CHAPTER1: Introduction

Juan Patricio Carrizales Torres

July 16, 2022

A computer's software can be divided into kernel and user modes. Traditionally, most of the operating system resides in the former while in the latter, the user interface program (i.e. GUI) and other application programs are to be found. However, there are cases where this line that divides user and operating system software gets blurred out and many important permitions and functions are adscribed to both.
Furthermore, there are two main perspectives for the description of the main job done by the operating system, namely, *top-down* and *bottom-up*. In the top-down view, the operating system creates some type of clean and elegant *abstractions* of the hardware ressources in the architecture, which can be a hassle to work with in a low-level sense, present them to the programs in user mode and execute these abstractions.
On the other hand, for the *bottom-up* view, the operating system is mainly in charge of the management and allocation of ressources. Ressource managment includes a multiplexing allocation of ressources through time and/or space. In the case of a time multiplexed ressource, programs and users take turns using it. The OS decides the order of the turns and how much they last. A space multiplexed ressource is partitioned for the programs and users. The memory and disks are examples of space multiplexed ressources.

1 Hardware

The otganization of th hardware of personal computers is of great importance for the OS. A basic paradigm is a single bus connecting each I/O device, CPU and Memory.

1.1 CPU

It has the job of fetching pieces of program instructions form the memory, decoding them and executing, until all jobs have been completed. It's worth noting that each CPU has an specific set of instructions and so programs specifically built to run for that CPU should be handled.Also, since it may take way more time to access the memory form the CPU, it contains a serie of registers that carry variables and inputs.
Some of the registers are the following:

1. **Program counter** holds the memory adress of the next instruction to be loaded form the memory.

2. **Stack Pointer** holds the memory adress of the top of the current memory stack.

3. **PSW (Program Status Word)** holds the conditions bits.

Some CPU's contain nodes in fetch, decode and execute nodes in pipelines so that an instruction can be feteched, while others are decoded and executed. On the other hand, a superscalar CPU contains multiple fetch/decode nodes connected to a buffer that leads them to execution nodes when available. This management process of instructions is said to become really complicated for the time multiplexing of CPU, since the outputs must be printed in the solicited order. Most CPUs are divided have a kernel and user mode. In the former, normally, the OSs runs in kernel and so has acces to all instructions and commands, while the user mode has a limited access which normally hinders the access to I/O and memory protection.