Unit 2

Introduction to Computer Architecture and Functionality

This week's consideration focused on the evolution of computing devices and the modern-day complexity. Complexity that arises due to advances made in smaller circuitry, increased storage capacities and greater processing capabilities.

Collaborative Discussion

I continued to engage fellow students on the topic of discussion started in Unit 1.

General Thoughts on this Unit

Going into unit 2 which dealt with computer hardware, I felt very familiar with the topic on hand. This is because I have been exposed to computing hardware for my entire career and as such, topics such as inputs, outputs, controllers were second-nature to me. However, I did enjoy engaging with the students in the seminar as they presented views on different computer architectures such as laptop, desktop, Internet-of-Things.

Reflections on Terminology

The notion of **abstraction** was introduced as a way to describe a way to manage evolving complexity. Abstractions are useless without well-defined **interfaces.** These interfaces define the art-of-the-possible for the concrete implementation of such abstractions. An interface is applicable even in modern-day items given that every biological kingdom on Earth will interact with *something* (whether to kill, mate or co-exist). And it is the interactions that are limited by the *interface* of those elements.

I found it interesting that in the context of computer hardware, the *interface* is classed as the **Instruction Set Architecture (ISA)** which serves as the boundary between hardware devices and the software that operates it.

Reflections on Virtualisation

Architecture of a computer system, I considered the pros and cons of this type of architecture, outlined below:

- Pros.
    - *Software.* System code is written once and executes on multiple hardware configurations. This results in reduced development costs
    - *Security.* Hardware configurations can be protected to a greater extent because of potential for arbitrary depth of "layering". Also if one "node" along the virtualized set of devices is compromised, the virtual instance can then be torn down and a new instance replaced; otherwise known as **nested-virtualisation**.
    - *Maintenance Costs.* Organisations do not generally need to maintain massive hardware infrastructure and servers, but can reduce the overall number of hardware devices required, down to a handful.
    - *Availability.* Availability is almost 100% ("almost" because nothing in life is ever 100%) because virtual machine managers can instantiate new instances of virtual hardware when required. However, this does not contribute to *scalability* because it is not always possible to scale-out in an on-demand manner--something which Kubernetes or Docker Containers perform for software applications.
- Cons.
    - *Data Risk.* An organisation's data can be put at risk because, typically, virtual machines are hosted by third party providers, especially when considering use of virtual storage.
    - *Performance.* Since software is no longer executed on the underlying hardware, and instead all interactions go through an interface (hypervisor), each traversal must be translated from source to target and vice versa. This translation negatively impacts performance.

Just as computer hardware has the **ISA**, virtualisation requires the use of a **hypervisor** which is a piece of software that enables multiple operating systems (virtual machines) to share the same physical server resources. A Hypervisor is responsible for managing each virtual machine, and broadly speaking they come in two classifications: **Type 1** and **Type 2** (IBM, 2019)

1. Type 1 Hypervisor. Executes directly on the underlying server's hardware and they replace the host operating system.
2. Type 2 Hypervisor. Executes as a software application within an operating system and is generally used in personal computers.

Investigating this topic further, there are several virtualisation types: **network virtualisation, server virtualisation** and **desktop virtualisation.**

- *Server Virtualisation.* Creation of multiple virtual server instances on a single server.
- *Network Virtualisation.* Separates network resources from computing hardware and makes the resources available in a software-based administrative collection.
- *Desktop Virtualisation.* Creates a virtual instance of a workstation complete with its associated operating system.

References

IBM, (2019) Hypervisors. Available from https://www.ibm.com/cloud/learn/hypervisors. [Accessed on ??? 2021]