Advanced Operating Systems (CS 202)

OS Evolution and Organization

Jan, 13, 2016
Expectations and little bit about me

• I am NOT an OS person
  - I drew the short straw 😊
    • My second time teaching this class
  - My favorite two answers are
    • I don’t know
    • What do YOU think?
  - I am looking forward to learn with you

• ...but I do know a lot about OS
  - I am a systems person
  - I work in architecture, networking, high performance computing and security
    • OS is at the intersection of all systems areas
Coming up

• Reading for next week (Extensibility):
  - For Wednesday, Spin (critique) and Exokernel
  - Friday, L4
  - Links on class website

• Programming assignment released Monday
  - Optional Lab0 released today
Today

• Evolution of Operating Systems (and computers!)
  - Some slides modified from Silberschatz and Gavin, as well as Margo Seltzer

• Operating Systems models

• Why study history?
  - Understand why OS’s look like they are
  - Appreciate how and why different pieces evolved
  - Explain how external forces also shape OS
  - Provide context for the rest of the quarter
  - It’s interesting!
Dawn of computing

- Pre 1950: the very first electronic computers
  - valves and relays
  - single program with dedicated function

- Pre 1960: stored program valve machines
  - single job at a time; OS is a program loader
Phase 0 of OS Evolution (40s to 1955)

• No OS
  – Computers are exotic, expensive, large, slow experimental equipment
  – Program in machine language and using plugboards
  – User sits at console: no overlap between computation, I/O, user thinking, etc..
    • Program manually by plugging wires in
    • Goal: number crunching for missile computations

• Imagine programming that way
  – Painful and slow
OS progress in this period

• Libraries of routines that are common
  – Including those to talk to I/O devices
  – Punch cards (enabling copying/exchange of these libraries) a big advance!
  – Pre-cursor to OS
Phase 1: 1955-1970

• Computers expensive; people cheap
  - Use computers efficiently – move people away from machine
  - OS becomes a batch monitor
    • Loads a job, runs it, then moves on to next
    • If a program fails, OS records memory contents somewhere
    • More efficient use of hardware but increasingly difficult to debug
• **Batch systems on mainframe computers**
  - collections of jobs made up into a *batch*
  - example: IBM 1401/7094
    • card decks spooled onto magnetic tape and from tape to printer

  - example: English Electric Leo KDF9
    • 32K 48-bit words, 2µsec cycle time
    • punched paper-tape input 'walk-up' service or spooling via mag tape
Advances in technology in this stage

- Data channels and interrupts
  - Allow overlap of I/O and computing
  - Buffering and interrupt handling done by OS
  - Spool (buffer) jobs onto “high speed” drums
Phase 1, problems

• Utilization is low (one job at a time)
• No protection between jobs
• Short jobs wait behind long jobs
  - So, we can only run one job at a time
• Coordinating concurrent activities
• Still painful and slow (but less so?)
Advances in OS in this period

- Hardware provided memory support (protection and relocation)
- Multiprogramming (not to be confused with time sharing)
- Scheduling: let short jobs run first
- OS must manage interactions between concurrent things
  - Starts emerging as a field/science
- OS/360 from IBM first OS designed to run on a family of machines from small to large
Some important projects

• Atlas computer/OS from Manchester U. (late 50s/early 60s)
  - First recognizable OS
  - Separate address space for kernel
  - Early virtual memory

• THE Multiprogramming system (early 60s)
  - Introduced semaphores
  - Attempt at proving systems correct; interesting software engineering insights
Not all is smooth

- Operating systems didn’t really work
- No software development or structuring tools; written in assembly
- OS/360 introduced in 1963 but did not really work until 1968
  - Reported on in mythical man month
- Extremely complicated systems
  - 5–7 years development time typical
  - Written in assembly, with no structured programming
  - Birth of software engineering?
Phase 2: 1970s

- Computers and people are expensive
  - Help people be more productive
  - Interactive time sharing: let many people use the same machine at the same time
  - Emergence of minicomputers
    - Terminals are cheap
  - Keep data online on fancy file systems
  - Attempt to provide reasonable response times (Avoid thrashing)
Important advances and systems

• Compatible Time-Sharing System (CTSS)
  - MIT project (demonstrated in 1961)
  - One of the first time sharing systems
  - Corbato won Turing award in 1990
  - Pioneered much of the work in scheduling
  - Motivated MULTICS
MULTICS

• Jointly developed by MIT, Bell Labs and GE
• Envisioned one main computer to support everyone
  – People use computing like a utility like electricity – sound familiar? Ideas get recycled
• Many many fundamental ideas: protection rings, hierarchical file systems, devices as files, ...
• Building it was more difficult than expected
• Technology caught up
Sabre system

• System to run airline systems
  – Still in use!
• Minicomputer (then) with terminals for reservation agents
• Important ideas such as transaction processing
Unix appears

- Ken Thompson, who worked on MULTICS, wanted to use an old PDP-7 laying around in Bell labs.
- He and Dennis Richie built a system designed by programmers for programmers.
- Originally in assembly. Rewritten in C.
  - If you notice for the paper, they are defending this decision.
  - However, this is a new and important advance: portable operating systems!
- Shared code with everyone (particularly universities)
Unix (cont’d)

- Berkeley added support for virtual memory for the VAX
- DARPA selected Unix as its networking platform in arpanet
- Unix became commercial
  - ...which eventually lead Linus Torvald to develop Linux
Some important ideas in Unix

• OS written in a high level language
• OS portable across hardware platforms
  – Computing is no longer a pipe stove/vertical system
• Pipes
• Mountable file systems
• Many more (we’ll talk about unix later)
Phase 3: 1980s

- Computers are cheap, people expensive
  - Put a computer in each terminal
  - CP/M from DEC first personal computer OS (for 8080/85) processors
  - IBM needed software for their PCs, but CP/M was behind schedule
  - Approached Bill Gates to see if he can build one
  - Gates approached Seattle computer products, bought 86-DOS and created MS-DOS
  - Goal: finish quickly and run existing CP/M software
  - OS becomes subroutine library and command executive
New technologies in Phase 3

• Personal workstations
  – The PERQ
  – Xerox Alto
  – SUN workstation

• Personal computers
  – Apple II
  – IBM PC
  – Macintosh
New technologies (cont’d)

• Business applications!
  - Word processors
  - Spreadsheets
  - Databases

• Marketplace is broken up horizontally
  - Hardware
  - OS
  - Applications
New advances in OS

• PC OS was a regression for OS
  – Stepped back to primitive phase 1 style
    OS leaving the cool developments that
    occurred in phase 2

• Academia was still active, and some
developments still occurred in
mainframe and workstation space
Phase 4: Networked systems
1990s to 2010s

• Machines can talk to each other
  – it's all about connectivity
• We want to share data not hardware
• Networked applications drive everything
  – Web, email, messaging, social networks, ...
• Protection and multiprogramming less important for personal machines
  – But more important for servers
Phase 4, continued

• Market place continued horizontal stratification
  - ISPs (service between OS and applications)
  - Information is a commodity
  - Advertising a new marketplace

• New network based architectures
  - Client server
  - Clusters
  - Grids
  - Distributed operating systems
  - Cloud computing (or is that phase 5?)
New problems

• Large scale
  - Google file system, mapreduce, ...

• Concurrency at large scale
  - ACID (Atomicity, Consistency, Isolation and Durability) in Internet Scale systems
    • Very large delays
    • Partitioning

• Security and Privacy
Phase 5
2010s -- ??

• New generation?
• Mobile devices that are powerful
• Sensing: location, motion, ...
• Cyberphysical systems
• Computing evolving beyond networked systems
  - But OS for them looks largely the same
  - Is that a good idea?