WEB SYSTEMS

LECTURE 1

INTRO

What is an operating system

*An operating system is a piece of **software** that sits between all programs and the computer's hardware.*

E.g. windows, linux, os x

- Manages your computer
- Runs programs, applocations
- Interface between user and hardware
- Provides services to programs & users - accounts
- Protects users and programs from each other

COMMON OPERATING SYSTEMS

LARGE systems

Mainframes pioneered OS since the 1960's

- IBM still run mainframes – z/OS will still run ancient programs!
- Handles 1000's of users simultaneously
  - IBM Mainframes also run Linux!
- Supercomputers now tend to run Linux

Minicomputers

- openVMS, IBM OS/400
- UNIX & Unix-like OS
  - e.g. Linux, BSD, Solaris, HP-UX, Mac OSX

Personal Computers

- Linux
- Microsoft Windows, Mac OS/X (actually Unix!)
- Anyone heard of BeOS? CP/M? Amiga?

Embedded systems

- Military, telecommunications, etc
  - Eg: VxWorks, QNZ, Windows CE, Linux

AN OS ARCHITECTURE: THE UNIX "ONION" MODEL

Hardware = CPU, memory, input/outputs of mouse, display, printer, storage of hard drive, flash

Kernel =

- controls the hardware directly (device drivers, firmware)
- Provides resources and services to applications

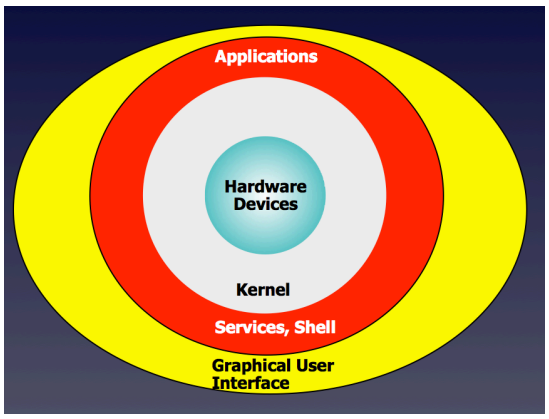
(CPU, memory, storage, video, mouse , keyboard, memory),

- Manages access to privileged resources

Top layers:

Sometimes called "***userland***", "user space", "application layer"

- **Applications**
  - Programs to do "something" for the user.
- **Services**
  - Services are programs that run "behind the scenes"
    - usually provides system support
      - eg: security, networking



COMPUTER INTERFACES

Top most layer: "**User Interface**"

Shell

- Also known as **Command Line Interface (CLI) (or Interpreter), Command Prompt, Terminal etc**
- A program that makes a set of commands available to the user

Graphical User Interface (GUI)

- A user-friendly interface on top of the operating system
- Often runs the "shell" commands transparently
- Sometimes a WEB interface

CLI & GUI

The Command Line Interface (CLI).

Interact through the keyboard and a monitor which only prints text.

- sh 1969: predecessor of bash, csh
- CPM 1973: predecessor of MS-DOS
- cmd.exe – windows shell

The Graphical User Interface (GUI).

Interact via windows, icons, menu, pointer device *called "WIMP" interface*

1983: Apple – Lisa, Mac OS
1984: Unix - Gnome, KDE
1985: Microsoft – Windows 1.0
2001: Apple – Mac OS X &
Microsoft Windows XP
2006: Microsoft Vista Aero

GUI VS CLI

- Each of them has an appropriate and important role in computing.
- Both the CLI and the GUI have their strengths and weaknesses.

Multiple interfaces

- Customization
- Automation
- Understanding

Graphical User Interface

Strengths

- Little/no experience required
- Good for graphics e.g.

Command Line interface

Strengths

- Greater flexibility
- Fine tuning, → parameters
- Essential for system administration
- Faster, less overhead
- Runs on simple hardware.
- Can run remotely
- Robust - Difficult to crash.

Weaknesses

- Hard to learn
  → cryptic commands & parameters
- Multiple options
  → more than 1 way to do things
- Output often cryptic or non-existent
- Inconsistent commands
  → different versions of Unix.
- No graphics
- No safety net
  → 'expert mode'

→ artwork, desktop publishing,

- User friendly, **intuitive**
- Hides complexity from users.

Weaknesses

- Can't do everything.
  → Using keyboard can be faster.
- Can crash the system!
- User is unsure of what the O/S is **really** doing.
- Slows computer down
- Needs better hardware
- Hides complexity from users.

Batch files and scripting languages

Automate CLI's through batch files (a computer file containing a list of instructions to be carried out in turn)

- You can put a sequence of commands into an executable file
  → CLI treat's the file as a command.
- Most CLI's include **programming** features
  → logic, calculations, variables, user input…
- Some GUI's also have batch facilities thus is a SCRIPTING LANGUAGE

Examples of scripting languages

- Bash, Korn Shell, C shell, Z shell
- DOS Batch Language
- WMI (Windows Scripting Language), VBScript
- JCL (Job Control Language) (used in Mainframes)
- Applescript

Characteristics of scripts

- Variables are usually untyped
  (called "loosely bound")
  → the same variable can be used as a number or a string.
- Language syntax is often inconsistent.
- Often designed and created by one person to get a particular job done.
- Usually run through an interpreter, not a compiler.

Evolution of scripting languages

- Scripting Languages tend to gain extra features as they evolve.
- **Perl** is a good example of this.
  - Started as a scripting language
  - now almost a generic programming Language.
- **Windows Shell** replaced by powershell
  - "Real" programming features
  - **Bash** (Linux default CLI) also evolved.
  - Now includes arrays, data types etc