**2do teorico- u4: The abstraccion procces**

-the process: it is a running program

-how can the OS provide the illusion of a nearly-endless supply of said CPUs?: by virtualizing the CPU

By running one process, then stopping it and running another, and so forth, the OS can

promote the illusion that many virtual CPUs exist when in fact there is only one physical CPU (or a few). This basic technique, known as time sharing of the CPU, allows users to run as many concurrent processes as they would like; the potential cost is performance

-mechanisms: are low-level methods or protocols that implement a needed piece of

functionality

-space sharing: where a resource is divided (in space) among those who wish to use it

-context switch: gives the OS the ability to stop running one program and start running another on a given CPU

-Policies: algorithms for making some kind of decision within the OS.

-scheduling policy

-machine state: what a program can read or update when it is running.

-program counter (PC) (sometimes called the instruction pointer or IP) tells us which instruction of the program will execute next are used to manage the stack for function parameters, local variables, and return addresses.

-stack pointer and associated frame pointer

-modularity

4.2 Process API

These APIs, in some form, are available on any modern operating system:

• Create: An operating system must include some method to cre ate new processes. When you type a command into the shell, or double-click on an application icon, the OS is invoked to create a new process to run the program you have indicated.

• Destroy: As there is an interface for process creation, systems also provide an interface to destroy processes forcefully. Of course, many processes will run and just exit by themselves when complete; when they don’t, however, the user may wish to kill them, and thus an in-

terface to halt a runaway process is quite useful.

• Wait: Sometimes it is useful to wait for a process to stop running; thus some kind of waiting interface is often provided.

• Miscellaneous Control: Other than killing or waiting for a process, there are sometimes other controls that are possible. For example, most operating systems provide some kind of method to suspend a process (stop it from running for a while) and then resume it (continue it running).

• Status: There are usually interfaces to get some status information about a process as well, such as how long it has run for, or what state it is in.

4.3Process Creation: A Little More Detail

how programs are transformed into processes

-file descriptors: these descriptors let programs easily read input from the terminal

and print output to the screen.

4.4 Process States

a process can be in one of three states:

• Running: In the running state, a process is running on a processor. This means it is executing instructions.

• Ready: In the ready state, a process is ready to run but for some reason the OS has chosen not to run it at this given moment.

• Blocked: In the blocked state, a process has performed some kind of operation that makes it not ready to run until some other event takes place. A common example: when a process initiates an I/O request to a disk, it becomes blocked and thus some other process can use the processor.

-Being moved from ready to running means the process has been scheduled; being moved from running to ready means the process has been descheduled.

4.5 Data Structures

-The register context will hold, for stopped process, the contents of its registers

A SIDE : K EY P ROCESS T ERMS

• The process is the major OS abstraction of a running program. At any point in time, the process can be described by its state: the con tents of memory in its address space, the contents of CPU registers (including the program counter and stack pointer, among others),

and information about I/O (such as open files which can be read or written).

• The process API consists of calls programs can make related to processes. Typically, this includes creation, destruction, and other useful calls.

• Processes exist in one of many different process states, including running, ready to run, and blocked. Different events (e.g., getting scheduled or descheduled, or waiting for an I/O to complete) transition a process from one of these states to the other.

• A process list contains information about all processes in the system. Each entry is found in what is sometimes called a process control block (PCB), which is really just a structure that contains information about a specific process.