Hilos

Juan Camilo Arteaga Ibarra

Tutor
Augusto Enrique Salazar Jiménez

Universidad de Antioquia
Facultad de Ingeniería

Informática 2

2019 - 2

El flujo de control de las tareas de un microprocesador son administradas por una unidad básica llamada hilo, este se encarga de dividir en dos o más tareas (subprocesos), que pueden ser simultáneas o seudo simultáneas, a un programa que está en ejecución. Cada subproceso se asigna a una de estas unidades lógicas denominadas como hilos, existentes debido a la programación de la unidad de control. Una vez asignadas, las divisiones de la tarea principal se alternan en ejecución de modo en que aparenta realizarse dos o más procesos al mismo tiempo, permitiendo lograr un control eficaz en el tiempo de espera de los procesos y el ahorro de recursos, una optimización del funcionamiento de un microprocesador [1].

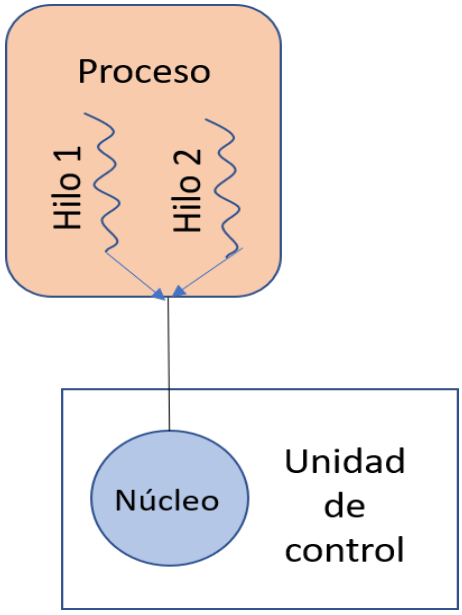

Figura 1: gráfico funcionamiento.

El origen de los hilos no es muy exacto, se tiene una noción de su aparición a mediados del año 1965, con el diseño del OS/360 de IBM mencionada en los comentarios de Dijkstra [2]. El lenguaje PL/I definido por IBM realizaba procesos pesados, tareas que dieron como resultado un hilo, pero se cree que nunca fue implementado debido a falta de protección entre de los hilos de control. Años más tarde en 1970 Max Smith hizo un prototipo en donde utilizaba múltiples “stacks” en un proceso pesado para poder aguantar compilaciones en segundo plano. Alrededor de la década de los setentas llegó Unix, que permitiendo la realización de procesos con máquinas pesadas, las cuales no podían compartir memoria. Al ver este problema se crearon los hilos, subprocesos que compartían las direcciones de memoria de un único proceso de Unix, dando entrada a los primero micro-núcleos. Se cree que gracias a estos acontecimientos se pudo continuar mejorando los “subprocesos” hasta consolidar la idea de lo que es son los hilos. Pero es importante mencionar que no se tienen datos certeros sobre la utilización de estos métodos para la realización de las tareas de un procesador [3].

Hay dos tipos de hilos [4]:

- **A nivel de usuario (ULT):** la aplicación es quien se encarga de la gestión de los hilos, por lo que para estos no es necesario que el SO ayude en el proceso. Para poder programar estos hilos es necesario hacer uso de alguna librería de hilos. Gracias a estos hilos el usuario tiene permitido realizar planificaciones específicas de ejecución.
- **A nivel de kernel o núcleo (KLT):** todo el trabajo es realizado por el núcleo, los crea y los organiza para las tareas a ejecutar. Gracias a este tipo de hilos los procesos se realizan mucho más rápido y se ahorran recursos.

Estos hilos a nivel de usuario y de kernel tienen tres relaciones [4], que son:

- **One to one:** cada hilo del usuario se asigna a uno del kernel.
- **Many to one:** varios hilos de usuario se asignan a uno del kernel. Son hilos a nivel de usuario.
- **Many to many:** Un número de hilos de usuario se asignan a una igual o menor cantidad de hilos del kernel.

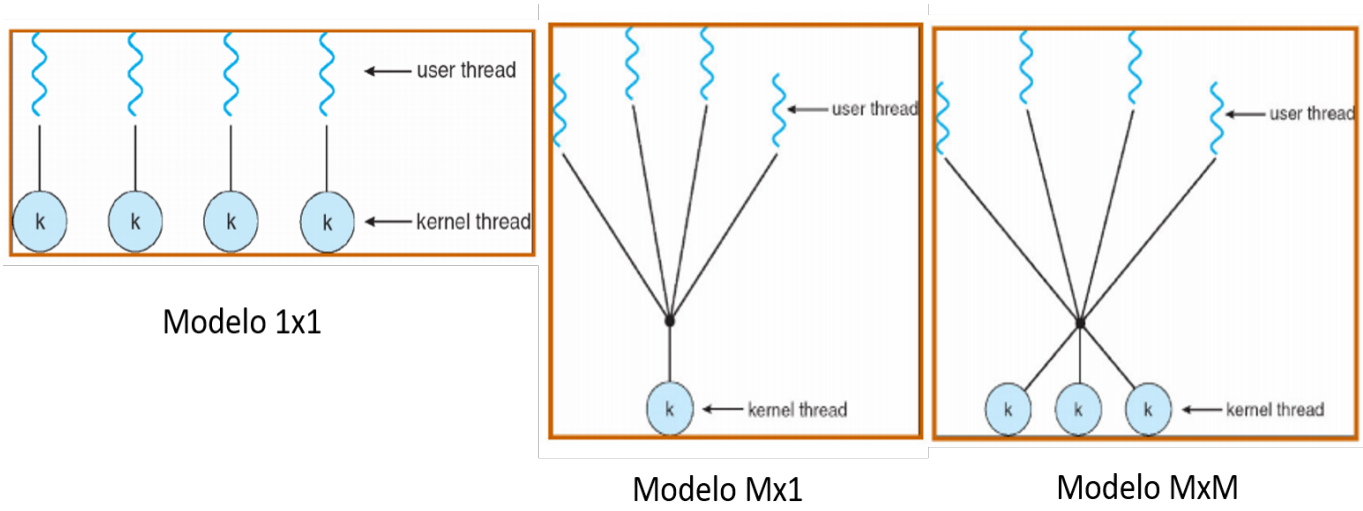

Figura 2: Pg 2, 3 [4]

Un procesador y/o un microprocesador pueden tener un solo hilo o varios hilos. Si la unidad de control solo tiene un hilo (single-threading) entonces solo procesa un comando a la vez, pero si tiene varios hilos entonces puede hacer que una tarea se divida en subprocesos que pueden ser ejecutados simultáneamente o muy a la par. El tener más de un hilo permite que haya una respuesta rápida a problemas, una ejecución más rápida, un menor consumo de recursos, un mejor uso del sistema y una forma más eficaz de comunicación interna [5].

Actualmente existen dos implementaciones de los hilos, a nivel de hardware y a nivel de software.

- En cuanto a la implementación de los hilos a nivel de hardware es la que viene por defecto en el microprocesador o el procesador, la estos poseen una cantidad de núcleos e hilos que pueden utilizar para la realización de procesos, los hilos que posee la unidad de control son reconocidos por el kernel, es decir, son hilos a nivel de kernel, lo que quiere decir que son limitados, dependen del hardware, por ejemplo si de fábrica se establece que tenga ocho hilos no puede poseer un número mayor de hilos a ocho, pero si puede dejar de utilizar un número menor al establecido de fábrica al momento de realizar la ejecución de un proceso [6].
- Por otro lado la implementación a nivel de software, hilos a nivel de usuario, se pueden tener tantos hilos como se deseen, pero hay que atender a la relación que se esté haciendo con los que están a nivel del kernel. En la relación uno a uno hay una limitación de cuántos hilos implementados en el software se pueden tener, dependería del hardware que se tenga. Otra cosa a tener en cuenta con los hilos implementados a nivel de software es el lenguaje en el que se están programando, la velocidad con la que se realicen los subprocesos de una tarea establecida dependerán de este, además, algunas funciones para el uso de estos hilos dependen de las librerías diseñadas para la creación de los mismos, por tanto algunos lenguajes pueden tener limitadores en las funciones que tienen los hilos [7].

Referencias

[1] R. Lopes, *Threading Concepts*. New Jersey Institute of Technology, Desconocido. [Online]. Available: https://web.njit.edu/~rlopes/Mod11.2.pdf

[2] E. Dijkstra, "The next fifty years," 2007. [Online]. Available: http://www.cs.utexas.edu/users/EWD/transcriptions/EWD12xx/EWD1243a.html

[3] B. O'Sullivan, "The history of threads," 2005. [Online]. Available: http://www.serpentine.com/blog/threads-faq/the-history-of-threads/

[4] Fing/CETP, *Arquitectura de Computadoras*, Desconocido. [Online]. Available: https://www.fing.edu.uy/tecnoinf/mvd/cursos/arqcomp/material/teo/arq-teo08.pdf

[5] A. Tanenbaum, *MODERN OPERATING SYSTEMS*. VRIJE UNIVERSITEIT AMSTERDAM, Desconocido. [Online]. Available: https://www.cs.vu.nl/~ast/books/mos2/sample-2.pdf

[6] J. Khoury, "What is the difference between hardware and software threads and how do they çommunicate¿" 2017. [Online]. Available: https://www.quora.com/What-is-the-difference-between-hardware-and-software-threads-and-how-do-they-communicate

[7] Múltiples autores, "software threads vs hardware threads," 2011. [Online]. Available: https://stackoverflow.com/questions/5593328/software-threads-vs-hardware-threads