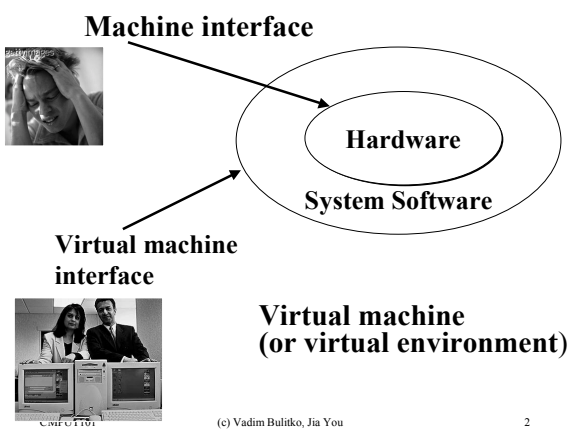
Chapter 6.4 : Operating Systems

Outline:

- Functions of an OS
- Historical Overview
- The Future



Typical Types of System Software

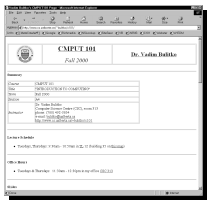
- Language translator
  - assemblers, compilers, interpreters, …
- Memory manager
  - allocate space for program execution
- File system
  - storage and retrieval for mass storage devices
- Scheduler
  - select a task to run
- Utilities
  - a collection of programs that provide services

Operating System Functions

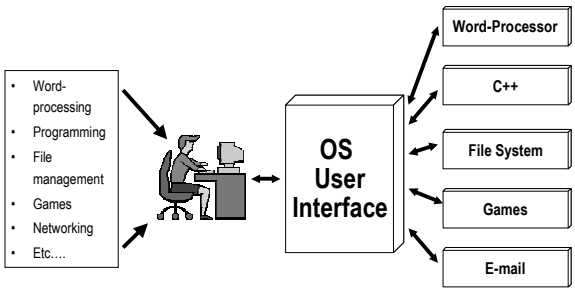
- User Interface
- System Security & Protection
- Efficient Allocation of Resources (including file system)
- Safe Use of Resources

User Tasks

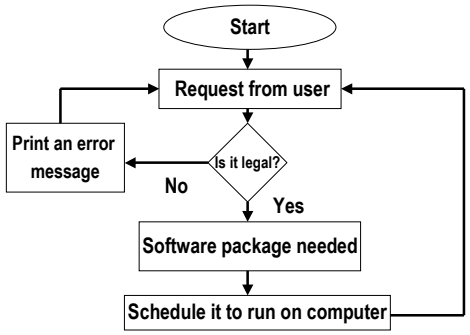
- Word-processing
- Programming
- File management
- Games
- Networking
- Etc.…



OS User Interface



User Interface



Types of User Interfaces

- Command language (e.g. Unix)
  by a prompt character

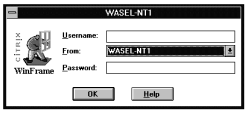
  > cd /usr/prof/you/cmput101

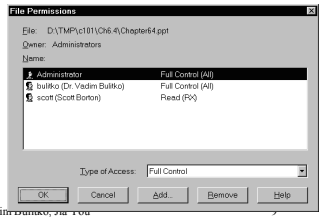
- Graphical user interface (e.g. Windows)
  uses icons, pull-down menus, scrolling windows

- Which one is superior?

System Security & Protection

- Log in permissions
- File permissions
- System access levels
- Encryption





Encryption

- You may see it, but you don't know what it is.

  A text can be encrypted by performing some sequence of mathematical operations

E.g.

01000001 01000010 01000011

A B C

After some operations, e.g. left-shifting, they will become something else other than ABC. One must know what operations have been performed in order to know what the original text is.

Why Encryption Works (in theory)

- Without a key, a correct guess is almost impossible

E.g. Given an extremely large number N that is the multiplication of two prime numbers,

- if we know one of the prime numbers, its easy to figure out the other
- otherwise it can take millions of years for the fast computer to generate the two prime numbers.

- In reality, no matter how sophisticated an encryption algorithm is, it seems someone could break it.

Efficient Resource Allocation

- Consider the following code:

```
While j < 10
    While (PrinterStatus == 'busy')
        Wait
    Print page_j
    j = j + 1
```

- Processor time for the loop is 1/1000 second
- Printing time for a page is 1 minute
- How much idling will the processor do?

Solution

- Execute another program while the first program waits for the printer
- The same approach is used for all input/output (I/O) waits: printer, display, hard-drives, network, etc.

Time Sharing

There are many programs that need to run at a given time:

E.g. Editing using Microsoft Word
Surfing the net using a browser
Compiling a C++ program

But there is only one processor on a computer

Program Queues

Programs that are requested to run are divided into

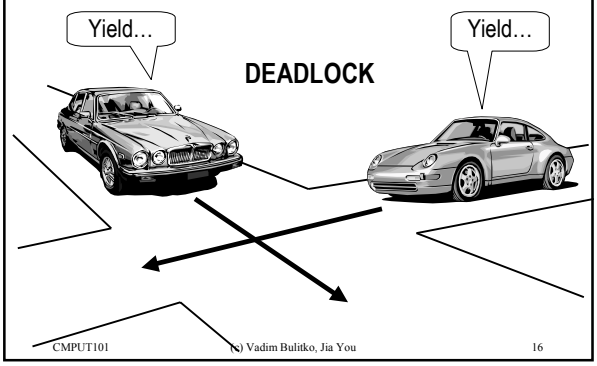
Running: the program executing on processor
Ready: programs that are loaded to RAM and ready to run
Waiting: programs that cannot run, waiting for I/O or some other time consuming event

E.g. in C++ when your program contains

```
cin >> A;
```

processor won't sit idle waiting for the user to enter something, if there are other tasks to run

Safe Resource Allocation & Deadlocks



Deadlock: another example

- John borrowed book A from the library.
  - John is holding book A but also needs book B to complete his assignment
- Marry borrowed book B from the library.
  - Marry is holding book B but also needs book A to complete her homework

This is a deadlock situation. As a result, no one can complete his/her work.

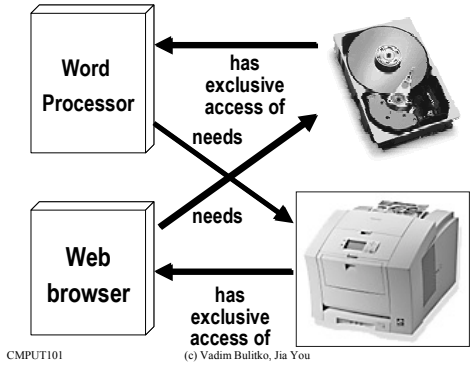
Deadlock

A set of programs each of which is waiting for an event to Occur before it may proceed, but that event can be caused only by another waiting program in the set.

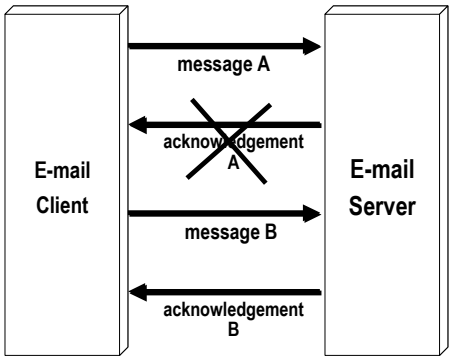
Example.

	Program A	Program B
Holds	Get disk drive	Get laser printer
Requests	Get laser printer	Get disk drive

Deadlocks



Deadlock could occur in telecommunication



Deadlock Solutions

- Prevention
  - Give every program all resources or none
- Deadlock Recovery
  - If a program cannot get all it needs, it must give up all resources it currently owns
- In the example of telecommunication, resend messages if no acknowledgement is received within so many seconds

OS History

- 1st generation (1945-1955): none
- 2nd generation (1955-1965): batch OS
- 3rd generation (1965-1985): multi-prog. OS
- 4th generation (1985-now): network OS, GUI

The Future of OS…

- Extensive multimedia Interfaces (sound, graphics, video, 3D, voice-recognition, tactile input devices, etc.)
- Parallel processing (a multitude of processors on a single computer)
- Massively and transparently distributed (extensive networking, wireless, fiber-optics)

Summary

- OS is a part of system software
- Functions:
  - User Interface
  - System Security & Protection
  - Efficient Allocation of Resources
  - Safe Use of Resources
- History of OS
- The Future…