

Conceptos Básicos

Un **proceso** se puede definir simplemente como un programa en ejecución. Esto incluye los valores activos del contador, registros y variables del programa. De manera conceptual cada proceso tiene su propia CPU virtual. En la realidad la CPU verdadera alterna entre los procesos. La diferencia entre un programa y un proceso es sutil pero también crítica. Por ejemplo, cuando una persona está cocinando tiene una receta y una cocina abastecida con los ingredientes necesarios para preparar su platillo. En esta analogía, la receta es el programa, la persona que cocina es la CPU y los ingredientes son los datos de entrada. El proceso es la actividad en la que la persona lee la receta, busca los ingredientes y cocina.

La idea clave es que un proceso es una actividad de cierto tipo. Tiene un programa, entrada, salida y estado. Un solo procesador puede ser compartido entre varios procesos, con cierto algoritmo de planificación, que se utiliza para determinar cuando detener el trabajo en un proceso y dar servicio a otro distinto.

Thread.

Windows introduce las hebras (thread), los principales objetos con los que trata el planificador del sistema.

La hebra es la unidad básica de la [planificación](#) en Windows. Es el componente real de un proceso que se ejecuta en un momento dado. Se ejecuta en el espacio de direcciones del proceso y utiliza los recursos asignados a dicho proceso.

Una hebra:

- Es un camino de ejecución dentro de un proceso.
- Se puede crear mediante cualquier aplicación de 32 bits de Windows o VxD que se ejecute en Windows.
- Tiene su propio almacenamiento de pila y contexto de ejecución (principalmente, los registros del procesador) privados.
- Comparte la memoria asignada al proceso padre.
- Puede ser una de las muchas hebras concurrentes creadas por un único proceso.

La propiedad de los recursos corresponde al proceso, no a las hebras. Las hebras utilizan los recursos, pero el proceso conserva la propiedad. Por ejemplo, si una aplicación solicita el uso de un puerto, es el proceso quien controla dicho puerto. Cualquiera de las hebras de ese proceso puede utilizar el puerto, pero una hebra no puede solicitar el uso del mismo.

En un programa de múltiples hebras, el programador tiene la responsabilidad de garantizar que las diferentes hebras no interfieran unas con otras. Esto puede conseguirse utilizando los recursos compartidos de forma que no se entre en conflicto con otra hebra que esté utilizando el mismo recurso.

En particular, el hecho de que las hebras compartan todo el código y los datos globales con su proceso padre significa que establecer una nueva hebra implica hacer cantidades mínimas de asignación de memoria. Cuando Windows carga una aplicación y crea las estructuras de los datos de los procesos asociados, el sistema establece el proceso como una sola hebra. Muchas aplicaciones utilizarán sólo una única hebra durante toda su ejecución, pero una aplicación puede (y muchas lo hacen) utilizar otra hebra para realizar alguna operación no prioritaria de corta duración.

En Windows, los servicios para hebra sólo están disponibles para las aplicaciones de 32 bits y los VxD. Las VM MS-DOS y otras aplicaciones de 16 bits para Windows anteriores no pueden llamar API de hebras. Una VM MS-DOS representa una sola hebra: una VM MS-DOS es un proceso que es una hebra. Cada aplicación de 16 bits utiliza una única hebra para su ejecución y con ello se conserva el modelo de [multitarea cooperativa](#) de las aplicaciones anteriores de Windows.

Cualquier aplicación de 32 bits de Windows o VxD puede crear hebras adicionales y Windows puede [planificarlas con derecho preferente](#).

Planificación en Windows

El componente del sistema operativo que gestiona la multitarea en Windows es el planificador.

El planificador trata principalmente con el tiempo y los sucesos. Un proceso de Windows consigue un lapso de tiempo que determina durante cuánto tiempo puede usar la CPU. Al final del lapso del proceso, el planificador decide si permite que un proceso diferente utilice la CPU. Los sucesos influyen en las decisiones del planificador. Para éste, una pulsación del ratón es un suceso que puede significar que debe asignar la CPU al proceso al que pertenece la ventana sobre la que se produjo la pulsación del ratón, o el planificador puede considerar que la conclusión de una transferencia de datos de red realizada de forma simultánea es un suceso que debe recibir una mayor atención que la pulsación del ratón. En ese caso, el proceso que gestiona la red conseguiría la CPU y el otro proceso tendría que esperar.

Windows utiliza sistema de [multitarea cooperativa](#) y [multitarea con derecho preferente](#) o asignación prioritaria, para mantener la compatibilidad con los programas existentes.

La planificación es el proceso por el cual se determina qué hebra debe utilizar el procesador. Este proceso se basa en una unidad de tiempo predeterminada. En la práctica, el lapso de tiempo real depende de la configuración del sistema.

Planificación de la máquina virtual. El proceso de planificación en Windows está tan relacionado con las máquinas virtuales que es apropiado examinar la planificación como parte del estudio del VMM.

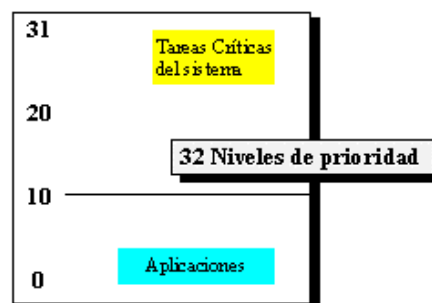
Dentro del VMM de Windows existen dos planificadores:

- **Planificador principal:** Que es el responsable de calcular las prioridades de las hebras.
- **Planificador de lapsos de tiempo:** Que se responsabiliza de la asignación de los lapsos. Decide qué porcentaje del tiempo disponible del procesador se asigna a cada diferente [hebra](#). Si una hebra no recibe tiempo de ejecución, quedará suspendida y no podrá volver a ejecutarse hasta que el planificador vuelva a evaluar la situación.

Los planificadores de Windows Cuando hay varios procesos activos en el sistema, es necesario disponer de algún método para determinar el orden en que se ejecutan las [hebras](#).

Cada [hebra](#) tiene una prioridad base. La prioridad determina cuando se ejecuta una [hebra](#) en relación con otras [hebras](#) del sistema. Se otorga el uso del procesador a la [hebra](#) que tenga la mayor prioridad. La prioridad base de las [hebras](#) de un proceso puede alterarse en dos niveles ascendentes o descendentes.

Hay 32 niveles de prioridad, que abarcan desde el nivel de prioridad más bajo, de 0 hasta el 31. La prioridad base de las [hebras](#) de un proceso se puede cambiar en un máximo de dos niveles hacia arriba o hacia abajo. Uno de los métodos para cambiar la prioridad base de un programa es que el planificador modifique la prioridad de las [hebras](#).



A continuación se explica cómo trabaja el proceso de planificación:

1. El planificador principal examina cada [hebra](#) presente en el sistema y calcula un valor de prioridad de ejecución para la [hebra](#), un entero entre 0 y 31.
2. El planificador principal suspende cualquier [hebra](#) que tenga un valor de prioridad de ejecución inferior al valor más alto. (Si dos hebras tienen el valor de prioridad de ejecución igual a 20 y las demás tienen un valor menor de 20, entonces 20 es el valor más alto hasta que se vuelva a calcular la prioridad). Una vez que suspende una [hebra](#), el planificador principal no le presta más atención en el cálculo de prioridad durante el lapso en cuestión.
3. Después, el planificador de lapsos calcula el porcentaje de lapso a asignar a cada [hebra](#), usando los valores de prioridad y conociendo el estado actual de la VM.
4. Se ejecutan las [hebras](#). Por omisión, el planificador principal volverá a evaluar las prioridades cada 20 milisegundos.

También entran en juego en este proceso tres señalizadores de control que mantiene cada VM:

VM Stat Exclusive. Indica al planificador que la VM en cuestión debe recibir el 100 por 100 del lapso siguiente; ninguno de los otros dos señalizadores restantes estará activado.

Alguno de los dos indicadores restantes: VM Stat Background y VM Stat High Pri Background debe estar activado para que el planificador pueda garantizar una VM de segundo plano alguna asignación en el lapso siguiente; de otro modo, la VM de primer plano obtendrá toda la asignación.

Planificación dentro de la máquina virtual del sistema. Todas las [hebras](#) de las aplicaciones Windows se ejecutan dentro del contexto de la VM del sistema. La VM del sistema es la única VM que admite múltiples hebras: una por cada aplicación Windows de 16 bits, y al menos una por cada aplicación Windows de 32 bits. A partir del algoritmo de planificación, la VM del sistema frecuentemente contendrá múltiples hebras no ociosas con prioridades igual de altas.

Para la gestión de esta situación, el planificador de lapsos adopta una política de planificación igualitaria para asegurar una asignación justa de tiempo de ejecución entre las hebras que tienen igual prioridad.

Una vez que una [hebra](#) dentro de la VM del sistema consume su tiempo asignado de ejecución, el planificador la coloca al final de la cola que contiene las hebras de igual prioridad.

Esta técnica asegura que todas las hebras con el nivel más alto de prioridad tienen una misma oportunidad de consumir su tiempo de procesador.

Control del planificador. Dos influencias diferentes controlan el planificador.

Una de ellas es la de sus propios algoritmos internos que intentan proporcionar un entorno de multitarea uniforme donde cada hebra recibe una compartición equitativa del tiempo del procesador.

La otra influencia en el planificador son las llamadas directas a los servicios del sistema que podrían hacer los VxD.

Internamente, el planificador utiliza tres técnicas como ayuda para lograr su objetivo de distribución equitativa del tiempo del procesador, con el fin de dar una impresión de respuesta rápida y uniforme:

Prioridad dinámica estimulada. Permite al planificador principal subir o bajar la prioridad de una hebra. Por ejemplo, una pulsación del teclado o del ratón indica que hay que estimular la prioridad de la hebra receptora.

Decaimiento de tiempo. Hace que la prioridad estimulada de una hebra vuelva, gradualmente, a su valor normal.

Herencia de prioridad. Convierte rápidamente una hebra de prioridad baja en una hebra de prioridad más alta. Normalmente se invierte la prioridad de una hebra para permitir que una hebra con una prioridad baja termine rápidamente su utilización de un recurso exclusivo por el que esperan hebras de prioridad más alta.