</td>
</tr>
<tr>
<td></td>
<td>swapping block I/O system</td>
</tr>
<tr>
<td></td>
<td>terminal drivers</td>
</tr>
<tr>
<td></td>
<td>disk and tape drivers</td>
</tr>
<tr>
<td></td>
<td><strong>kernel interface to the hardware</strong></td>
</tr>
<tr>
<td></td>
<td>terminal controllers terminals</td>
</tr>
<tr>
<td></td>
<td>device controllers disks and tapes</td>
</tr>
<tr>
<td></td>
<td>memory controllers physical memory</td>
</tr>
<tr>
<td>Hardware</td>
<td>Kernel</td>
</tr>
</tbody>
</table>
New Structures for Multicore chips?
Tessellation: The Exploded OS

- Normal Components split into pieces
  - Device drivers (Security/Reliability)
  - Network Services (Performance)
    » TCP/IP stack
    » Firewall
    » Virus Checking
    » Intrusion Detection
  - Persistent Storage (Performance, Security, Reliability)
  - Monitoring services
    » Performance counters
    » Introspection
  - Identity/Environment services (Security)
    » Biometric, GPS, Possession Tracking
- Applications Given Larger Partitions
  - Freedom to use resources arbitrarily

Large Compute-Bound Application
Real-Time Application
Persistent Storage & File System

Firewall
Virus Intrusion
Monitor And Adapt
Video & Window Drivers
Device Drivers

HCI/Voice Rec
Identity
OS Systems Principles

• OS as illusionist:
  - Make hardware limitations go away
  - Provide illusion of dedicated machine with infinite memory and infinite processors

• OS as government:
  - Protect users from each other
  - Allocate resources efficiently and fairly

• OS as complex system:
  - Constant tension between simplicity and functionality or performance

• OS as history teacher
  - Learn from past
  - Adapt as hardware tradeoffs change
Why Study Operating Systems?

• Learn how to build complex systems:
  - How can you manage complexity for future projects?

• Engineering issues:
  - Why is the web so slow sometimes? Can you fix it?
  - What features should be in the next Mars Rover?
  - How do large distributed systems work? (Kazaa, etc)

• Buying and using a personal computer:
  - Why different PCs with same CPU behave differently
  - How to choose a processor (Opteron, Itanium, Celeron, Pentium, Hexium)? [Ok, made last one up]
  - Should you get Windows XP, 2000, Linux, Mac OS ...?
  - Why does Microsoft have such a bad name?

• Business issues:
  - Should your division buy thin-clients vs PC?

• Security, viruses, and worms
  - What exposure do you have to worry about?
“In conclusion...”

• 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
• Operating systems can provide an array of fault containment, fault tolerance, and fault recovery

• The study of operating systems combines things from many other areas of computer science -
  - Languages, data structures, hardware, and algorithms
A Short History of Computing Systems and their Underlying Technologies - II

Tiziano Villa (U. Verona)

Adapted by Tiziano Villa from lecture notes by John Kubiatowicz (UC Berkeley)
Moore's Law Change Drives OS Change

<table>
<thead>
<tr>
<th></th>
<th>1981</th>
<th>2009</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU MHz, Cycles/inst</strong></td>
<td>10, 3–10</td>
<td>Quad 3.2G, 0.25–0.5</td>
<td>1,280, 6–40</td>
</tr>
<tr>
<td><strong>DRAM capacity</strong></td>
<td>128KB</td>
<td>6GB</td>
<td>49,152</td>
</tr>
<tr>
<td><strong>Disk capacity</strong></td>
<td>10MB</td>
<td>1.5TB</td>
<td>150,000</td>
</tr>
<tr>
<td><strong>Net bandwidth</strong></td>
<td>9600 b/s</td>
<td>1 Gb/s</td>
<td>110,000</td>
</tr>
<tr>
<td><strong># addr bits</strong></td>
<td>16</td>
<td>64</td>
<td>4</td>
</tr>
<tr>
<td><strong>#users/machine</strong></td>
<td>10s</td>
<td>≤ 1</td>
<td>≤ 0.1</td>
</tr>
<tr>
<td><strong>Price</strong></td>
<td>$25,000</td>
<td>$3,500</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Typical academic computer 1981 vs 2009
Moore’s law effects

- Nothing like this in any other area of business
- Transportation in over 200 years:
  - 2 orders of magnitude from horseback @10mph to Concorde @1000mph
  - Computers do this every decade (at least until 2002)!
- What does this mean for us?
  - Techniques have to vary over time to adapt to changing tradeoffs
- I place a lot more emphasis on principles
  - The key concepts underlying computer systems
  - Less emphasis on facts that are likely to change over the next few years...
- Let’s examine the way changes in $/MIP has radically changed how OS’s work
A pioneer  
**J. Von Neumann: (1903–1957)**

- J. Von Neumann described in 1945 in the paper “First Draft of a Report on the EDVAC” a computer architecture consisting of a processing unit with an arithmetic logic unit and processor registers, of a control unit with an instruction register and program counter, of a memory to store both data and instructions, of external mass storage, and of input and output devices.

- A Von Neumann architecture in practice denotes any stored program computer where an instruction fetch and a data operation cannot occur at the same time because they share a common bus (Von Neumann bottleneck).

- Von Neumann contribution is built on a flow of previous ideas due to A. Turing and others.
"The machine designed by Drs. Eckert and Mauchly was a monstrosity. When it was finished, the ENIAC filled an entire room, weighed thirty tons, and consumed two hundred kilowatts of power."

http://ei.cs.vt.edu/~history/ENIAC.Richey.HTML
History Phase 1 (1948–1970)
Hardware Expensive, Humans Cheap

- When computers cost millions of $’s, optimize for more efficient use of the hardware!
  - Lack of interaction between user and computer

- **User at console**: one user at a time
- **Batch monitor**: load program, run, print

- Optimize to better use hardware
  - When user thinking at console, computer idle $\Rightarrow$ BAD!
  - Feed computer batches and make users wait
  - Autograder for this course is similar

- *No protection*: what if batch program has bug?
Core Memories (1950s & 60s)

The first magnetic core memory, from the IBM 405 Alphabetical Accounting Machine.

- Core Memory stored data as magnetization in iron rings
  - Iron “cores” woven into a 2-dimensional mesh of wires
  - Origin of the term “Dump Core”
  - Rumor that IBM consulted Life Saver company
- See: http://www.columbia.edu/acis/history/core.html
History Phase $1 \frac{1}{2}$ (late 60s/early 70s)

- **Data channels, Interrupts**: overlap I/O and compute
  - DMA - Direct Memory Access for I/O devices
  - I/O can be completed asynchronously

- **Multiprogramming**: several programs run simultaneously
  - Small jobs not delayed by large jobs
  - More overlap between I/O and CPU
  - Need memory protection between programs and/or OS

- **Complexity gets out of hand**:
  - Multics: announced in 1963, ran in 1969
    » 1777 people “contributed to Multics” (30-40 core dev)
    » Turing award lecture from Fernando Corbató (key researcher): “On building systems that will fail”
  - OS 360: released with 1000 known bugs (APARs)
    » “Anomalous Program Activity Report”

- **OS finally becomes an important science**:
  - How to deal with complexity???
  - UNIX based on Multics, but vastly simplified
A Multics System (Circa 1976)

• The 6180 at MIT IPC, skin doors open, circa 1976:
  - “We usually ran the machine with doors open so the operators could see the AQ register display, which gave you an idea of the machine load, and for convenient access to the EXECUTE button, which the operator would push to enter BOS if the machine crashed.”

• http://www.multicians.org/multics-stories.html
1973: 1.7 Mbit/sq. in 140 MBytes
1979: 7.7 Mbit/sq. in 2,300 MBytes

Contrast: Seagate 2TB, 400 GB/SQ in, 3½ in disk, 4 platters
History Phase 2 (1970 - 1985)
Hardware Cheaper, Humans Expensive

- Computers available for tens of thousands of dollars instead of millions
- OS Technology maturing/stabilizing
- **Interactive timesharing:**
  - Use cheap terminals (~$1000) to let multiple users interact with the system at the same time
  - Sacrifice CPU time to get better response time
  - Users do debugging, editing, and email online
- **Problem: Thrashing**
  - Performance very non-linear response with load
  - Thrashing caused by many factors including
    » Swapping, queueing
The ARPANet (1968-1970's)

- Paul Baran
  - RAND Corp, early 1960s
  - Communications networks that would survive a major enemy attack
- ARPANet: Research vehicle for "Resource Sharing Computer Networks"
  - 2 September 1969: UCLA first node on the ARPANet
  - December 1969: 4 nodes connected by 56 kbps phone lines
  - 1971: First Email
  - 1970's: <100 computers

BBN team that implemented the interface message processor
ARPANET GEOGRAPHIC MAP, OCTOBER 1980

SATELLITE CIRCUIT
○ IMP
□ TIP
△ PLURIBUS IMP
◊ PLURIBUS TIP
● C30

(NOTE: THIS MAP DOES NOT SHOW ARPA'S EXPERIMENTAL SATELLITE CONNECTIONS)
NAMES SHOWN ARE IMP NAMES, NOT (NECESSARILY) HOST NAMES
**ARPANet Evolves into Internet**

- **First E-mail SPAM message**: 1 May 1978 12:33 EDT

- **80-83**: TCP/IP, DNS; ARPANET and MILNET split

- **85-86**: NSF builds NSFNET as backbone, links 6 Supercomputer centers, 1.5 Mbps, 10,000 computers

- **87-90**: link regional networks, NSI (NASA), ESNet (DOE), DARTnet, TWBNet (DARPA), 100,000 computers

<table>
<thead>
<tr>
<th>ARPANet</th>
<th>SATNet</th>
<th>TCP/IP</th>
<th>NSFNet</th>
<th>Deregulation &amp; Commercialization</th>
<th>ISP</th>
<th>ASP</th>
<th>AIP</th>
</tr>
</thead>
</table>

SATNet: Satelite network  
PRNet: Radio Network
What is a Communication Network?
(End-system Centric View)

• Network offers one basic service: move information
  - Bird, fire, messenger, truck, telegraph, telephone, Internet ...
  - Another example, transportation service: move objects
    » Horse, train, truck, airplane ...

• What distinguish different types of networks?
  - The services they provide

• What distinguish the services?
  - Latency
  - Bandwidth
  - Loss rate
  - Number of end systems
  - Service interface (how to invoke the service?)
  - Others
    » Reliability, unicast vs. multicast, real-time...
What is a Communication Network?  
(Infrastructure Centric View)

- Communication medium: electron, photon
- Network components:
  - Links - carry bits from one place to another (or maybe multiple places): fiber, copper, satellite, …
  - Interfaces - attach devices to links
  - Switches/routers - interconnect links: electronic/optic, crossbar/Banyan
  - Hosts - communication endpoints: workstations, PDAs, cell phones, toasters
- Protocols - rules governing communication between nodes
  - TCP/IP, ATM, MPLS, SONET, Ethernet, X.25
- Applications: Web browser, X Windows, FTP, …
Network Components (Examples)

**Links**
- Fibers
- Coaxial Cable

**Interfaces**
- Ethernet card
- Wireless card

**Switches/routers**
- Large router
- Telephone switch
Types of Networks

• Geographical distance
  - Local Area Networks (LAN): Ethernet, Token ring, FDDI
  - Metropolitan Area Networks (MAN): DQDB, SMDS
  - Wide Area Networks (WAN): X.25, ATM, frame relay
  - Caveat: LAN, MAN, WAN may mean different things
    » Service, network technology, networks

• Information type
  - Data networks vs. telecommunication networks

• Application type
  - Special purpose networks: airline reservation network, banking network, credit card network, telephony
  - General purpose network: Internet
History Phase 3 (1981—)
Hardware Very Cheap, Humans Very Expensive

- Computer costs $1K, Programmer costs $100K/year
  - If you can make someone 1% more efficient by giving them a computer, it’s worth it!
  - Use computers to make people more efficient
- Personal computing:
  - Computers cheap, so give everyone a PC
- Limited Hardware Resources Initially:
  - OS becomes a subroutine library
  - One application at a time (MSDOS, CP/M, …)
- Eventually PCs become powerful:
  - OS regains all the complexity of a “big” OS
  - multiprogramming, memory protection, etc (NT, OS/2)
- Question: As hardware gets cheaper does need for OS go away?
History Phase 3 (con’t)
Graphical User Interfaces

- CS160 ⇒ All about GUIs
- Xerox Star: 1981
  - Originally a research project (Alto)
  - First “mice”, “windows”
- Apple Lisa/Machintosh: 1984
  - “Look and Feel” suit 1988
- Microsoft Windows:
  - Win 1.0 (1985)
  - Win 3.1 (1990)
  - Win 95 (1995)
  - Win NT (1993)
  - Win XP (2001)
  - Win Vista (2007)
History Phase 4 (1988–): Distributed Systems

- **Networking (Local Area Networking)**
  - Different machines share resources
  - Printers, File Servers, Web Servers
  - Client – Server Model

- **Services**
  - Computing
  - File Storage
History Phase 4 (1988—): Internet

- Developed by the research community
  - Based on open standard: Internet Protocol
  - Internet Engineering Task Force (IETF)
- Technical basis for many other types of networks
  - Intranet: enterprise IP network
- Services Provided by the Internet
  - Shared access to computing resources: telnet (1970's)
  - Shared access to data/files: FTP, NFS, AFS (1980's)
  - Communication medium over which people interact
    » email (1980's), on-line chat rooms, instant messaging (1990's)
    » audio, video (1990's, early 00's)
  - Medium for information dissemination
    » USENET (1980's)
    » WWW (1990's)
    » Audio, video (late 90's, early 00's) - replacing radio, TV?
    » File sharing (late 90's, early 00's)
Network “Cloud”
Regional Nets + Backbone

LAN: Local Area Network
Backbones + NAPs + ISPs

ISP: Internet Service Provider
NAP: Network Access Point
Computers Inside the Core

- DSL Always on
- Cable Head Ends
- @home
- Covad
- IPS
- NAP
- Satellite Fixed Wireless
- Cingular
- Sprint
- Cell
- Cell
- LAN
- Dial-up
- AOL
The Morris Internet Worm (1988)

- **Internet worm (Self-reproducing)**
  - Author Robert Morris, a first-year Cornell grad student
  - Launched close of Workday on November 2, 1988
  - Within a few hours of release, it consumed resources to the point of bringing down infected machines

- **Techniques**
  - Exploited UNIX networking features (remote access)
  - Bugs in *finger* (buffer overflow) and *sendmail* programs (debug mode allowed remote login)
  - Dictionary lookup-based password cracking
  - Grappling hook program uploaded main worm program
LoveLetter Virus (May 2000)

- E-mail message with VBScript (simplified Visual Basic)
- Relies on Windows Scripting Host
  - Enabled by default in Win98/2000
- User clicks on attachment ➔ infected!
  - E-mails itself to everyone in Outlook address book
  - **Replaces** some files with a copy of itself
  - Searches all drives
  - Downloads password cracking program
- **60-80% of US companies infected and 100K European servers**
History Phase 5 (1995—): Mobile Systems

• Ubiquitous Mobile Devices
  - Laptops, PDAs, phones
  - Small, portable, and inexpensive
    » Recently twice as many smart phones as PDAs
    » Many computers/person!
  - Limited capabilities (memory, CPU, power, etc...)

• Wireless/Wide Area Networking
  - Leveraging the infrastructure
  - Huge distributed pool of resources extend devices
  - Traditional computers split into pieces. Wireless keyboards/mice, CPU distributed, storage remote

• Peer-to-peer systems
  - Many devices with equal responsibilities work together
  - Components of “Operating System” spread across globe
CITRIS's Model: A Societal Scale Information System

- Center for Information Technology Research in the Interest of Society
- The Network is the OS
  - Functionality spread throughout network

MEMS for Sensor Nets

Scalable, Reliable, Secure Services

Mobile, Ubiquitous Systems
Datacenter is the Computer

- (From Luiz Barroso’s talk at RAD Lab 12/11)
- Google *program* == Web search, Gmail,…
- Google *computer* ==

- Thousands of computers, networking, storage
- Warehouse-sized facilities and workloads may be unusual today but are likely to be more common in the next few years
Migration of Operating-System Concepts and Features

- 1950: Mainframes, no software
- 1960: Minicomputers, no software
- 1970: Multics, multiuser
- 1980: Distributed systems, multiprocessor
- 1990: Fault tolerant
- 2000: Interactive, multiprocessor

Time lines:
- Batch
- Time shared
- Multiuser
- Networked
- Clustered