CMSC131

Hardware and Software

Hardware

Hardware is what makes up the physical machine.

Two of the same “type” of machines can have different brands of hardware used to build them.

- eg : Hayes Modem -vs- US Robotics Modem -vs- Zoom Modem
- Intel processor -vs- AMD processor

Different types of machines can have some hardware components in common.

- eg : The **PowerPC** chip was meant to be used in both "Windows" type machines (sometimes called IBM-compatible) as well as Apple Macintosh machines. Intel's **Dual-Core** technology started being used in Macs as of early 2006.

Machines might have a common hardware interface to allow for peripheral devices to be used on different platforms.

- eg: Both PCs and Macs now have USB ports, and with the appropriate software, an external device that uses a USB connection (such as a printer, camera, or MP3 player) can be use across platforms.

Hardware might have similar names, yet be very different from each other (Telephone Modem -vs- Cable Modem). Some software is designed to only work with certain brands of hardware.

"Standardized" devices might even have different specifications (iPhone screen resolutions now differ).

Operating Systems

The Operating System is the piece of software that runs and controls the computer hardware. It acts as an interface between application programs and the hardware being used to run them.

Every physical machine has a set of operating systems that can be installed onto it. There is usually a “typical” operating system.

- eg : · "PC" machines (still sometimes called IBM-compatible) will typically run some version of Windows
- Macintosh systems typically run a version of MacOS
- Sun stations typically run a version of UNIX

However, a single machine might be able to run different operating systems. For example, the so-called "MacTel" architecture is able to natively support both OS X and Windows XP (though a user needs to boot to one or the other at a given time). There are UNIX-family operating systems that can be run on an "IBM-compatible PC" (mostly various flavors of Linux) and versions that can be run on a Macintosh. To add to the puzzle, the current Macintosh operating system (OS X) is actually built on top of a UNIX foundation raising the question of whether OS X is an operating system or a shell.

You can also purchase software that will allow you to emulate one piece of hardware on another platform (for example, VirtualPC and Parallels are products that runs on the Apple Macintosh platform and allows you to have a Windows XP environment within a window on your Mac). These are often referred to as "Virtual Machines".

Software

- There are many different types of software. An operating system is a type of software known as *systems software*. What is most commonly referred to generically as "software" is more accurately called *application software*.
- Word processors, spreadsheets, database management systems, web browsers, etc... are examples of applications software.
- Within application software, the distinction between *purpose* and *brand* is an important one. There might be several applications (each sold by different companies) that can equally be used to create a CDR.
- The appearance of software that we interact with essentially falls into two categories today; text-based and graphical. Most applications we use today are often referred to as Graphical User Interfaces (GUIs). However, there are still a variety of text-based systems in use. We will explore text-based interactions with the Windows, Mac OS X, and UNIX operating systems. Another example of a text-based system might be your cell phone.
- Some software is "open source" software. With these applications, the source code (written in languages like C or Java) is freely posted and "anyone" can compile it for their machine (or even modify the program if they wanted to). Precompiled versions are typically made available and can even be made available as an installation package just like for-sale apps.

Some Hardware Details

- **C**entral **P**rocessing **U**nit (eg: Intel Duo Core)
- Main Memory (eg: 1GB DIMM)
- Secondary Storage (eg: hard drive)
- **I**nput/**O**utput Devices (eg: keyboard)
- Networking (eg: wifi card)

Information Storage

Information is stored and transmitted in binary representation on hard drives, CDs, DVDs, memory cards, solid state drives, networks, etc.

We will represent binary values as 0 and 1 in class, but they can also be represented in other ways based on the properties of the physical medium such as "on -vs- off" or "North/South -vs- South/North" or "pit -vs- no pit“ or "low -vs- high tone".

Some common standardized terms:

- BIT = single binary value (short for **B**inary dig**IT**)
- Byte = group of 8 bits
- Word = unit of memory (commonly 4 bytes)
- Kilobyte (K) = group of 1024 (2^{10}) bytes

Sometimes things are discussed in terms of bits, such as Internet bandwidths being expressed in terms of megabits per second, in which case it is essentially a measurement of millions of bits per second.

Some measurements use base 10 powers (1000^3 is a gigabyte but 1024^3 is a gibabyte).

Information in Main Memory

- Can think of it as a table of bytes where each byte has an address.
- Each byte can store Base 2 values between 00000000 and 11111111 (so Base 10 values 0 to 255).
- The range of addresses supported by your OS limits how much memory you can install.
- Everything with computers must get converted to Base 2 (Base 10 integers, floating point numbers, characters, pictures, etc).

Address	Value
47892	01010101
47893	11001100
47894	10010111
47895	00001100

Characters in Base 2?

- The typical way to represent a character (like “A”) is to have a standard conversion table that assigns a number to each characters.
- ASCII (American Standard Code for Information Interchange) is one:
  - (“A” = 65 = 01000001)
- Unicode is another:
  - (“ 蓝” = 34013 = 1000010011011101)

Hardware/Operating System/Software

- How do these three interact with one another?
- The hardware is what executes the commands which results in what you see on the monitor, get on the printer, etc...
- The application software has the instructions that, if executed according to the logic designed into the software, will ***do*** something.
- The operating system acts as the go-between for these two.

- It is worth noting that when you purchase a new peripheral device, you sometimes need to install both software and **device drivers** in order for the operating system to be able to communicate with them. In essence, the drivers "teach" the operating system about the new piece of hardware (which might not have even been imagined when the operating system was written) so that when the software asks the OS to interact with the hardware, it is able to do so.

When you double-click…

- As an example, when you double-click an application icon, the operating system will begin to read that application's code into main memory and then execute the code (instruction by instruction) on the CPU.
- Things to consider in the future:
  - What if an instruction says to use a peripheral?
  - What if the operating system is configured to use virtual memory?
  - What if the application code isn't in the same language that your CPU speaks?

Programming Language “Generations”

- 1st Generation: machine code (0s and 1s).
- 2nd Generation: assembly language (slightly more human-readable) and that was converted into machine code by a program.
- 3rd Generation: more human-friendly languages (Fortran, COBOL, Lisp, C, C++, Java).
- 4th Generation: application/domain-specific languages (SPSS, SQL, Mathematica).
- 5th Generation: constraint/logic programming.

Code Examples

- 1st Generation
  - 000000 00001 00010 00110 00000 100000
- 2nd Generation
  - addcc %r3,%r5,%r3
- 3rd Generation
  - x = Math.pow(2,10);
- 4th Generation
  - SELECT * FROM Book WHERE price > 100
- 5th Generation
  - solve(A) :- clause(A,B), solve(B)

Compilers and Interpreters

- For anything other than first-generation code, a program is used to convert the code that you write into machine code.
- Compilers are used to do this conversation all in advance (ie: compile an executable). Once the executable exists for an operating system or virtual machine, it can typically be run without any special program on a specific machine.
- Interpreters are essentially used to do this conversion in real-time as the program is being run.
- There are sometimes “blurred lines” between the way these two types of tools are used.

Object Oriented Programming

- Object
- Class
- Method
- Statements
- Main Method

We will be seeing all of these this semester and discuss what they are and how they are used as we learn Java programming.