# Computer Architecture

January 31, 2023

# Administrative Notes

# An overview of computer systems

Hardware and Software

- ***Hardware:*** those physical parts you can reach out and touch
- ***Software***: the abstract logic that you can't see or touch; it executes on the hardware

Software Hierarchy

    Operating Systems

    Applications

Networks and Networking

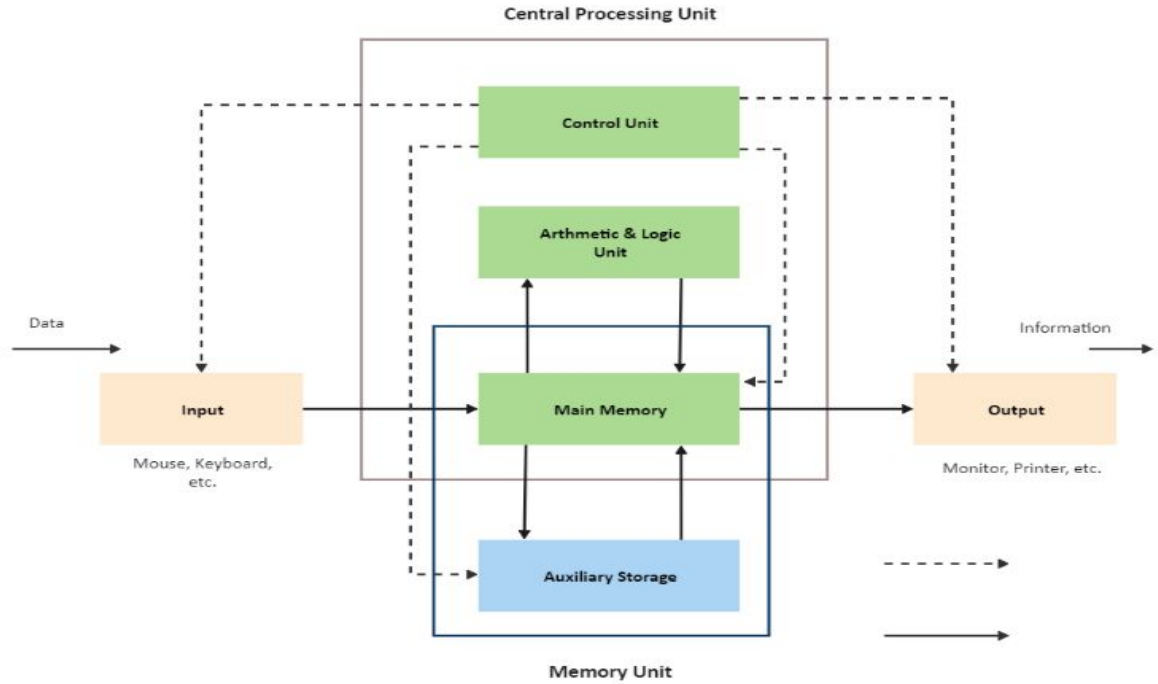
    The Internet

    The World-Wide Web

    Web Services

Cloud computing

# Typical Computer architecture




***Phone, tablet, laptop, desktop - it really doesn't matter; the ideas are the same***

From *https://www.edrawsoft.com/article/computer-architecture.html*

# Networks, Clouds and Virtual Machines

A network is a bunch of computers connected together to improve productivity

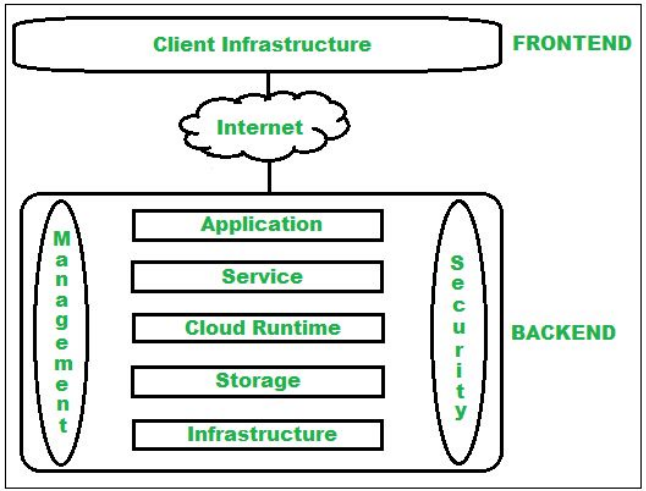
- Exchange information
- Work together to solve problems
- An “internet” was a network of networks; now there’s just “the Internet”

A “cloud”

# Clouds

A “cloud” is a big collection of computers, storage and networking capability and software owned by somebody else and rented out to whoever needs it

- Amazon AWS
- Microsoft Azure
- Google Cloud
- others




From *https://www.geeksforgeeks.org/architecture-of-cloud-computing/*

# Virtual Machine

A “pretend” hardware/software computer constructed entirely out of software on a bigger computer

- Software emulates (pretends to be) the actual hardware, so to your program there’s no difference

Popular because computing is so cheap that most computers today are way more powerful than would be needed.

“Container” - related but different

- Figure out what each application really needs; package that together
- Less than a full VM
- Think of shipping containers on ships & trains

# Where do the software and hardware fit in the previous picture?

The hardware is what you can see and touch

- Display screen/monitor
- Keyboard
- Mouse
- Touch pad
- Fingerprint reader/faceID reader

There is ***system software*** that tells every piece of hardware what to do

There is ***application software*** that performs “useful” operations:

-

# System software vs. Application software

***System software*** is the software that controls the hardware device

- The ***Operating System*** is the overall control program
  - Windows, MacOS, Apple iOS, Android, Linux, …
- ***Device drivers*** control the interactions of the control unit with the keyboard, mouse, screen, USB drive, hard drive or SSD, etc.

***Application software*** does “useful things”

- “Productivity apps” like Google Docs/MS Word
- Grades homework & tests
- Runs video games
- Manages money/does taxes
- Shows videos/live events
- …

# A quick digression on operating systems

1960s/1970s: project at MIT to build MULTICS - a multi-user, high performance, secure operating system

1970s: AT&T Bell Laboratories pulls out of MULTICS; Thompson, Ritchie and Kernighan write UNIX

1970s/1980s: Bell Labs shares UNIX code; UC-Berkeley writes its own “improved” version & distributes it - lots of lawsuits follow about who owns what code

1990s: Linus Torvalds, a student in Finland, writes a version of Unix he calls “Linux” and distributes it free - “open source software”

- Decades of advancements follow, but Linux still embodies many of the original UNIX principles

# Today:

Windows: Microsoft-proprietary, but has incorporated some principles from other operating systems

Apple MacOS and iOS: derived from the version of UNIX that was written at UC-Berkeley

Android: derived from Linux; purchased and controlled by Google

Linux: still open source and free; version control by Linus Torvalds and a small team. Many versions:

- RedHat
- Kali
- Debian
- Ubuntu
- ….

# Data Hierarchy

***Bit*** - one binary digit - a “1” or a “0”

***Character*** - a sequence of bits (often, 8 or 16) that together represent one character in the language you care about - e.g, an “A” or a “9”

***Fields*** - a collection of characters that together represent something meaningful - e.g., a “student last name” or a “student ID” or a “test grade”

***Records*** - a collection of fields related to the same entity - e.g., everything UMBC knows about one student that’s relevant to this class

***Files*** - a collection of records treated together - e.g., the student records of all of you in this class

***Database*** - one or more files that contain all data relevant to some project/effort

***Big Data*** - huge collections of data related in some way - e.g, “all Tweets sent out on Monday,” “all What’sApp messages sent from Germany”

# Programming

# Language Hierarchy

- Machine languages
  - what actually runs on the hardware
  - A string of 1’s and 0’s
  - Unique to every hardware manufacturer
  - You’re never going to have to deal with this!
- Assembly languages
  - Quasi-human-readable instruction sets
  - One instruction per line of code
  - Has to be ***assembled*** (translated to machine code) prior to executing
- High-level languages
  - What we deal with in this class
  - Somewhat human-readable
  - Examples:
    - C/C++ (this class)
    - Python
    - Java
    - JavaScript

# Where does programming fit?

Everywhere

- What kind of programs do you want to write?
  - Operating systems? E.g., contribute to Linux
  - Networking
  - Games
  - Financial applications?
- The language you use and the operating system you write for are tied to the kind of programming you want to do
  - C/C++ - operating systems, compilers, applications that need high performance,...
  - Python - big data, analytics, simple programs
  - JavaScript - Web services

# The C Language

# C history

Written in the 1970s along with UNIX

- The UNIX operating system is written in C
- The C compiler is written in C

Began being taught in Computer Science departments when those departments adopted UNIX computers

- Solid base of highly-competent C programmers
- It's a good, efficient language ***(it has its critics)***
- Began being widely used in industry

1980s/1990s: C++ added object-oriented concepts to C

- Bjarne Stroustrup, also Bell Labs

# Why learn C/C++ programming?

It’s a good, solid language

It’s widely used in the world

It’s efficient - runs much faster than e.g. Python

Tools, documentation, help are widely available

It continues to advance