# Procesos

Un proceso es una entidad activa en un SO que cambia con el tiempo el cual se define para abstraer la ejecución. Nace desde el momento en que se lo dispara y muere cuando termina. Va llevando un PC (program counter). Se lo considera como dinámico a diferencia de un programa que se lo considera estático ya que no cambia.

Los SO modernos permiten múltiples procesos al mismo tiempo, pero solo uno puede ejecutarse a la vez en un solo procesador.

Los procesos están compuestos por: sección de código, sección de datos, pilas o stacks (para pasar parámetros, guardar datos temporales, etc). También usa la CPU.

Las pilas o stacks se crean automáticamente y están compuestas por stack frames. Hay una pila para cada modo (kernel y usuario).

**Atributos de un proceso**: identificador (para distinguirlo del resto), estado, prioridad, contador de programa, punteros a memoria, datos de contexto, información de estado de E/S, información de auditoría (tiempo de procesador y reloj utilizados).

Todos los atributos se almacenan en la **PCB (Process Control Block)**, que es un gran registro. Existe una por proceso y es lo primero que se crea cuando se crea un proceso y lo último que se borra cuando termina.

Espacio de direcciones: Conjunto de direcciones de memoria que ocupa el proceso

***No se incluye el PCB ni tablas asociadas***

Un proceso en modo usuario solo puede acceder a su espacio de direcciones mientras que en modo kernel puede acceder a estructuras internas como el PCB o a espacios de direcciones de otros procesos.

**Contexto de un proceso**: Incluye toda la información que el SO necesita para administrar el proceso, y la CPU para ejecutarlo correctamente.

Son parte del contexto, los registros de cpu, inclusive el contador de programa, prioridad del proceso, si tiene E/S pendientes, etc.

***Context switch***: Se produce cuando la CPU cambia de un proceso a otro. Se debe resguardar el contexto del proceso saliente, que pasa a espera y retornará después a la CPU. Se debe cargar el contexto del nuevo proceso y comenzar desde la instrucción siguiente a la última ejecutada en dicho contexto. Es tiempo no productivo de CPU. El tiempo que consume depende del soporte de HW

Estados de un proceso:

**Nuevo:** se inicializa y se carga en memoria

**Listo:** Está en memoria, está listo para ejecutarse. Compite por obtener tiempo de CPU.

**Ejecutando:** Ya tiene CPU, puede pasar a saliente si termina, a listo si se le termina el quantum o es interrumpido por otro o a bloqueado si realiza E/S.

**Saliente:** Se empiezan a eliminar las estructuras que se cargaron en memoria. Al final, se borra la PCB y el proceso deja de existir.

**Bloqueado:** acá esperan mientras realizan E/S, y cuando termina vuelven a listo.

Cada estado posee una o varias colas de planificación, que enlazan PCBs de los procesos.

Los cambios entre estados son realizados por los módulos de planificación:

**Short Term Scheduler (STS):** selecciona entre los procesos listos cuál es el que se ejecuta. Tiene asociado el Dispatcher que realiza el cambio de contexto para que se ejecute el proceso.

**Medium Term Scheduler (MTS):** reduce el grado de multiprogramación (cantidad de procesos que tienen memoria con capacidad de ejecutarse), saca temporalmente de memoria los procesos. Tiene asociado el Swapper que realiza el swap out y swap in (sacar y llevar a memoria).

**Long Term Scheduler (LTS):** admite los procesos al estado de listo, determina qué procesos se cargan en memoria. Tiene asociado al Loader que se encarga de cargar el espacio de direcciones del proceso.

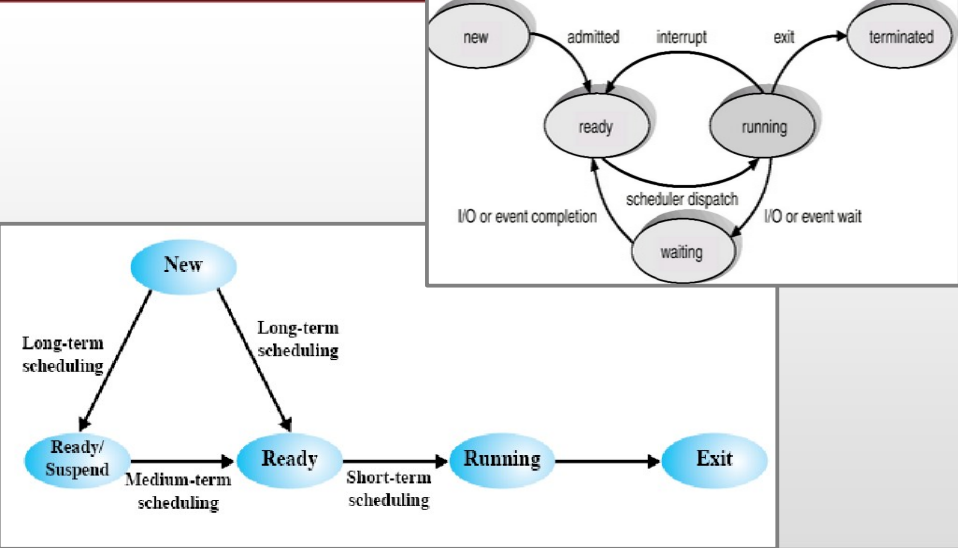


Diagrama de transiciones UNIX

