CS101 Introduction to Computing

Lecture 11

Operating Systems

Focus of the last lecture: computer SW

1. We found out about the role SW plays in a computing environment

2. We learned to distinguish between SW belonging to the system & application categories

3. Also discussed the different types of SW licenses:
    1. Proprietary
    2. Free
    3. Open source
    4. Shareware
    5. Trialware

Learning Goals for Today

- The role of the operating system in a computing environment
- The various functions that an operating system performs
- The main components of an operating system
- Various types of operating systems

Why Have OSes?

1. User/programmer convenience

2. Greater resource utilization

The Role of An OS

- The 1st program that runs when a typical computer is turned ON, and the last one to finish running when the computer is turned OFF

- It manages the HW and SW resources of the computer system, often invisibly. These include the processor, memory, disk drives, etc.

- It provides a simple, consistent way for applications to interact with the HW without having to know all the details of the HW

Advantage for App. Developers

- App developers do not need to know much about the HW while they are developing their app

- They just develop with a particular OS in mind. If the OS runs on many types of computers having different HW configurations, so will the app – without making any HW-specific modifications in the app SW. The OS hides the HW differences from the app

Are OS'es Essential?

- No. If a computer has been designed for limited functionality (e.g. it runs just a single program all the time as in a automatic clothes washing machine), it does not require a traditional OS
- In limited-functionality computers, an OS just adds to the overhead unnecessarily, which impedes the computer's performance
- In these situations, the required parts of the OS are integrated into the the only program that is going to run

In the beginning …

- A single user ran a single program ran on a single computer – there was no need for an OS
- Then came computer operators who ran multiple programs for multiple users one after the other – still, no need for an OS
- Later computers became powerful, & became able to run multiple programs, simultaneously. That's when the need for OS'es arose for:
  - Managing the resources of the computers efficiently
  - Making use of computers convenient for users/programmers

Core Tasks of an OS

1. Processor management
2. Memory management
3. Device management
4. Storage management
5. Application Interface
6. User Interface

Processor Management

- Various programs compete for the attention of the uP for their own purposes
- The OS plays the role of the honest referee, making sure that each app gets the necessary attention required for its proper execution
- It tries to optimally manages the limited processing capacity of the uP to the greatest good of all the users & apps

Memory Management

- Straight forward for a single-user, single tasking
- Each app must have enough private memory in which to execute
- App can neither run into the private memory space of another app, nor be run into by another app
- Different types of memory (e.g. main, cache) in the system must be used properly, so that each app can run most effectively

Storage Management

- The OS manages storage through one of its sub-modules, the File Manager
- A file system is a collection of directories, subdirectories, and files organized in a logical order
- File manager maintains an index of the filenames & where they are located on the disk
- File manager make it easy to find the required file in a logical and timely fashion

Device Management

- Applications talk to devices through the OS and OS talks to and manages devices through Device Drivers

- Example: When we print to a laser printer, we do not need to know its details. All we do is to tell the printer device driver about what needs to be printed and it takes care of the details

Application Interface

- App developers do not need to know much about the HW, especially the uP, while they are developing their app
- The OS provides all apps with a straight-forward and consistent interface to the HW
- Example: An app uses the OS to store data on the disk drive. For that, the app does not need to know about the exact physical characteristics of that drive; it just tells the OS to do that through the app interface, and the OS takes cares of all the details of the task

User Interface

- Users communicate with the computer using a consistent user interface provided by the OS

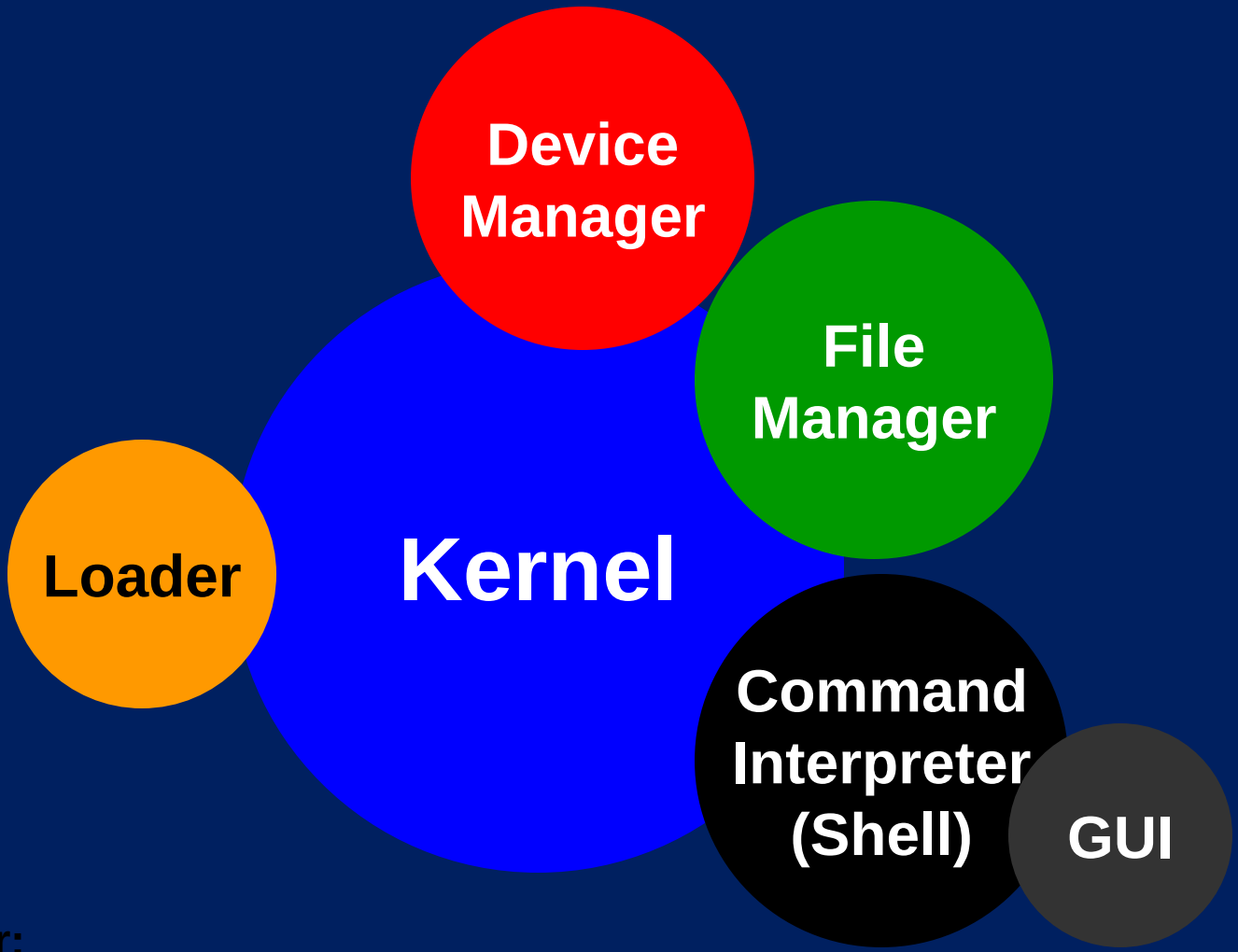
- This UI can be a command-line interface in which a user types in the commands. Example:

  `copy a:/file1.html c:/file1.html`

- Or, it can be a graphical UI, where Windows, Icons, Menus, and a Pointing device (such as a mouse) is used to receive and display information. Example:

  With the help of the mouse, drag `file1.html` from drive `a` to drive `c`

OS Components



Loader:
When you turn on a computer, first of all, Loader is the component of OS which comes into action. It checks is the hardware ok? Then it searches the main part of OS called as Kernel. As its name implies it loads the Kernel into memory.

Kernel

- The heart of the OS
- Responsible for all the essential operations like basic house keeping, task scheduling, etc. Also contains low-level HW interfaces
- Size important, as it is memory-resident

Types of OS’es

Classification w.r.t. the type of computers they run on and the type of applications they support

- Real-Time Operating System (RTOS)
- Single-User, Single Task
- Single-User, Multi-Tasking
- Multi-User

RTOS (1)

- Used to run computers embedded in machinery, robots, scientific instruments and industrial systems

- Typically, it has little user interaction capability, and no end-user utilities, since the system will be a "sealed box" when delivered for use

- Examples: Wind River, QNX, Real-time Linux, Real-time Windows NT

RTOS (2)

- An important part of an RTOS is managing the resources of the computer so that a particular operation executes in precisely the same amount of time every time it occurs

- In a complex machine, having a part move more quickly just because system resources are available may be just as catastrophic as having it not move at all because the system was busy

Single-User, Single Task

- OS’es designed to manage the computer so that one user can effectively do one thing at a time

- The Palm OS used in many palmtop computers (PDA’s) is an example of a single-user, single-task OS

Single-User, Multi-Tasking

- Most popular OS
- Used by most all PC's and Laptops
- Examples: Windows, Mac OS, Linux
- Lets a single user interact with several programs, simultaneously

Multi-User

- A multi-user OS allows many users to take advantage of the computer's resources, simultaneously
- The OS must make sure that the requirements of the various users are balanced, and that the programs they are using each have sufficient and separate resources so that a problem with one user doesn't affect any of the other users
- Examples: Linux, Unix, VMS and mainframe OS’es, such as MVS

Another Way of Classifying

Uni-processor OS’es

Designed to schedule tasks on a single uP only

Example: DOS

Multi-processor OS’es

Can control computers having multiple uPs, at times 1000’s of them

Example: Current versions of Windows, Mac OS, Linux, Solaris

How many different OS’es are there?

- 100’s
- OS’es from the Windows family dominate the desktops and run on millions of PC’s
- OS’es from the Unix family (Unix, Linux, etc) are quite popular on servers
- There are hundreds more. Some designed for mainframes only. Some for embedded applications only.

Comparing Popular OS'es

OS	*HW*	*Stability*	*Cost*	*Apps.*	*Support*	*Security*	*Popularity*
Windows (GUI)	PC	Poor	$300	Huge no.	OK	Poor	Amazing
Mac OS (Shell/GUI)	Mac	Good	$60	Many	OK	Good	Low
Linux (Shell/GUI)	Many	Good	Low	Many	Variable	Good	Low
Unix (Shell/GUI)	Many	Excellent	High	Many	Expensive	Excellent	Servers

What have we learnt today?

- The role of the OS in a computing environment
- The various functions that an OS performs
- The main components of an OS
- Various types of OS’es

Next Lecture: Application SW

We'll learn about application SW, i.e. programs that interact directly with the user for the performance of a certain type of work

We'll try to become familiar with various SW used in the following application areas:

- Scientific/engineering/graphics
- Business
- Productivity
- Entertainment
- Educational