Operating System

There is no consensus on the exact meaning of the term "**Operating System**" and all the definitions are deliberately vague. Here is one (Silberschatz et al.):

> *A program that acts as an intermediary between a user of a computer and the computer hardware.*

(In jest, one might say that this is the definition of a keyboard driver.)

Another, much more adequate, definition (Tanenbaum):

> *It is hard to pin down what an operating system is other than saying it is the software that runs in kernel mode—and even that is not always true.*

Operating System

is a **collection** of system programs[a] that exist permanently and have as sole purpose providing services to other programs.

A system program is one that executes privileged instructions (i.e. at least occasionally executes in kernel mode).

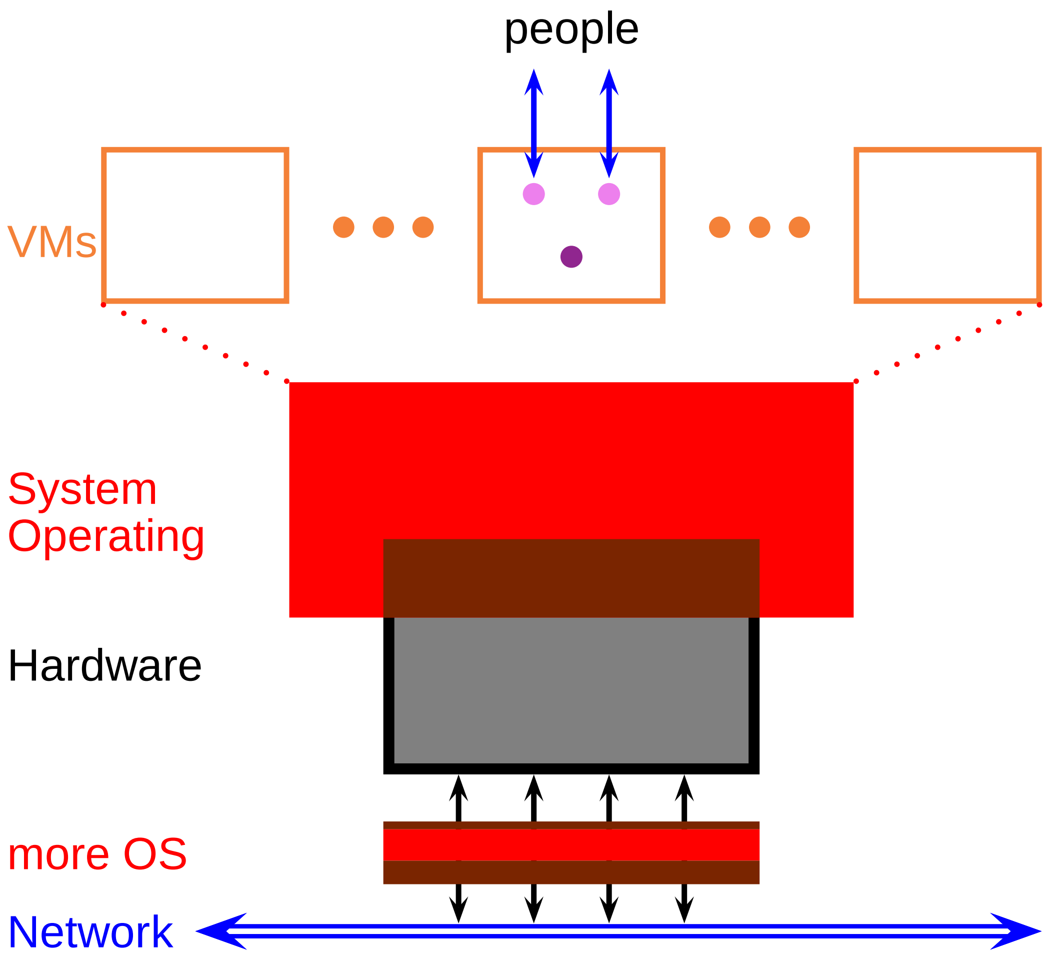
The key properties of an OS program are:

- It lives “forever” that is from booting a system to system shutdown.
- It executes privileged instructions, i.e. it performs things that an ordinary “user” program cannot do.
- Its sole role is to provide services to clients (user programs and other system programs).

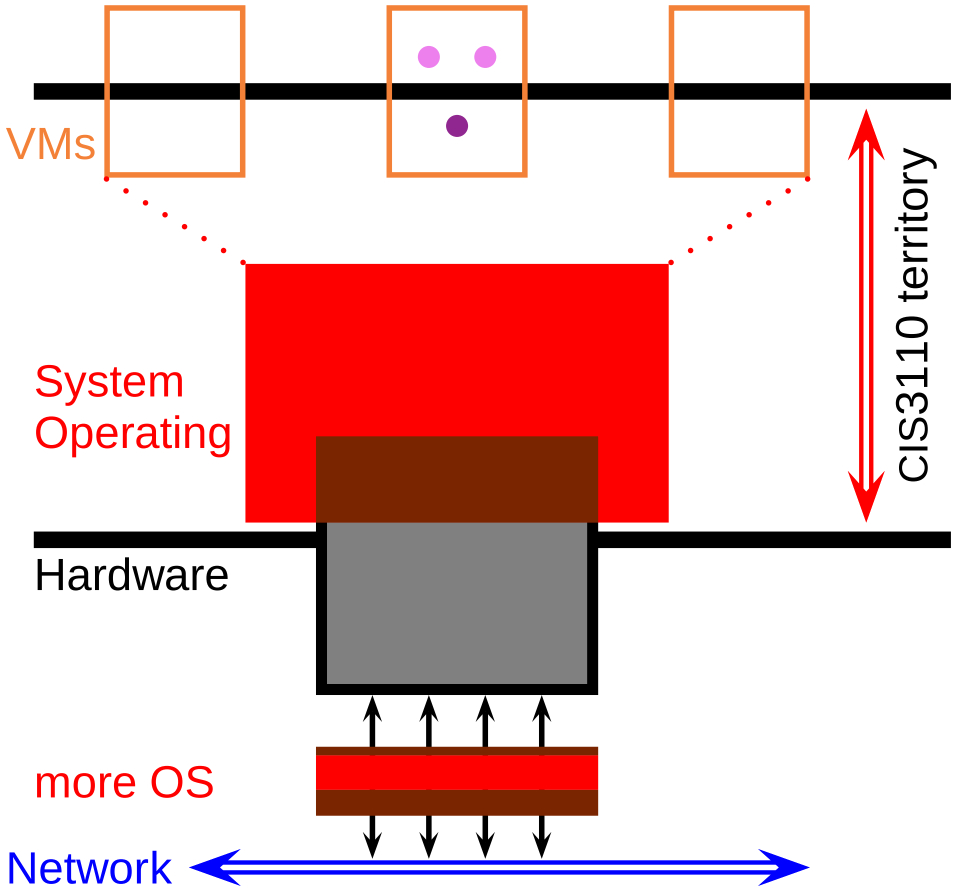
[a] The term **program** is used incorrectly and will, in due time, be replaced by the correct terms “process” or “handler.”

The context

The OS is a layer of software extending the hardware to better serve the users' (●) and the management's agents (●) called **processes**.



CIS3110


VMs
System
Operating
Hardware
more OS
Network
CIS3110 territory

The main components

The computer milieu is made of entities of two kinds:

Hardware devices. In this course, CPU and device controllers (memory, disk, network, etc.).

Software entities which can be subdivided in many ways:

- Kernel *vs.* system *vs.* application software.
- Processes *vs.* handlers/drivers *vs.* data.

Note the absence of **users** from the above; computers do not deal with people but with their **agents** which are application processes (often "shells" or GUI interfaces).

Virtual machine

A **virtual machine** is a result of the extension of the capabilities of a real machine (hardware) created by an operating system. A virtual machine is part hardware and part software and eludes the standard HW/SW division.

The OS uses the hardware protection mechanisms and interrupts to present to the outside world a view of the hardware that differs from reality (hence **virtual**; see Wikipedia for various definitions). One may use the term **emulator** to describe the role of the OS in creating an imaginary machine (VM) on top of a real one (RM).

Hardware aspects

The OS exists to make some hardware entities usable to applications; the expectation is that the OS will make the hardware “easy to use” and that use will be “orderly” i.e., **sharing** of hardware will be safe.

On the other hand, many hardware features were introduced solely for the purpose of serving the OS itself and supposedly have no real value to applications (e.g. virtual memory, kernel mode and privileged instructions, i/o buffering, etc.). However, some of these were put to a use different than the use originally intended. Some examples:

The memory protection mechanism is commonly used in maintaining the runtime stack of application processes.

Interactive debuggers use memory protection violations and signals to track the changes of values of memory locations.

Signals themselves are regularly used by application processes to communicate with other processes.