Chapter 1

CS 3204: Introduction
What is an Operating System (OS)?

- Definition 1:
  - An OS is the *interface* between the hardware and the software environment, equivalent to an *extended* or *virtual* machine.

- Definition 2:
  - An OS is a *resource manager* – provides “resource abstraction”

- In fact, it achieves 1 through 2.
- Therefore, both definitions are applicable at some times.
System Software and the OS interface

Diagram showing the layers of system software and the operating system:

1. Hardware
2. Software-Hardware Interface
3. Operating System
4. Operating System Interface
5. Other System Software
6. Application Programming Interface
7. Application Software
8. Your software

Legend:
- Drivers
- Resource Sharing
- Resource Abstraction

Editors, Compilers, Loaders

From the textbook

CS 3204: Operating Systems
Resource Abstraction

- How does the OS “manage resources”? 
  - By providing *Resource Abstraction* to the other system software and applications
- What is Abstraction? 
  - Abstraction hides the details

- *Resource Abstraction* 
  - hides the “nitty-gritty” details of the underlying resource
Resource Abstraction ... an example

(Consider the C language statement `fprintf`)

\[
\begin{align*}
\text{fprintf ( fileId , "\%d" , var1 )} \\
\text{write ( block , 100 , device , 266 , 9 )} \\
\text{load ( block , 100 , device )} \\
\text{seek ( device , 266 )} \\
\text{out ( device , 9 )}
\end{align*}
\]
Resource Abstraction

- Typical resource abstractions
  - Memory
  - Disk
  - Keyboard
  - Monitor
Resource Sharing

- Managing resources through abstractions implies the ability to 'share resources'

Types of Sharing:

- Space Multiplexed
  - Divided into 2 or more distinct units of resource
  - Example: disk, memory

- Time multiplexed
  - Exclusive control for a short period of time
  - Example: processor
Resource Sharing

- Multiple processes accessing *same* resource concurrently

- Isolation: only one processor has access at any given time
Terminology

- **Concurrency**
  - The simultaneous execution of different programs
  - **Types of Concurrency**
    - **Physical** – multiple processors
      - Example: CPU, I/O
    - **Logical** – interleaved execution
      - Example: processes
  - **Problems:**
    - Simultaneous access to memory
    - Lost updates

- **Multiprogramming**
  - The concurrent execution of multiple programs on a single processor
  - Could be space-multiplexed into memory and time-multiplexed in processors
Multiprogramming

Abstract Machine $P_i$

Abstract Machine $P_j$

Abstract Machine $P_k$

OS Resource Sharing

Time-multiplexed Physical Processor

Space-multiplexed Physical Memory

$P_i$ Memory

$P_k$ Memory

$P_j$ Memory

Machine Pi

OS Resource Sharing

Machine Pj

MachinePk

CS 3204: Operating Systems
Multiprogramming (2)

- Technique for *sharing* the CPU among *runnable* processes
  - Process may be *blocked* on I/O
  - Process may be *blocked* waiting for other resource, including the CPU
- While one process is blocked, another might be able to run
- Multiprogramming OS accomplishes CPU sharing “automatically” – *scheduling*
- Reduces time to run *all* processes
How Multiprogramming Works

<table>
<thead>
<tr>
<th>Process 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process 2</td>
</tr>
<tr>
<td>Process 3</td>
</tr>
<tr>
<td>Process 4</td>
</tr>
</tbody>
</table>

Time-multiplexed CPU

Space-multiplexed Memory
OS Strategies for Providing Services

- Batch
- Time share
- PCs and Workstations
- Process Control & Real-time systems
- Networked
- Distributed
- Small computers
Batch processing systems

- Sequentially loaded set of jobs
- Supported multiprogramming
- Jobs compete for Resources
  - 1\textsuperscript{st}: memory
  - 2\textsuperscript{nd}: processor
  - 3\textsuperscript{rd}: ???
- No “real time” interaction between user and computer
- Current examples include .bat files under DOS – Windows, shell files under Unix/Linux
Batch Processing

Job 19

Input Spooler

Input Spool

Output Spooler

Output Spool

Job 3
Time share (1970s)

- Multiprogramming environment
- Multiple interactive users

- Why time-share (TS)?
  - To spread the cost of large machine
  - To fully utilize computing power

- TS provides each user with his/her own Virtual Machine
Time share system...

Abstract Machines

Result

Command

Physical Machine

Result

Command

Result

Command
Time share... ctd.

- TS eventually supported multitasking
  - Multitasking:
    - A time share system that support multiple processes per user, where.
    - A process is a “program in execution

- TS elevated the importance of
  - Need for barriers and safeguards among users and there processes - User/User & Process/Process
    - Memory protection
    - File Protection
Personal Computers (PCs) & Workstations

- Originally
  - Single User
  - Single Processor

- Now
  - Single or Multiple Users
  - Multiprogrammed
PCs  Workstations... Evolution

- Earlier machines
  - Too large, too expensive, and too fast for one person

- Mini-computers
  - Smaller versions (like DEC PDP), yet they too grew in size

- Micro-computer
  - Single chip processor

- Workstation
  - Multiple user
  - Multiprogrammed
  - Multitasking
PCs & Workstations... Contribution

- Contributed to the growth of

  - Networking
    - Email
    - File server

  - Point and click interface
    - Like that in Mac and Windows
Process Control & Real time Systems

- Process Control Systems (PCS)
  - Single application monitoring one process
  - Example: System to monitor the heat of a liquid

- Real Time Systems (RTS)
  - Tied together Process Control Systems
Real Time Systems... type

- **Hard RTS**
  - Had timing constraints that **COULD NOT** be missed
  - Example: Chemical processes, Nuclear power plants, Defense systems

- **Soft RTS**
  - Make best effort to accommodate time constraints
  - Example: Transaction processing (ATM)

**RTS:** Tradeoff of generality of operations/functionality to ensure that deadlines can be made
Networks of Computers

- Problem is too large
  - Partition it among machines

- Communication exchange
  - Email
  - File transfers

- Servers
  - File
  - Printer
  - Database

- Provide access to non-local resources
  - LAN, WAN
  - Client / Server
Distributed OS

- Wave of the future

Multiple Computers connected by a Network
Small Computers

- PDAs, STBs, embedded systems became commercially significant
- Have an OS, but
  - Not general purpose
  - Limited hardware resources
  - Different kinds of devices
    - Touch screen, no keyboard
    - Graffiti
  - Evolving & leading to new class of Oses
- PalmOS, Pocket PC (WinCE), VxWorks, ...
The Evolution of Modern Operating Systems

- Timesharing Systems
- Network Systems
- Personal Computer and Workstation Systems
- Real-time Systems

- Batch Multiprogramming Systems
- Memory Mgmt.
- Protection
- Scheduling
- Files
- Devices

- Memory Mgmt.
- Scheduling
- Protection
- System Software
- Human-Computer Interface
- Client-server Model
- Protocols
- Scheduling

Contemporary Operating System

from the text book
Examples of Modern OS

- UNIX variants (e.g. Linux) -- have evolved since 1970
- Windows NT/2K -- has evolved since 1989 (much more modern than UNIX
  - Win2K = WinNT, V5
- Research OSes – still evolving ...
- Small computer OSes – still evolving ...