Lecture 3

SOFTWARE.
OPERATING SYSTEM.

Plan.

1. Software. Types of the software, purpose and characteristic. Basic concepts of OS.

2. Evolution of operating systems. Classification of operating systems, including for mobile devices. Classification of desktop applications.

1. Computer Software

What we'll cover for this lecture topic:

- Software categories
  - **Applications** software
  - **Systems** software
    - What is an **operating system**?
      - What does it do for me?
      - What does it do for application programs?
    - What is a **translator**?

<u>Application</u> software

○ It is the reason that one wants to buy a computer:

- printout out paychecks
- play Mortal Kombat
- keep track of a stamp collection
- do your taxes
- generate a fancy newsletter
- guide robots
- keep a budget
- draw a flowchart
- browse the Web
- design a car

- **System software**
  - Helps computer carry out its *basic tasks*.
  - Includes:
    - **Operating systems (OS)** - master control programs
      - BIOS (Basic Input/Output System)—see Rdg Handout!
      - Some utilities are built into OS
    - **Translators** (program language translators/compilers)

Application software

- Includes many *executable* files and *data* files:
  - **Installer** program   (eg: setup.exe)
  - **Uninstaller** program   *(why important?)*
  - **Main** executable file   (eg: winword.exe)
  - **Support** modules   (eg: .dll   files)
    - Called by the PROGRAM, not by the user
  - **Data** modules   (eg: MS Word dictionary)

- "**Installing**" has gotten easy…most use **wizards**
- **Excellent** coverage in the textbook Chapter 3 Section D (and an interactive lab you can try).

SYSTEMS Software...

I. The Operating System

- A type of *system* software that underlies *all other software.*
- It manages all software and hardware tasks.
- It provides a common set of **computer functions** such as input from a keyboard and output to a monitor.
- It provides the **user interface**……that is:
- How can something as *simple-minded* as a processor and memory present you with something as rich as the Mac or Windows GUI?

- What does an OS do for me? *LOTS!*

In general terms……

– Provides *user interface* …as we saw.

– Allows **applications** to run.

– What are some tasks you need done even when *no apps* are running?

– Does nearly *half* of what we ask an **application** *program* to do!

- *Common* tasks useful to all programs —put those in the OS (the *most basic* are put in BIOS) so each app doesn't have to handle those tasks.
- So what are some of these tasks?

System calls

- You ask **MS Word** to OPEN a document
  - File menu/Open…
  - WHAT WILL YOU SEE?
  - What *really* just happened *in the box*?

WORD called on **O.S.** to present you with that file list!

O.S. must look at:

1. Disk Directory (list of *filenames*)
2. File Allocation Table (**FAT**)…. (list of file *locations: starting cluster number on the disk)*

Allows ***consistency*** from app to app…

Input/output

Reads from and _writes to_ the I/O devices.

- In the past, **application** programmers had to write control programs for I/O devices. Painful!

- Today, **O.S.** _reads from_ and _writes to_ the I/O devices: mouse, keyboard, printer, monitor…

- About **half** the instructions in today's OS are to manage input and output operations.

EXAMPLES of I/O operations

- OS reads mouse **movement** and writes to display screen.
  - ◦ You *move the mouse*---what do you expect to happen?
  - ◦ What that involves…
    - Manage interrupt.
    - OS reads mouse wheels.
    - OS draws cursor arrow (changes pixel colors so arrow *appears* to be "moving").

- OS **identifies selected objects** on the desktop
  - You *select an icon*--what do you expect to see?
  - What that involves…!
    - FIRST: you move the cursor to point to icon…
      - Draws "moving cursor…"
    - OS keeps **table** of icon placements…
    - Looks at current *cursor placement* and compares it to that table.
    - NEXT: you single-click mouse button to SELECT it:
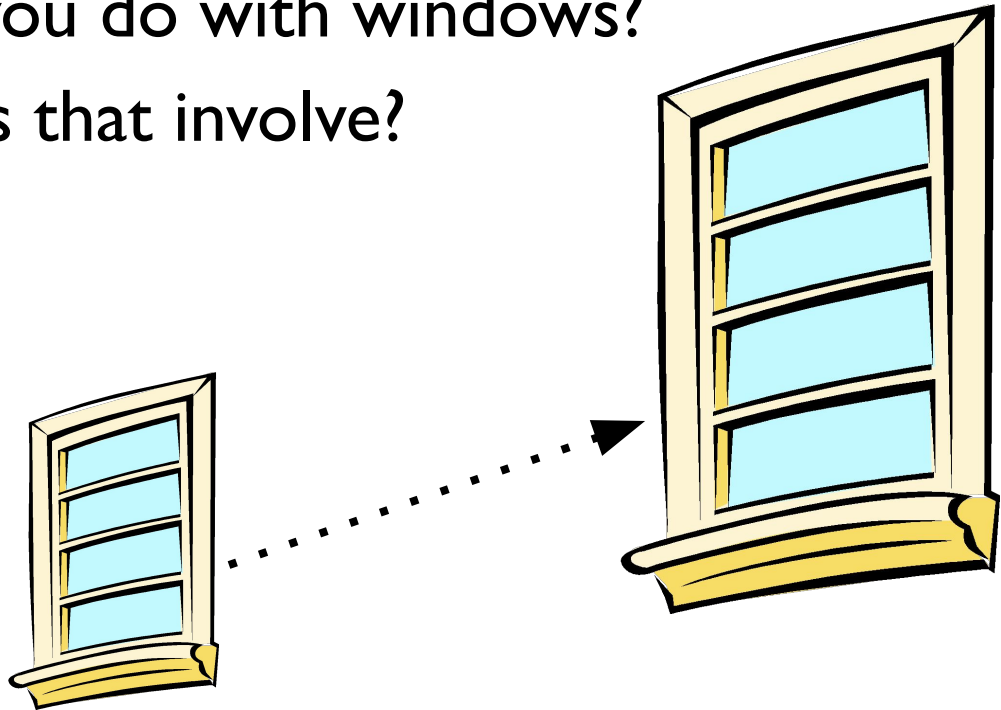   - What do you expect to see?
      - OS does all this!

Manages windows

◦ What do you do with windows?

◦ What does that involve?

Manages files and folders

– *What do **you** do with files and folders?*

– You **Create** a new file or folder….

– You **Move** files and folders; you "nest" folders

– You **Open** a document file… whoa!   Let's see:
  – OS looks at <u>file extension</u>
  – OS checks if enough <u>free memory space</u>
  – **OS finds** and **loads** the APP *(if not loaded)*
  – *OS finds and loads the **document***
  – OS keeps track of what <u>**data**</u> goes with what <u>**program**</u> (all sharing same RAM)
  – OS turns **control** over to the APP

Misc Services and Utilities

- **OS** does **system control options** from **Start button:**
  - Shut down; Restart.

- **OS** does **universal ops** from the **Edit menu**:
  - Cut, Copy, Paste, Clear, Select All
  - Clipboard … (also between different apps)

- **OS** does **universal ops** from the **<u>View</u> menu**:
  - Show/Hide Toolbars & Status bar;
  - Large icons, Small icons;
  - List; Details;
  - Arrange Icons etc.

- **OS also** has many built-in **UTILITIES** & goodies that are universally provided: *(**differs** from OS to OS; and version to version)*
  - **Taskbar** and **Start** button: unique to Win O.S.
    - Control panel, Find, Help, Format or Erase disk, Properties
  - **Right**-click menus
    - **Properties**; Rename; Shortcuts, and more
  - And lots more, depending on **OS** you use, and on the **version** you have.

- Lots of *3rd-party* **utilities** as well!

READ about these utilities in the Reading handout.

- Data recovery
- Compression *(NOTE: textbook pages 371-375—just know the general gist of how it works, not all the specifics!)*
- Anti-virus protection *(included with Windows XP)*
- Firewalls *(included with Windows XP)*
- Diagnostics
- Uninstall programs
- Screen savers
- File defragmentation
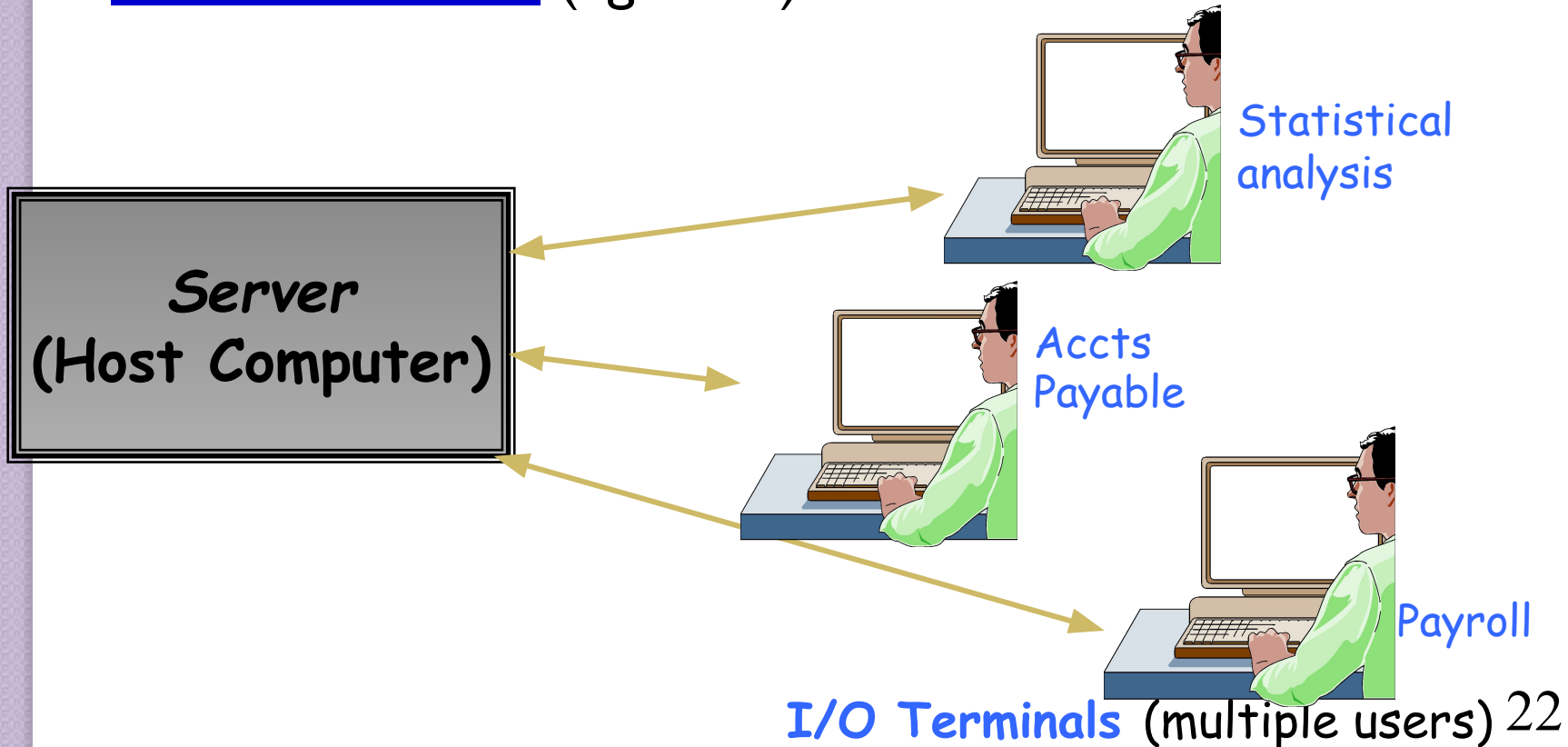- *and MORE!*

Manages/allocates time & memory space

- **>1** program *or* person can share computer **resources**.
  - CPU is idle ~90% of time, waiting for user input !
 - **Multiprogramming**
    Run two or more programs *concurrently*.
    eg: Calculate payroll *and* Accts payable *and* WP *and* …
    1. **Multi-user OS** (eg: Unix).

Server
(Host Computer)

Statistical
analysis

Accts
Payable

Payroll

I/O Terminals (multiple users) 22

2. Single-user OS (eg, Windows)

- Referred to as **multitasking,** which is multiprogramming for *single*-**user OS**.

- One active app; others run in the background

PC

(1

Payroll
Email
Web browser
Word Processing
Mortal Kombat
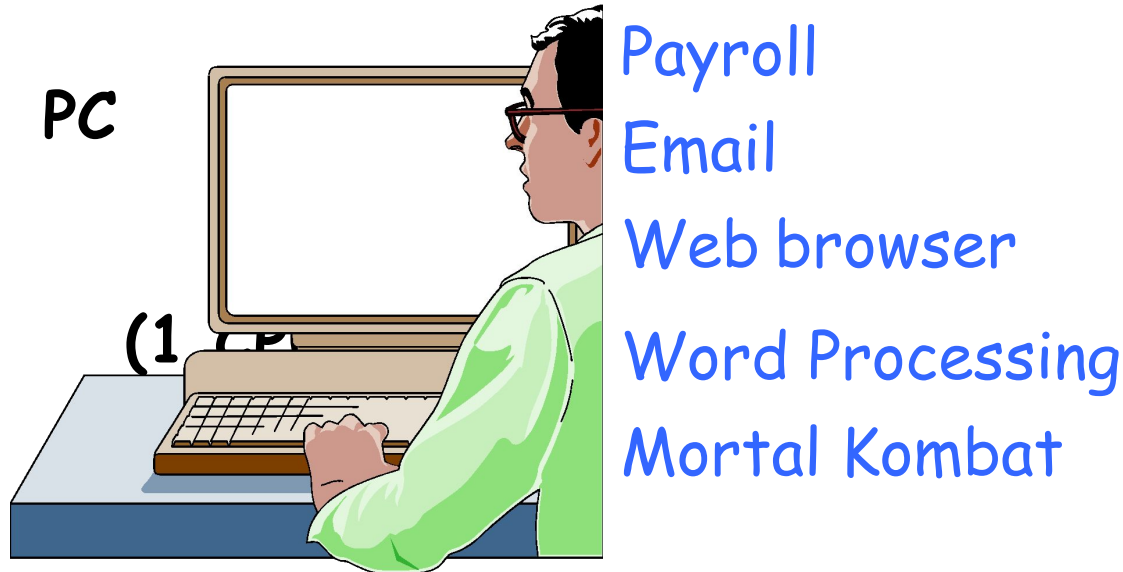
OS must protect each program's *memory area* to ensure that instructions and data don't "leak" into an area allocated to *another* program.   If it fails, programs can **crash**…more shortly!

Handles interrupts

- Mouse click; mail sound; alarm clock; app bombed...
  - OS breaks into current process and instructs CPU to do something else. *And keeps track!*

Important Digression: software bombs

- **APP** freezes**--**"Program crash"

  -- Windows OS usually allows you to continue working in other apps; try to close the confused app:

  Right-click on the app's button on Taskbar, select Close.

- **OS** freezes**--** "System crash"

  -- Ctrl/Alt/Del: sometimes can Cancel current Task (Applications Tab, select End Task). If that fails, restart ("**Soft** boot") from Start button.

  -- Power off button, wait, then Power on ("**Hard** boot"). *LAST RESORT!*

Provides (and loads) Device Drivers

- **Small programs that control a peripheral device** (printer, hard disk, tape drive, modem …)

  – Allow OS & applications to activate *(drive)* the hardware device.

  – The driver accepts commands from the operating system and converts them into a form that a *particular device* can understand.

  – Newer OSs: provide ***most*** device drivers.

  – Else: find and download device driver program from manufacturer's web site.

SYSTEMS Software...cont'd

II. Translators *(revisited!)*

- How do people write programs?

- Only language a **computer** understands?

- **A translator** (or **compiler**):

  - Program that converts high-level *source code* into low-level *machine language (object code)*-- can then be processed directly by the computer's binary circuits.

  - Running a **source** program is a **two-step process**:

    1. Execute the translator program first:
       - converts ASCII source into executable machine language
       - creates a new file containing the object code.
    2. Execute that NEW object code file.

In ENGLISH:

Find and print the names of all freshmen who scored greater than 79% on the first exam.

High Level: If Year = 1 and Score1 > 79 then put StName

Low Level:

Assembly: LDR A5FD R1   *More readable form of binary;*
        CMP R1, 1 …   *symbolic representation.*

ML: 00000010 10001100 01100000 00010001
     00000111 …

Translation will:

– chop up every *command* word into ~25+ *op codes*.
– convert *variable names* (Year, Score, StName) into actual binary memory *address* numbers.

2. Operating Systems

What is an Operating System (1)?

- A modern computer consists of:
   - One or more processors
   - Main memory
   - Disks
   - Printers
   - Various input/output devices.
- Managing all these varied components requires a layer of software – the **Operating System (OS).**

What is an Operating System (2)?

- An Operating System is a program that acts as an intermediary/interface between a user of a computer and the computer hardware.

- OS goals:
  - Control/execute user/application programs.
  - Make the computer system convenient to use.
  - Ease the solving of user problems.
  - Use the computer hardware in an efficient manner.

Where does the OS fit in?
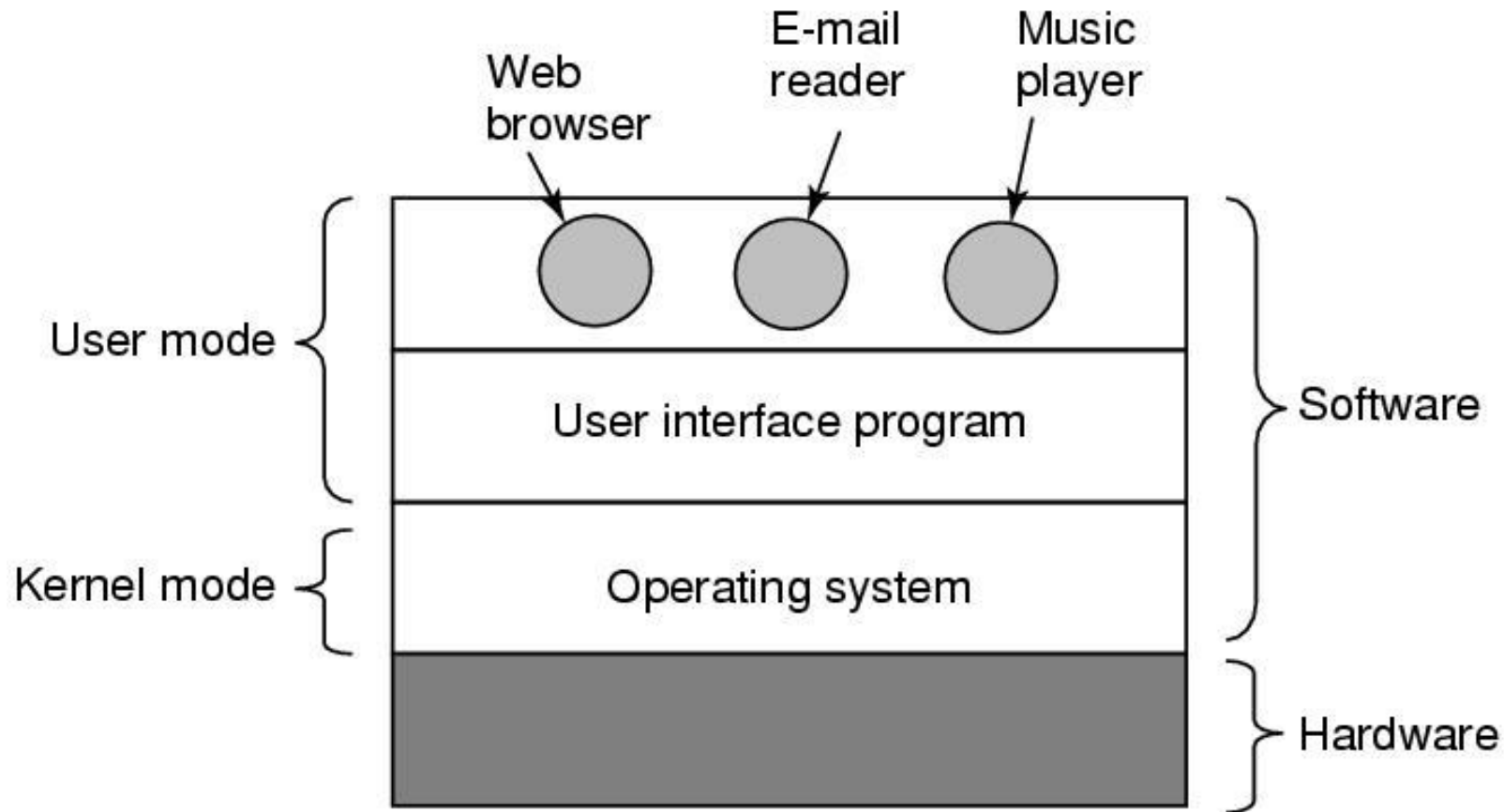
- Facilities for program creation
  - editors, compilers, linkers, debuggers, etc.
- Program execution
  - loading in memory, I/O and file initialization.
- Access to I/O and files
  - deals with the specifics of I/O and file formats.
- System access
  - resolves conflicts for resource contention.
  - protection in access to resources and data.

Why are Operating Systems Important?

- Important to understand and know how to correctly use when writing user applications.
- Large and complex systems that have a high economic impact and result in interesting problems of management.
- Few actually involved in OS design and implementation but nevertheless many general techniques to be learned and applied.
- Combines concepts from many other areas of Computer Science: Architecture, Languages, Data Structures, Algorithms, etc.

Evolution of Operating Systems

- The evolution of operating systems is directly dependent to the development of computer systems and how users use them. Here is a quick tour of computing systems through the past fifty years in the timeline.

Early Evolution

- 1945: ENIAC, Moore School of Engineering, University of Pennsylvania.
- 1949: EDSAC and EDVAC
- 1949 BINAC - a successor to the ENIAC
- 1951: UNIVAC by Remington
- 1952: IBM 701
- 1956: The interrupt
- 1954-1957: FORTRAN was developed

Operating Systems by the late 1950s

- By the late 1950s Operating systems were well improved and started supporting following usages :
- It was able to Single stream batch processing
- It could use Common, standardized, input/output routines for device access
- Program transition capabilities to reduce the overhead of starting a new job was added
- Error recovery to clean up after a job terminated abnormally was added.
- Job control languages that allowed users to specify the job definition and resource requirements were made possible.

Operating Systems In 1960s

- 1961: The dawn of minicomputers
- 1962 Compatible Time-Sharing System (CTSS) from MIT
- 1963 Burroughs Master Control Program (MCP) for the B5000 system
- 1964: IBM System/360
- 1960s: Disks become mainstream
- 1966: Minicomputers get cheaper, more powerful, and really useful
- 1967-1968: The mouse
- 1964 and onward: Multics
- 1969: The UNIX Time-Sharing System from Bell Telephone Laboratories

Supported OS Features by 1970s

- Multi User and Multi tasking was introduced.
- Dynamic address translation hardware and Virtual machines came into picture.
- Modular architectures came into existence.
- Personal, interactive systems came into existence.

Control questions

- What is Software?
- Differentiate System software and Application software.
- What are the responsibilities of Operating Systems?
- Define the following with suitable examples.
- Single–user OS
- Multi-user OS
- What are utility programs? Define some tasks performed by them.
- What is meant by library programs?
- What are program language translators? Briefly describe three translating approaches.
- State the advantages and disadvantages of Bespoke Application Software.