T1.5. Procesos vs Hilos

Procesos

Un proceso es cualquier programa en ejecución. este necesita ciertos recursos para realizar satisfactoriamente su tarea:

- Tiempo de CPU.
- Memoria.
- Archivos.
- Dispositivos de E/S.

Las obligaciones del SO como gestor de procesos son:

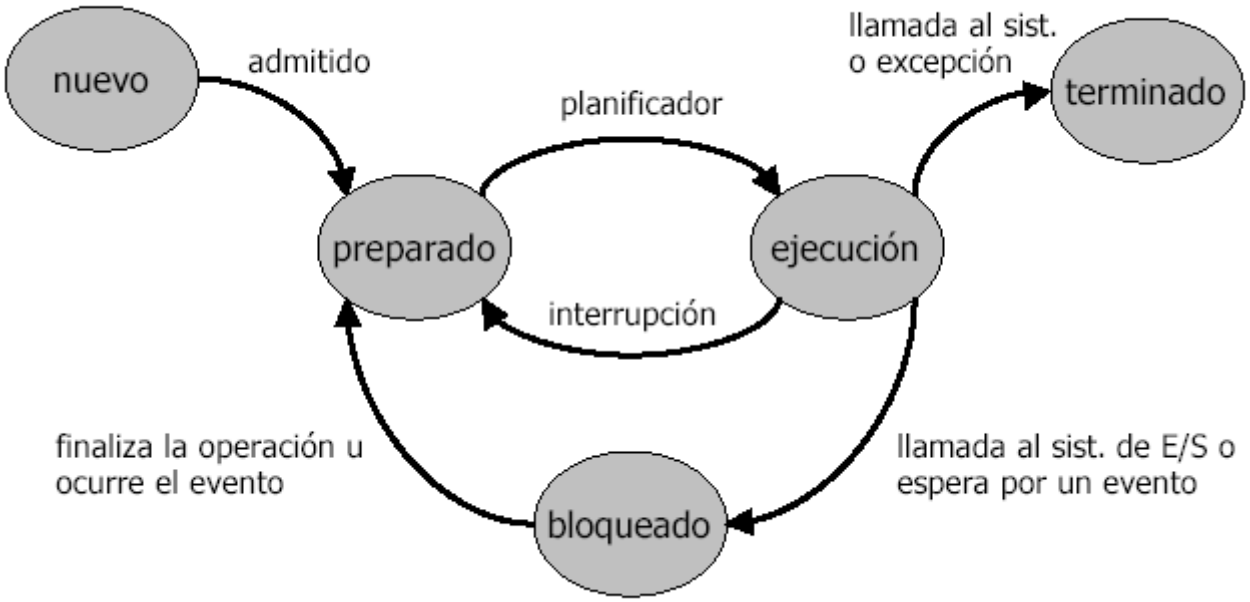
- Creación y eliminación de procesos.
- Planificación de procesos (procurando la ejecución de múltiples procesos maximizando la utilización del procesador).
- Establecimiento de mecanismos para la sincronización y comunicación de procesos.
- Manejo de bloqueos mutuos.

Estados de un proceso
A medida que un proceso se ejecuta cambia de estado. Cada proceso puede estar en uno de los estados:

- Nuevo (new): el proceso se está creando.
- En ejecución (running): el proceso está en la CPU ejecutando instrucciones.
- Bloqueado (waiting, en espera): proceso esperando a que ocurra un suceso (ej. terminación de E/S o recepción de una señal).
- Preparado (ready, listo): esperando que se le asigne a un procesador.
- Terminado (terminated): finalizó su ejecución, por tanto no ejecuta más instrucciones y el SO le retirará los recursos que consume.

Nota: Sólo un proceso puede estar ejecutándose en cualquier procesador en un instante dado, pero muchos procesos pueden estar listos y esperando.

Diagrama de estados de un proceso



Para que un programa se ejecute, el SO debe crear un proceso para él. En un sistema con multiprogramación el procesador ejecuta código de distintos programas que pertenecen a distintos procesos.

Aunque dos procesos estén asociados al mismo programa, se consideran dos secuencias de ejecución separadas, cada una de las cuales se considera un proceso.

Llamamos traza de un proceso al listado de la secuencia de instrucciones que se ejecutan para el mismo.

Creación de Procesos

Los procesos se crean mediante una llamada al sistema de “crear proceso”, durante el curso de su ejecución. El proceso creador se denomina proceso padre, y el nuevo proceso, hijo.

Cuando un proceso crea un proceso nuevo, hay dos posibilidades en términos de ejecución:

- Padre e hijo se ejecutan concurrentemente.
- Padre espera por la finalización del hijo.

En UNIX existen dos funciones básicas para crear procesos: Fork y Exec.

Función fork(): Cuando se la llama crea un proceso hijo que es una copia casi exacta del proceso padre (duplicado del padre). Ambos procesos continúan ejecutándose desde el punto en el que se hizo la llamada a fork().

En UNIX los procesos se identifican mediante un “identificador de proceso” (PID) que es un entero único. Ambos procesos continúan su ejecución con la instrucción que sigue al fork() con una diferencia:

- El código que el hijo recibe del fork es cero.
- El que recibe del padre es el propio pid.

Funciones exec: Tras crear un nuevo proceso, después de llamar a fork, Linux llama a una función de la familia exec. Éstas funciones reemplazan el programa ejecutándose en el proceso por otro

programa. Cuando un programa llama a una función exec, su ejecución cesa de inmediato y comienza a ejecutar el nuevo programa desde el principio, suponiendo que no ocurriera ningún error durante la llamada.

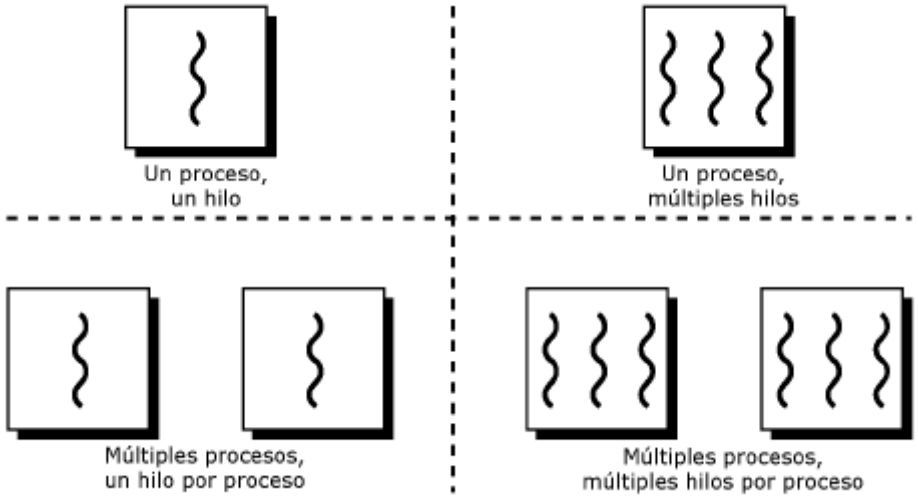
Generalmente uno de los dos procesos (padre o hijo) utiliza la llamada al sistema exec ve después del fork para reemplazar su espacio de memoria con un programa nuevo.

Nota: Si el padre no tiene nada que hacer mientras el hijo se ejecuta, puede emitir una llamada wait (esperar) para sacarse a sí mismo de la cola de procesos listos hasta que el hijo termine.

Hilos

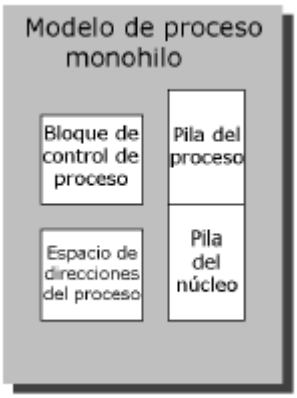
Los hilos son un concepto relativamente nuevo de los SO. En este contexto, un proceso recibe el nombre de proceso pesado, mientras que un hilo recibe el nombre de proceso ligero. El término hilo se refiere sintáctica y semánticamente a hilos de ejecución.

El término multihilo hace referencia a la capacidad de un SO para mantener varios hilos de ejecución dentro del mismo proceso.

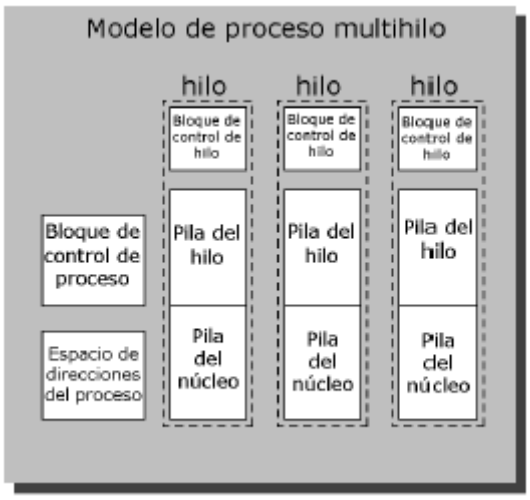


En un SO con procesos monohilo (un solo hilo de ejecución por proceso), en el que no existe el concepto de hilo, la representación de un proceso incluye su PCB, un espacio de direcciones del proceso, una pila de proceso y una pila núcleo.

(***PCB*** *o* ***Process Control Block*** *es un registro especial donde el agrupa toda la información que necesita conocer respecto a un proceso particular. Cada vez que se crea un proceso el sistema operativo crea el BCP correspondiente para que sirva como descripción en tiempo de ejecución durante toda la vida del proceso.*)



En un SO con procesos multihilo, sólo hay un PCB y un espacio de direcciones asociados al proceso, sin embargo, ahora hay pilas separadas para cada hilo y bloques de control para cada hilo.



Estructura de los Hilos
Un hilo (proceso ligero) es una unidad básica de utilización de la CPU, y consiste en un contador de programa, un juego de registros y un espacio de pila.

Los hilos dentro de una misma aplicación comparten:

- La sección de código.
- La sección de datos.
- Los recursos del SO (archivos abiertos y señales).

Un proceso tradicional o pesado es igual a una tarea con un solo hilo.

Los hilos permiten la ejecución concurrente de varias secuencias de instrucciones asociadas a diferentes funciones dentro de un mismo proceso, compartiendo un mismo espacio de direcciones y las mismas estructuras de datos del núcleo.

Estados de un Hilo
Los principales estados de un hilo son: ejecución, preparado y bloqueado; y hay cuatro operaciones básicas relacionadas con el cambio de estado de los hilos:

- Creación: En general, cuando se crea un nuevo proceso se crea también un hilo para ese proceso. Posteriormente, ese hilo puede crear nuevos hilos dándoles un puntero de instrucción y algunos argumentos. Ese hilo se colocará en la cola de preparados.
- Bloqueo: Cuando un hilo debe esperar por un suceso, se le bloquea guardando sus registros. Así el procesador pasará a ejecutar otro hilo preparado.
- Desbloqueo: Cuando se produce el suceso por el que un hilo se bloqueó pasa a la cola de listos.
- Terminación: Cuando un hilo finaliza, se liberan su contexto y sus pilas.

Un punto importante es la posibilidad de que el bloqueo de un hilo lleve al bloqueo de todo el proceso. Es decir, que el bloqueo de un hilo lleve al bloqueo de todos los hilos que lo componen, aún cuando el proceso está preparado.

Recursos compartidos y no compartidos
Los hilos permiten la ejecución concurrente de varias secuencias de instrucciones asociadas a diferentes funciones dentro de un mismo proceso, compartiendo un mismo espacio de direcciones y las mismas estructuras de datos del núcleo.

Recursos compartidos entre los hilos:

- Código (instrucciones).
- Variables globales.
- Ficheros y dispositivos abiertos.

Recursos no compartidos entre los hilos:

- Contador del programa (cada hilo puede ejecutar una sección distinta de código).
- Registros de CPU.
- Pila para las variables locales de los procedimientos a las que se invoca después de crear un hilo.
- Estado: distintos hilos pueden estar en ejecución, listos o bloqueados esperando un evento.

PROCESOS v/s HILOS

Semejanzas: Los hilos operan, en muchos sentidos, igual que los procesos.

- Pueden estar en uno o varios estados: listo, bloqueado, en ejecución o terminado.
- También comparten la CPU.
- Sólo hay un hilo activo (en ejecución) en un instante dado.
- Un hilo dentro de un proceso se ejecuta secuencialmente.
- Cada hilo tiene su propia pila y contador de programa.
- Pueden crear sus propios hilos hijos.

Diferencias: Los hilos, a diferencia de los procesos, no son independientes entre sí.

- Como todos los hilos pueden acceder a todas las direcciones de la tarea, un hilo puede leer la pila de cualquier otro hilo o escribir sobre ella. Aunque pueda parecer lo contrario la protección no es necesaria ya que el diseño de una tarea con múltiples hilos tiene que ser un usuario único.

Ventajas: de los hilos sobre los procesos.

- Se tarda mucho menos tiempo en crear un nuevo hilo en un proceso existente que en crear un nuevo proceso.
- Se tarda mucho menos tiempo en terminar un hilo que un proceso.
- Se tarda mucho menos tiempo en conmutar entre hilos de un mismo proceso que entre procesos.
- Los hilos hacen más rápida la comunicación entre procesos, ya que al compartir memoria y recursos, se pueden comunicar entre sí sin invocar el núcleo del SO.