Main Points

• Operating system definition
  – Software to manage a computer’s resources for its users and applications

• OS challenges
  – Reliability, security, responsiveness, portability, ...

• OS history
  – How are OS X, Windows 7, and Linux related?
What is an operating system?

- Software to manage a computer’s resources for its users and applications.
Operating System Roles

• Referee:
  – Resource (limited) allocation among users, applications
  – Parallel activities: need to keep them separate
  – Isolation of different users applications from each other (bugs)
  – Communication between users, applications - cooperation

• Illusionist
  – limited hw: it appears much large and powerful (memory)
  – Each application appears to have the entire machine to itself
  – Infinite number of processors, (near) infinite amount of memory, reliable storage, reliable network transport

• Glue
  – Provide common, standard services to applications, independence from specific peripherals
  – Simplifies application development : common simple interfaces
  – Libraries, user interface widgets, ...
Operating system design patterns

• Cloud computing
  – Referee: how to allocate resources between competing applications in the cloud?
  – Illusionist: computing resources in a cloud evolve continuously, how to isolate applications from this evolution?
  – Glue: how to provide common, standardized access to the cloud services?

• Web services
  – Referee: ensure responsiveness when multiple tabs are opened at the same time
  – Illusionist: web services are geographically distributed for fault tolerance and performance. Mask server failures to the users.
  – Glue: how does a browser achieve portable execution of scripts across different OS and HW platforms?
Operating system design patterns

• Multi-user database systems
  – Referee: how to enforce data access and privacy to different users?
  – Illusionist: how to mask failures so that data remains consistent and available to users?
  – Glue: what common services to programs development?

• Internet
  – Referee: guarantee differentiated services to users and protect against DoS, spam, phishing etc...
  – Illusionist: internet appears as a unique, world-wide network but it is not!
  – Glue: internet protocols make applications independent of the underlying network architecture
Example: web service

- It defines an astonishingly simple behavior:
  - Receives a packet with a web page request
  - Retrieves the web page from disk
  - Sends back the page
Example: web service

However:

- Many requests involve data and computations
  - Think about search engines, a request may involve deep computations over large clusters of machines
- Multiple users issue requests at the same time
  - These must be managed simultaneously
- The server uses caches to speed up
  - Cache is shared among users, need for synchronized access mechanisms
- Servers send to clients scripts for pages customization
  - How does the client can protect itself from the execution of third party code that may embed viruses/spyware?
Example: web service

However:

• Web sites need to be updated
  – How to manage consistency with concurrent read requests?

• Client and server may run at different speeds
  – Need for speed decoupling

• Hardware supporting the web site may be updated
  – How to take advantage of this without rewriting the web server code?
OS Challenges

• Reliability
  – Does the system do what it was designed to do? Malware and bugs
  – Availability
    – What portion of the time is the system working?
    – Mean Time To Failure (MTTF), Mean Time to Repair

• Security
  – Can the system be compromised by an attacker?
  – Privacy
    – Data is accessible only to authorized users

Both require very careful design and code
OS Challenges

• **Portability** *(independence from Hw and Sw)*
  – For programs:
    – Application programming interface (API)
    – Abstract machine interface
  – For the operating system
  – Hardware abstraction layer
  – Provides hardware-specific OS kernel routines
OS Challenges

• Performance
  – Latency/response time
  – How long does an operation take to complete?
  – Throughput
  – How many operations can be done per unit of time?
  – Overhead
  – How much extra work is done by the OS?
  – Fairness
  – How equal is the performance received by different users?
  – Predictability
  – How consistent is the performance over time?
OS Adoption

• Adoption is beyond control of an OS
  – Wide availability of applications
  – Wide availability of HW supporting it

• Network effect
  – App stores and iOS (proprietary)
  – Example: Android model vs iPhone model
    (reliability vs. adoption)
  – Applications \(\Rightarrow\) Hw \(\Rightarrow\) O.S. \(\Rightarrow\) Applications \(\Rightarrow\)... 

• Proprietary vs open systems
  – Not a clear winner
OS History

- MVS (60's)
  - MS/DOS (70's)
    - Windows (80's)
      - Windows Mobile NT
        - Windows 8 (2012)
    - VMS (70's)
  - Multics (60's)
    - UNIX (70's)
      - BSD UNIX (80's)
      - UNIX (90's - present)
        - Free BSD
        - Linux
          - Android
      - Mach (80's)
    - VMware
    - MacOS
      - MacOS X
      - iOS

- Influence
- Descendant
## Computer Performance Over Time

<table>
<thead>
<tr>
<th></th>
<th>1981</th>
<th>1996</th>
<th>2011</th>
<th>factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIPS</td>
<td>1</td>
<td>300</td>
<td>10000</td>
<td>10K</td>
</tr>
<tr>
<td>MIPS/$</td>
<td>$100K</td>
<td>$30</td>
<td>$0.50</td>
<td>200K</td>
</tr>
<tr>
<td>DRAM</td>
<td>128KB</td>
<td>128MB</td>
<td>10GB</td>
<td>100K</td>
</tr>
<tr>
<td>Disk</td>
<td>10MB</td>
<td>4GB</td>
<td>1TB</td>
<td>100K</td>
</tr>
<tr>
<td>Home Internet</td>
<td>9.6 Kbps</td>
<td>256 Kbps</td>
<td>5 Mbps</td>
<td>500</td>
</tr>
<tr>
<td>LAN network</td>
<td>3 Mbps (shared)</td>
<td>10 Mbps</td>
<td>1 Gbps</td>
<td>300</td>
</tr>
<tr>
<td>Users per machine</td>
<td>100</td>
<td>1</td>
<td>&lt;&lt; 1</td>
<td>100+</td>
</tr>
</tbody>
</table>
Early Operating Systems: Computers Very Expensive

• One application at a time
  – Had complete control of hardware
  – OS was runtime library
  – Users would stand in line to use the computer
• Batch systems
  – Keep CPU busy by having a queue of jobs
  – OS would load next job while current one runs
  – Users would submit jobs, and wait, and wait, and wait, and
Single task systems

• Sequential execution
Early batch systems

- SPOOL: Simultaneous Peripheral Operation On-Line
Multi-programmed batch systems

- multi-user system: several programs loaded in memory at the same time
- Spool optimization
- Resource optimization (processor, memory, devices)
  - Response time not important

<table>
<thead>
<tr>
<th>Operating system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program 1</td>
</tr>
<tr>
<td>Program 2</td>
</tr>
<tr>
<td>Program 3</td>
</tr>
</tbody>
</table>
Multi-tasking vs single-task
Time-Sharing Operating Systems: Computers and People Expensive

- Multiple users on computer at same time
  - Multiprogramming: run multiple programs at same time
  - Interactive performance: try to complete everyone’s tasks quickly
  - As computers became cheaper, more important to optimize for user time, not computer time
Time-Sharing Operating Systems

• time sharing v.s. multitasking

Multi tasking systems

Time sharing systems

(QdT)
Today’s Operating Systems: Computers Cheap

• Smartphones
• Embedded systems
• Web servers
• Laptops
• Tablets
• Virtual machines
• ...

Tomorrow’s Operating Systems

- Giant-scale data centers
- Increasing numbers of processors per computer
- Increasing numbers of computers per user
- Very large scale storage
Bonus Thought Question

• How should an operating system allocate processing time between competing uses?
  – Give the CPU to the first to arrive?
  – To the one that needs the least resources to complete? To the one that needs the most resources?
  – What if you need to allocate memory?
  – Disk?
Textbook

• Lazowska, Spring 2012: “The text is quite sophisticated. You won't get it all on the first pass. The right approach is to [read each chapter before class and] re-read each chapter once we've covered the corresponding material... more of it will make sense then. Don't save this re-reading until right before the mid-term or final – keep up.”