**ADMINISTRACION Y PLANIFICACION DE PROCESOS**

Programas

Son estaticos, no tienen program couter, existe desde que se edita hasta que se borra y se almacena en memoria secundaria.

Procesos

Son dinamicos, tienen program counter, su ciclo de vid ava desde que se ejecuta hasta que se termina, se almacena en RAM.

Los procesos son programas en ejecucion.

Un proceso, tarea o job son lo msimo

Se clasifican segun su historial de ejecucion en:

CPU BOUND: Son aquellos que va a usar unicamente la CPU.

I/O BOUND: Son aquellos que van a combinar CPU con E/S, suelen tener ráfagas cortas.

PCB (Process Counter Block)

Se crea una por proceso, contiene el context de cada proceso (PC, context, prioridad, etc).

Es lo primero que se crea cuando se realiza un fork y lo ultimo que se saca cuando se termina de ejecutar.

**Estados de un proceso:**

**NEW:** Ocurre cuando un proceso padre crea un proceso hijo. Si existe al capacidad de ser admitido en memoria, el proceeso pasa a estado de READY.

**READY:**  
 En esta instancia el proceso queda en RAM, peleando por la CPU, esperando que un *planificador* lo seleccione y lo pase al estado de running.

**RUNNING:**  
 Si el proceso termina su ejecusion pasa a estado EXIT, si require de alguna operacion I/O pasa a BLOCKED y si se agota el tiempo de ejecucion vuelve a estado READY.

**BLOCKED:**  
 Este estado puede estar en RAM o en la SWAP. En este estado se espera la operacion I/O para Volver al estado ready o pude pasar a estado SUSPENDIDO.

**SUSPENDED:**  
 En este estado el proceso se saca de la SWAP/RAM y Debera se reactivado desde otro pocesos para Volver al estado READY.

**EXIT:**El proceso termino su ejecución.

Cada estado posee su propia cola donde va almacenado sus procesos.

**PLANIFICADORES: (SCHEDULERS)**

Es la clave de la multiprogramacion.

Son los encargados de decidir que procesos se cargan a memoria RAM dependiendo lso recusros disponibles, se encargan de setear los nileves de prioridad y orden de ejecucucion.

Se basan primoridialmente en menor **tiempo de respuesta**, mayor **rendimiento** y en el **uso eficiente del CPU.**

Hay 3 tipos de planificadores:

**LONG TERM SCHEDULER:**  
 Se encarga de elegir que procesos cargar en RAM, controla el grado de multiprogramacion.  
 Se encarga del estado NEW.

**MEDIUM TERM SCHEDULER:**  
 realiza el swapping entre el disco y memoria cuando el SO lo require, libera RAM para permitir nuevos procesos disminuyendo el grado de multiprogramación.   
Se encarga de los estados SUSPENDED y BLOCKED.

**SHORT TERM SCHEDULER:**  
 Determina que proceso disponible (ready) sera el proximo a ser ejecutado.  
 Se encarga del estado READY.

El nombre de los planificadores indica la frecuencia con las que estos se ejecutan, siendo el short term el ejecutawdo mas frecuentemente.

Tiempos de los procesos:

**Tiempo de retorno:**  
 Tiempo que transcurre entre que el proceso entra al Sistema hasta que se termina de ejecutar.

**Tiempo de Espera:**  
 Tiempo que el proceso se queda en el Sistema esperando, es decir sin ejecutarse (TR - TCPU).

**Tiempo promedio:**  
 Tiempo promedio entre los anteriores, se usa para definir que algoritmo conviene para cada lote de procesos.

Tipos de planificación de los algortimos:

**Apropiativo (Preemptive):** El proceso en ejecución puede ser interrumpido por otro de mayor prioridad, pasando este a la cola de READY. Requiere cambios de contexto pero mejora el servicio y no permite que un proceso se adueñe del CPU.

**No-Apropiativo (Nonpreemptive):** Una vez comenzada la ejecución de un proceso este continua hasta terminar, salvo que sea un evento de I/O el cual debera pasar al estado BLOCKED.

Los procesos se catalogan con JOB (nombre/PID), LLEGADA(orden), CPU (TE), PRIORIDAD y E/S (recurso (DISPOSITIVO), n° instrucción, duracion(TE)).

**Algoritmos de planificacion:**

**Algoritmo FIFO (First In/Come First Out/Served):** Se selecciona el proceso disponible mas viejo, es decir el primero que haya entrado a ready. Algoritmo **No-Apropiativo**.

No favorece a ningún tipo de proceso pero los procesos CPU Bound terminan al comenzar su primer ráfaga y los I/O Bound deberán Volver a ready cuando requieran la operación I/O, haciéndolos volver a esperar.

**Algoritmo SJF (Shortest Job First):** Se selecciona al proceso más corto primero, es decir con menos CPU o ráfagas mas cortas.  
Algoritmo **No-Apropiativo**.

Posee la desventaja que los procesos largos pueden sufrir de inanición (Starving) ya que son constantemente adelantados por procesos cortos.

**Algoritmo RR (Round Robin):** Política basada en un reloj, se basa en una unidad de medida de tiempo denominada **QUANTUM (Q).**

Este Quantum determina cuanto tiempos de CPU puede usar cada proceso, los procesos son seleccionados por orden FIFO.

Si el Q es chico va a haber mucho *context switch*, generando *overhead*. Por el otro lado si el Q es mucho más grande de los necesario se terminaría con un resultado similar al FIFO ya que no se acabaría el Q antes que un proceso termine.

Cuando un proceso es expulsado de la CPU por timeout este vuelve a la cola de READY.

El contador de Quantum se puede asignar de manera global o local en la PCB.

En el RR el contador se puede inicializar de 2 maneras, de tiempo variable o tiempo fijo.

El **timer variable** (Timer Variable) el contador se inicializa en Q cada vez que entra un proceso. Es lo mas usado. El Quantum de cada proceso se suele guardar en su PCB.

Por el otro lado en el algoritmo de **timer fijo** el contador se incializa en Q, pero en caso que un proceso se salga del estadso RUNNING antes que Q=0 el siguiente proceso que entre a RUNNING se va a ejecutar únicamente el Quantum que restante que haya dejado el proceso anterior y volverá a estado READY. En este caso se considera como que el cuantum es común entre los procesos es decir global.

**Algoritmo con uso de prioridades:**Este algoritmo selecciona al proceso según su orden de prioridad establecido por el short scheduler.

Cuanto mas chivo en numero de prioridad, mayor es la prioridad.

Cada nivel de prioridad suele tener su propia cola de procesos disponibles en el estado READY.

Los procesos de baja prioridad suelen sufrir de inanición. Se soluciona permitiendo a los procesos cambiar su nivel de prioridad a medida que TIEMPO DE ESPERA aumenta (**AGING**) o lo contrario (**PENALTY**) que es penalizar a los procesos que están siendo seleccionados muy frecuentemente, bajando su nivel de prioridad.

Estos algortimos pueden ser APROPIATIVOS, o no.

**Algoritmo SRTF (Shortest Remaining Time First):**Es una version **APROPIATIVA** de SJF.

Selecciona el proceso que posee la siguiente ráfaga mas corta, con la menor siguiente ráfaga de CPU.

Pero con la diferencia que al ser apropiativo, si un proceso sale de un blockeo/suspensión con un CPU mas bajo que el proceso actual, se realizara un cambio de contexto.

Este tipo de algoritmo beneficia a los procesos I/O Bound, que se caracterizan por tener ráfagas cortas entre las operaciones I/O.

**Algoritmo VRR (Virtual Round Robin):**

Este algoritmo funciona igual que el RR, con la diferencia que los procesos al volver de una I/O se añaden a una cola auxiliar. Cuando se termina de ejecutar un proceso se verifica primero esa cola, en caso de que haya un proceso se prioriza el mismo. A dicho proceso se le asigna tantas unidades de procesador, como unidades no haya ejecutado en su ultima ráfaga, es decir Quantum original – tiempo de su ultima ráfaga.

**Algoritmos de planificación CPU I/O**

Cada dispositivo I/O posee su propia cola de procesos en espera, con un scheduler para cada una.

Las operaciones I/O se consideran independientes a la CPU ya que de ellas se encarga la DMA, PCI, etc. Es decir se pueden realizar operaciones de E/S al mismo tiempo que la CPU realiza sus operaciones.

**Criterios de desempate:**Cuando un proceso empata con otro en la cola de READY se debe desempatar.

Esto puede ser por **orden de llegada** a la cola de READY y en caso de haber llegado al mismo tiempo, se desempata por el **menor PID** (JOB).

**Esquema de colas MULTINIVEL:**Es lo usado hoy en día, es una combinación de varios de los algoritmos vistos.

La cola de READY se divide en **varias colas**, cada proceso se coloca en una cola según una clasificación que determina el SO, **similar a prioridades**.

Cada cola posee su algoritmo de **planificación horizontal** (RR, SJF, etc,) y a su vez las colas son ordenadas por un **planificador horizontal.**

También tiene el concepto de **retroalimentación** Cada proceso puede ir variando de una cola a otra (aging / penalty).

Las colas de prioridad son Procesos de Sistema (Scheduler, Dispatcher, Bugger, etc.), Procesos de interactivo , procesos intercativos de edición, procesos Batch y con la menor prioridad los proceso de los usuarios.

Si un proceso no termina en dicha ráfaga, se procede a mandarlo a la cola de ready del nivel inmediato inferior.

Con multiples CPU se complica la planificación, pero si uno falla el resto toma el control.

**Planificacion con multiples procesadores:**

**Planificacion temporal:**Determina que proceso y durante cuanto tiempo se va a ejecutar y con que procesador.

**Planificacion espacial:** En cual de los procesadores vamos a ejecutar, depende de la huella y la afinidad

**Huella:** Estado que el proceso va dejando en la cache del procesador.

Por lo que si un proceso se ejecuto en un procesador convendría que vuelva a lmismo ya que este posee la huella previa.

**Afinidad:** Un proceso prefiere un procesador especifico por sus características, por ejemplo procesos graficos tienen afinidad con los procesadores de la GPU.

**Asignacion de procesos a un procesador**

**Estatica:** Cuando hay una afinidad de un proceso a una cpu. Esta afinidad no puede cambiar.

**Dinamica:** La carga se va repartiendo, se distribuye según el uso actúal de cada procesador.

**Politica de multiprocesadores**

**Tiempo Compartido:** Es cuando se considera una cola global, o una local a cada procesador. En donde se van compartiendo los procesos entre todos los procesadores, generando procesamiento concurrente.

**Espacio Compartido:** Se divide los procesos en Grupos (Hilos) y en particiones.

**Clasificaciones:**

**Procesadores Homogeneos:** Todas las CPU son iguales, no hay diferencia. No existe GPU.

**Procesadores Heterogeneos:** Hay procesadores específicos para tareas, cada procesador posee su propia cola, reloj y algoritmo de planificación. Consigue una mejor perfomance.

**Otras clasificaciones**

**Procesadores débilmente acoplados:** Cada CPU tiene su propia memoria principal y canales, es decir no comparte nada con el resto de procesadores

**Procesadores fuertemente acomplados:** Las CPU comparten memoria y canales, se encuentran bajo control de un Sistema Operativo

**Procesadores especializados:** Donde hay procesadores de uso general y otros de uso especifico.