**Hilos**

**¿Qué es un hilo?**

Hoy día un procesador es capaz de llevar a cabo múltiples tareas a la vez, simultáneamente como en un procesador multinucleo, o empleando otros métodos como la concurrencia\*, que permitían realizar varias tareas aparentando simultaneidad cuando el procesador sólo contaba con un único núcleo [WebParalelismo]; los sistemas operativos están diseñados para coordinar todas estas tareas, asignándoles recursos de hardware y administrando la forma en que se ejecutan, de ahí surge la noción de proceso, la cual consta de dos elementos primordiales: un código a ejecutar y un conjunto de datos referentes a este, además de otros elementos para caracterizarlo, como un estado, una prioridad, múltiples registros, información sobre el tiempo de procesamiento, etc. donde esta información es almacenada en lo que se conoce como un bloque de control de proceso [SOLibroCap3], estos detalles hacen que un proceso se pueda considerar algo macro y ostentoso que utiliza recursos que quizá no sean tan relevantes a la hora de realizar actividades más específicas [UTLProyecto], por lo cual un proceso es subdivido en lo que se conoce como **hilos**, o también denominados **procesos ligeros** (*lightweight process*), ya que estos se encargan de ejecutar porciones del código sobre el mismo segmento de memoria, permitiendo no sólo programar códigos más ordenado y estructurados, sino también aprovechar más eficientemente el uso de los recursos de máquina al utilizar paralelismo\* para ejecutar múltiples hilos simultáneamente, o simplemente mediante la concurrencia, ya que como utilizan el mismo segmento de memoria y comparte los archivos abiertos, la comunicación entre hilos resulta ser muy eficiente [TutorialsPoint].

Concurrencia: A grandes rasgos, varias tareas se pueden ejecutar de forma concurrente cuando se va ejecutando instrucciones de cada una por ciertos periodos, haciendo que todas avancen.

Paralelismo: Varias tareas se pueden ejecutar en paralelo cuando literalmente se ejecutan simultáneamente en diferentes núcleos dentro del procesador.

**Tipos de Hilos**

Existen dos tipos de hilos, los de nivel de usuario (ULT *user level thread*) y los de nivel de kernel (KLT *kernel level thread*).

Los ULT son aquellos que se emplean al utilizar una librería de hilos para desarrollar una aplicación multihilo, de esta forma la librearía se encarga de todo el proceso del manejo de estos, desde su creación y destrucción hasta la forma en que se alternará entre los hilos y su comunicación, permitiendo al programador implementar el sistema multihilo para mejorar la gestión de su programa.

Como la librearía es la que realiza la gestión de los hilos, su implementación es independiente del SO, lo cual es una gran ventaja respecto a la portabilidad de la aplicación, pero es una gran desventaja respecto al uso de los recursos de máquina, pues el SO no puede detectar el uso de múltiples hilos, lo cual deriva en que trate a la aplicación como un proceso de un único hilo, es decir, lo clasifica como un todo en uno de los estados de un proceso\*, lo cual implica que si uno de los hilos realiza una llamada bloqueante al sistema\*, todo el proceso será bloqueado y se le indicará al procesador que continúe con otro diferente en lugar de sólo bloquear el hilo que realizó la llamada y procesar otro relativo a la misma aplicación que se encuentre en el estado listo [OSLibroCap4].

Estados: Un proceso puede encontrarse en uno de cinco estados, ejecutando, listo bloqueado, nuevo y saliente. En particular, que un proceso se encuentre en estado bloqueado significa que no puede continuar porque está esperando algo, como una operación I/O o a que otro hilo o proceso termine su ejecución [SOLibroCap3].

Llamada bloqueante: Consiste en una llamada al sistema operativo que implica que hace que este coloque al hilo o proceso en el estado bloqueado.

Una ventaja importante es que al utilizar ULTs se tiene un grado de control mucho mayor respecto a la forma en que los hilos van a implementarse como se menciona en [ULTProyecto], en particular se pueden determinar cómo será el planificador que gestione la ejecución de los hilos, algo que viene predeterminado por el SO dentro de los KLTs. Otra ventaja relevante al uso de ULTs es que el cambio de hilo no requiere privilegios de modo núcleo, lo cual ahorra el costo de tener que cambiar del modo usuario al modo núcleo [OSLibroCap4].

Por otra parte, dos desventajas importantes del ULT son que, como se mencionó antes, gracias a que el SO no detecta el proceso como multihilo, el programa no puede hacer uso de varios núcleos de procesamiento en caso de que se cuente con más de uno, eliminando así la posibilidad del paralelismo, mientras que la otra desventaja es la que se había mencionado antes, respecto al bloqueo de todo el proceso al momento en que un hilo realiza una llamada bloqueante al sistema [OSLibroCap4].

En contraste, los KLTs son administrados netamente por el kernel, por lo cual este reconoce el programa como un solo proceso pero de múltiples hilos, lo cual permite que el planificador por defecto del kernel sea capaz de asignar diferentes hilos de este mismo proceso a diferentes núcleos de procesamiento, permitiendo el paralelismo y el mejor aprovechamiento de los recursos de máquina, además de que si uno de los hilos es bloqueado, el kernel puede continuar con otro hilo del mismo proceso [OSLibroCap4]..

Sin embargo, utilizar KLTs también acarrea sus propias desventajas, como por ejemplo que para pasar de un hilo a otro se requiere entrar en el modo núcleo para tener los privilegios, lo cual lleva un costo adicional. Así, utilizar ULTs prohíbe el uso de varios núcleos de procesamiento al tiempo pero resulta menos costoso moverse entre hilos, mientas que al utilizar KLTs sucede todo lo contrario, por lo cual la elección de cual tipo de hilo utilizar depende intrínsecamente de la aplicación que se vaya a desarrollar [OSLibroCap4].

Antes de finalizar esta sección, no está demás mencionar que existen algunos SOs que cuentan con combinaciones entre los ULTs y los KLTs, como Solaris, en donde los hilos se crean como ULTs, pero estos con asociados a un número menor o igual de KLTs [OSLibroCap4].

Citar también el blog de Javier.

Los KLT dependen del sistema operativo, mientras que los ULT dependen del lenguaje, supongo. Ninguno depende del hardware, o eso creo hasta ahora.