Sistemas operativos II

**ASPECTOS DEL DISEÑO DE PAQUETES DE HILOS. (190 tarea)**

Un conjunto de primitivas relacionas con los hilos disponibles para los usuarios se llama un paquete de hilos.

El primer aspecto que analizaremos es el manejo de los hilos. Aquí se tienen dos alternativas, los hilos dinámicos y los estáticos. En un diseño estático, se elige el número de hilos al escribir el programa o durante su compilación.

Un método más general consiste en permitir la creación y destrucción de los hilos durante la ejecución. La llamada para la creación de hilos determina el programa principal del hilo y un tamaño de pila, así como otros posibles parámetros; ejemplo una prioridad de planificación. La llamada regresa por lo general un identificador de hilo; para utilizarlo en las llamadas posteriores relacionadas con el hilo. En este modelo, un proceso se inicia (manera implícita) con un hilo, pero puede crear el número necesario de ellos y estos pueden expirar al terminar.

Los hilos pueden concluir de dos maneras. Un hilo puede hacer su salida por su cuenta, al terminar su trabajo, o puede ser eliminado desde el exterior. Los hilos son como los procesos.

Puesto que los hilos comparten una memoria común, pueden y de hecho la utilizan, para guardar datos que comparten los distintos hilos, como los buffers en un sistema de productores y consumidores. El acceso a los datos compartidos se programa por lo general mediante regiones críticas, para evitar que varios hilos intenten tener acceso a los mismos datos al mismo tiempo. La implantación de las regiones críticas es más fácil si se utilizan los semáforos, monitores u otras construcciones similares. Una técnica en común en los paquetes de hilos es el MUTEX. Que es un cierto topo de semáforo moderado. Un MUTEX solo tiene dos estados cerrado y no cerrado. Se definen dos operaciones de MUTEX. La primera LOCK, intenta cerrar MUTEX. Si el MUTEX no está cerrado, LOCK tiene el éxito y el MUTLEX se cierra en una acción atómica. Si dos hilos intentan cerrar al mismo MUTEX en el mismo instante, evento que solo es posible en un multiprocesador, en el cual existen varios hilos en ejecución en distintos CPU, uno de ellos gana y el otro pierde. Si un hilo intenta cerrar un MUTLEX ya cerrado, se bloquea.

La operación UNLOCK elimina la cerradura de un MUTLEX. Si uno o más hilos esperan a un MUTLEX, se libera exactamente uno de ellos. El resto continua su espera.

Otras operaciones que se tienen en ciertos casos es TRYLOCK que intenta cerrar un MUTLEX, si el MUTLEX no está cerrado TRYLOCK regresa un código de estado que indica el éxito. MUTLEX cerrado TRYLOCK no bloque el hilo sino regresa un código de estado que indica la falla.

Otra característica de sincronización que a veces está disponible en los paquetes de hilos es la variable de condición, similar a la variable de condición que se utiliza para la sincronización en el caso de los monitores. Por lo general, se asocia una variable de condición a un MUTEX cuando este se crea. La diferencia entre el MUTLEX y la variable de condición es que el primero se utiliza para una cerradura a corto plazo, principalmente para proteger la entrada a las regiones críticas. Las variables de condición se utilizan para una espera a largo plazo hasta que un recurso esté disponible.

Una solución es el uso de las variables de condición para adquirir el recurso. En este caso, la espera de la variable de condición se define de modo que ejecute la espera y elimine la cerradura en forma atómica. Más adelante, cuando el hilo que conserva el recurso lo libera, llama a WAKEUP que se define de forma que despierte a un hilo o bien a todos los hilos que esperan a la variable de condición especifica. El uso de WHILE en vez de IF, evita que el hilo se despierte pero que alguien más se apropie del recurso antes de que se ejecute el hilo.

La necesidad de poder despertar a todos los hilos en vez de uno, se demuestra en el problema de los lectores y los escritores. Al terminar un escritor, puede adoptar por despertar a todos los escritores pendientes o a todos los lectores pendientes. Si elige los lectores, debe despertarlos a todos y no a uno. La flexibilidad necesaria se obtiene mediante las primitivas de hilos para despertar con exactitud un hilo o despertarlos a todos.

El código de un hilo consta por lo general de varios procedimientos, al igual que un proceso. Puede tener varias variables locales, variables globales y parámetros del procedimiento.

Las variables locales y los parámetros no provocan problema alguno, pero las variables globales de un hilo que no son globales en todo el programa pueden provocar ciertas dificultades.

Sin embargo, el acceso a las variables globales particulares es un tanto truculento, puesto que la mayoría de los lenguajes de programación tiene forma de expresar las variables locales y las globales, pero no tienen formas intermedias. Es posible asignar un bloque de memoria a las variables globales y transferirlo a cada procedimiento del hilo, como un parámetro adicional. Aunque esta no es una solución elegante, funciona.

En otra alternativa, se pueden introducir nuevos procedimientos de biblioteca para crear, dar valores y leer estas variables globales a lo largo de todo un hilo. La primera llamada se parecería a:

CREATE\_GLOBAL(“BUFPTR”);

Asignan un espacio de almacenamiento a un apuntador llamado BUFPTR en una pila o un área especial de almacenamiento, reservado para el hilo que hizo la llamada. Sin importar la posición del espacio de almacenamiento, el hilo que llamo es el único que tiene acceso a la variable global.

Si otro hilo crea una variable global con el mismo nombre, obtiene un espacio de almacenamiento distinto, que no entra en conflicto con el ya existente.

Se necesitan dos llamadas para tener acceso a las variables globales; una para escribirlas y otras para leerlas. Para escritura, funciona algo como:

SET\_GLOBAL(“BUFPTR”, &BUF);

Guarda el valor de un apuntador en el espacio de almacenamiento creado en la llamada a CREATE\_GLOBAL. Para leer una variable global, la llamada sería algo así como:

BUFPTR=READ\_GLOBAL(“BUFPTR”);

Esta llamada regresa a dirección almacenada en el variable global, de modo que se pueda tener el acceso al valor del dato.

Nuestro último aspecto del diseño relativo a los hilos es la planificación. Los hilos se pueden planificar mediante distintos algoritmos, entre los que se encuentran la prioridad, round robin y otros. Los paquetes de hilos proporcionan a menudo ciertas llamadas para que el usuario pueda especificar el algoritmo de planificación y establecer las prioridades, en su caso.