SISTEMAS OPERATIVOS

Ingeniería civil informática

Gonzalo Carreño

gonzalocarrenob@Gmail.com

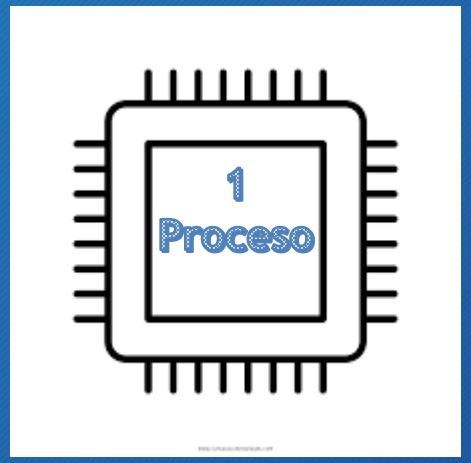
Proceso

- La tarea fundamental de cualquier sistema operativo moderno es la gestión de procesos. El sistema operativo debe reservar recursos para los procesos, permitir a los mismos compartir e intercambiar información, proteger los recursos de cada uno de ellos del resto, y permitir la sincronización entre procesos. Para conseguir alcanzar estos requisitos, el sistema operativo debe mantener una estructura determinada para cada proceso que describa el estado y la propiedad de los recursos y que permite al sistema operativo establecer el control sobre los procesos.

Proceso

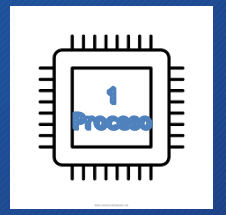
Monoprocesador

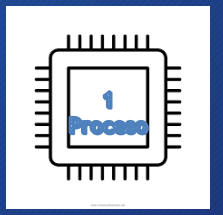
Proceso en espera



Multiprocesador

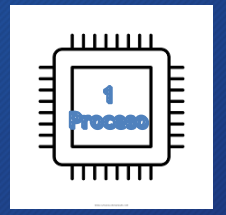
Proceso en espera

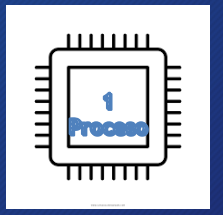




Proceso en espera

Proceso en espera





Proceso en espera

Aparece el concepto de hilos

Descripción y control de procesos

- El objetivo de los sistemas operativos tradicionales es la gestión de procesos. Cada proceso se encuentra, en un instante dado, en uno de los diferentes estados de ejecución, que incluyen:

- El sistema operativo sigue la traza de estos estados de ejecución y gestiona el movimiento de procesos entre los mismos. Con este fin el sistema operativo mantiene unas estructuras de datos complejas que describen cada proceso. El sistema operativo debe realizar las operaciones de planificación y proporcionar servicios para la compartición entre procesos y la sincronización.

Descripción y control de procesos

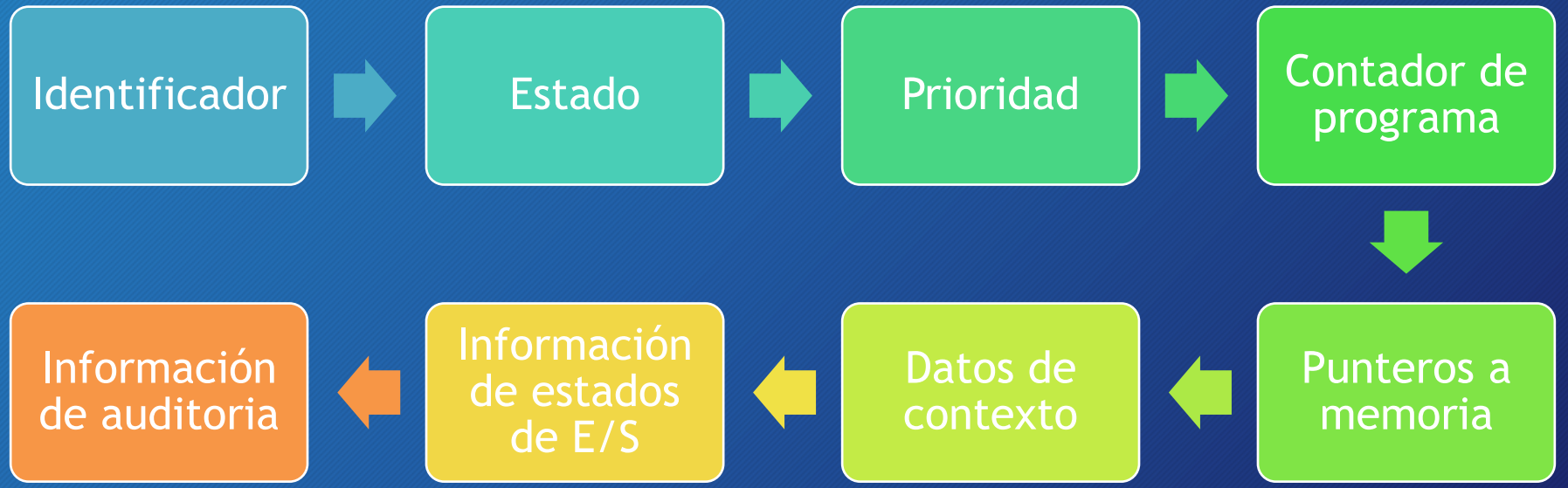
- La mayoría de los requisitos que un sistema operativo debe cumplir se pueden expresar con referencia a los procesos:
  - El sistema operativo debe intercalar la ejecución de múltiples procesos, para maximizar la utilización del procesador mientras se proporciona un tiempo de respuesta razonable.
  - El sistema operativo debe reservar recursos para los procesos conforme a una política específica (por ejemplo, ciertas funciones o aplicaciones son de mayor prioridad) mientras que al mismo tiempo evita interbloqueos.
  - Un sistema operativo puede requerir dar soporte a la comunicación entre procesos y la creación de procesos, mediante las cuales ayuda a la estructuración de las aplicaciones.

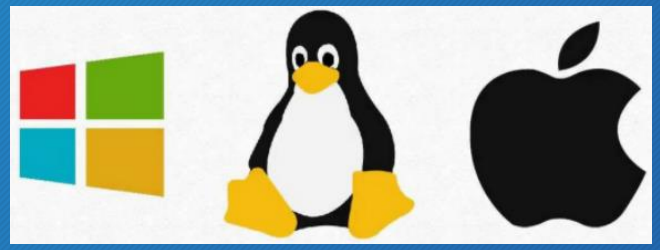
¿QUÉ ES UN PROCESO?

- Un proceso es en esencia un programa en ejecución. Cada proceso tiene asociado un espacio de direcciones, una lista de ubicaciones de memoria que va desde algún mínimo hasta cierto valor máximo, donde el proceso puede leer y escribir información.
- El espacio de direcciones contiene el programa ejecutable, los datos del programa y su pila.
- También hay asociado a cada proceso un conjunto de recursos, que comúnmente incluye registros (el contador de programa y el apuntador de pila, entre otros), una lista de archivos abiertos, alarmas pendientes, listas de procesos relacionados y toda la demás información necesaria para ejecutar el programa. En esencia, un proceso es un recipiente que guarda toda la información necesaria para ejecutar el programa.

Descripción y control de procesos

- En cualquier instante puntual del tiempo, mientras el proceso está en ejecución, este proceso se puede caracterizar por una serie de elementos, incluyendo los siguientes:





Sistema operativo crea y gestiona una lista, llamada bloque de control

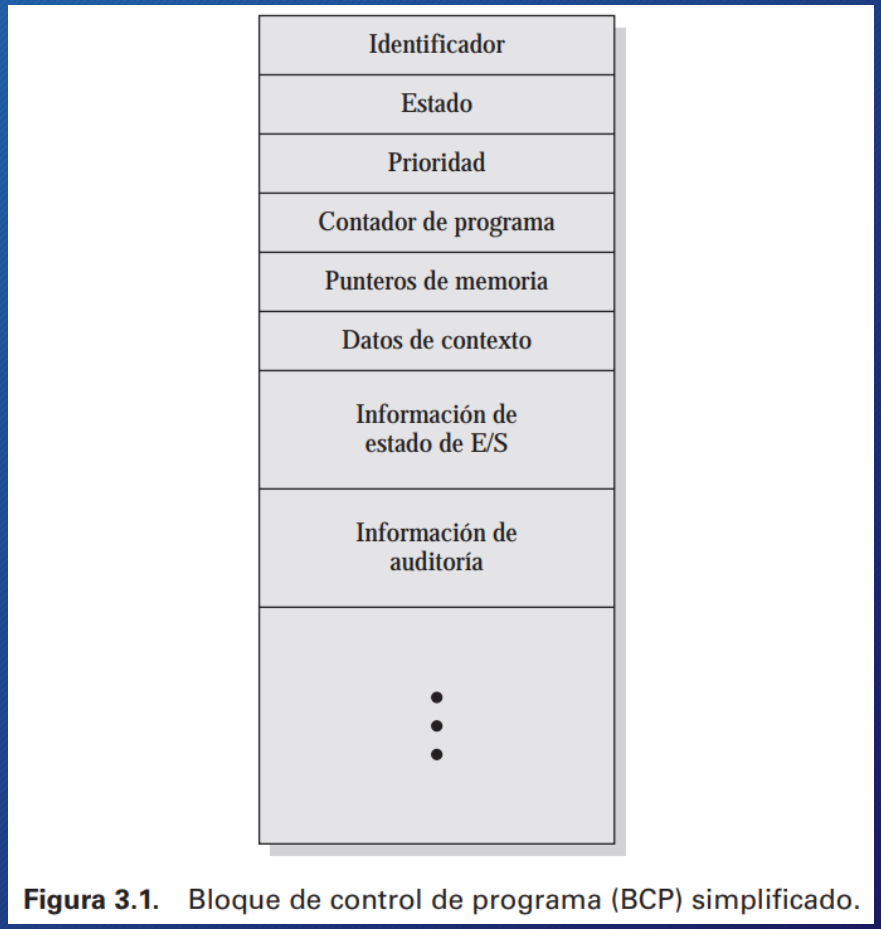
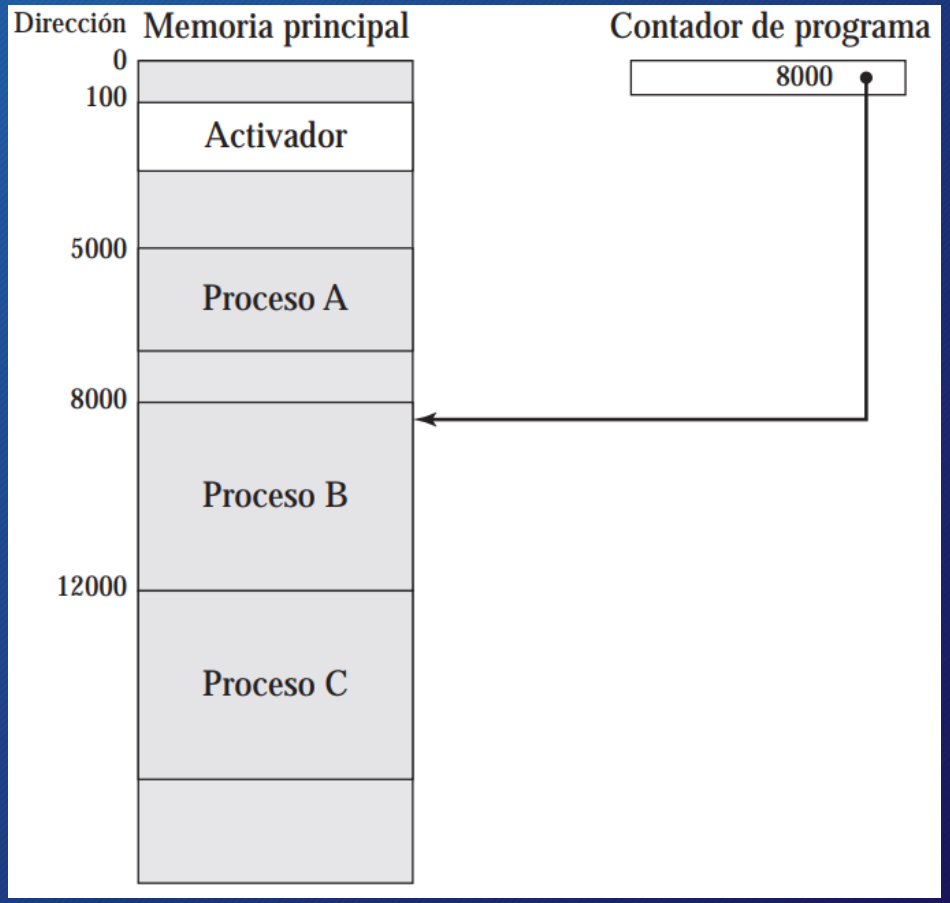


Figura 3.1. Bloque de control de programa (BCP) simplificado.

- Se puede caracterizar el comportamiento de un determinado proceso, listando la secuencia de instrucciones que se ejecutan para dicho proceso. A esta lista se la denomina traza del proceso. Se puede caracterizar el comportamiento de un procesador mostrando cómo las trazas de varios procesos se entrelazan.



- La Figura muestra las trazas de cada uno de los procesos en los primeros instantes de ejecución. Se muestran las 12 primeras instrucciones ejecutadas por los procesos A y C. El proceso B ejecuta 4 instrucciones y se asume que la cuarta instrucción invoca una operación de E/S, a la cual el proceso debe esperar.

5000	8000	12000
5001	8001	12001
5002	8002	12002
5003	8003	12003
5004		12004
5005		12005
5006		12006
5007		12007
5008		12008
5009		12009
5010		12010
5011		12011
(a) Traza del Proceso A	**(b) Traza del Proceso B**	**(c) Traza del Proceso C**

5000 = Dirección de comienzo del programa del Proceso A.
8000 = Dirección de comienzo del programa del Proceso B.
12000 = Dirección de comienzo del programa del Proceso C.

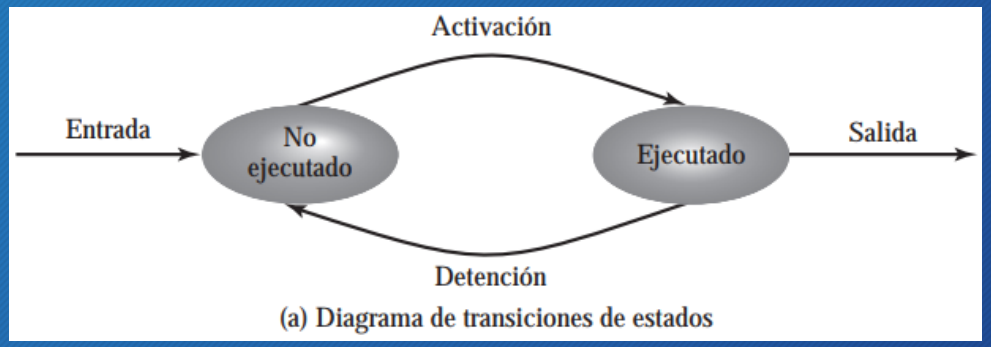
Figura 3.3. Traza de los procesos de la Figura 3.2.

1	5000	
2	5001	
3	5002	
4	5003	
5	5004	
6	5005	
		Temporización
7	100	
8	101	
9	102	
10	103	
11	104	
12	105	
13	8000	
14	8001	
15	8002	
16	8003	
		Petición de E/S
17	100	
18	101	
19	102	
20	103	
21	104	
22	105	
23	12000	
24	12001	
25	12002	
26	12003	
27	12004	
28	12005	
		Temporización
29	100	
30	101	
31	102	
32	103	
33	104	
34	105	
35	5006	
36	5007	
37	5008	
38	5009	
39	5010	
40	5011	
		Temporización
41	100	
42	101	
43	102	
44	103	
45	104	
46	105	
47	12006	
48	12007	
49	12008	
50	12009	
51	12010	
52	12011	
		Temporización

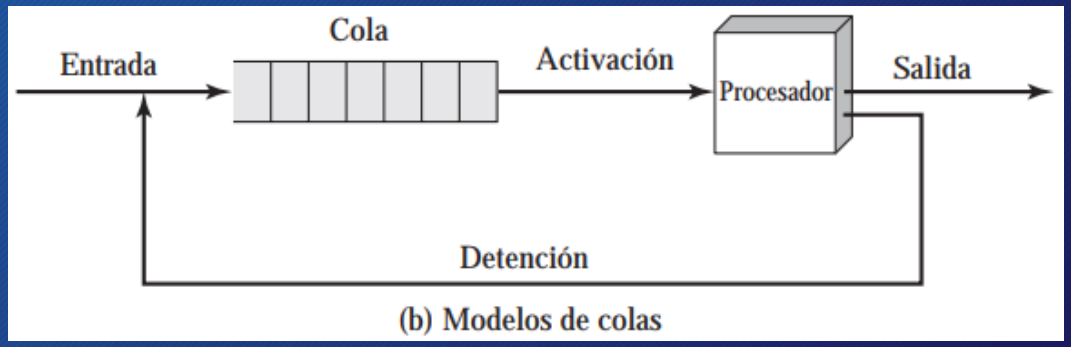
100 = Dirección de comienzo del programa activador.
Las zonas sombreadas indican la ejecución del proceso de activación;
la primera y la tercera columna cuentan ciclos de instrucciones;
la segunda y la cuarta columna las direcciones de las instrucciones que se ejecutan

Un modelo de proceso de dos estados

- La responsabilidad principal del sistema operativo es controlar la ejecución de los procesos; esto incluye determinar el patrón de entrelazado para la ejecución y asignar recursos a los procesos. El primer paso en el diseño de un sistema operativo para el control de procesos es describir el comportamiento que se desea que tengan los procesos.



(a) Diagrama de transiciones de estados



(b) Modelos de colas

Creación de procesos

- Creación de un proceso. Cuando se va a añadir un nuevo proceso a aquellos que se están gestionando en un determinado momento, el sistema operativo construye las estructuras de datos que se usan para manejar el proceso y reserva el espacio de direcciones en memoria principal para el proceso. Estas acciones constituyen la creación de un nuevo proceso.
- Cuando un proceso lanza otro, al primero se le denomina proceso padre, y al proceso creado se le denomina proceso hijo. Habitualmente, la relación entre procesos necesita comunicación y cooperación entre ellos.

Terminación de procesos

- Todo sistema debe proporcionar los mecanismos mediante los cuales un proceso indica su finalización, o que ha completado su tarea. Un trabajo por lotes debe incluir una instrucción HALT o una llamada a un servicio de sistema operativo especifica para su terminación.
- Para una aplicación interactiva, las acciones del usuario indicarán cuando el proceso ha terminado. Por ejemplo, en un sistema de tiempo compartido, el proceso de un usuario en particular puede terminar cuando el usuario sale del sistema o apaga su terminal. En un ordenador personal o una estación de trabajo, el usuario puede salir de una aplicación (por ejemplo, un procesador de texto o una hoja de cálculo). Todas estas acciones tienen como resultado final la solicitud de un servicio al sistema operativo para terminar con el proceso solicitante.

Modelo de proceso de cinco estados

- Si todos los procesos estuviesen siempre preparados para ejecutar, la gestión de colas sería efectiva. La cola es una lista de tipo FIFO y el procesador opera siguiendo una estrategia cíclica (round-robin o turno rotatorio) sobre todos los procesos disponibles (cada proceso de la cola tiene cierta cantidad de tiempo, por turnos, para ejecutar y regresar de nuevo a la cola, a menos que se bloquee). Sin embargo, esta implementación es inadecuada.

Modelo de proceso de cinco estados

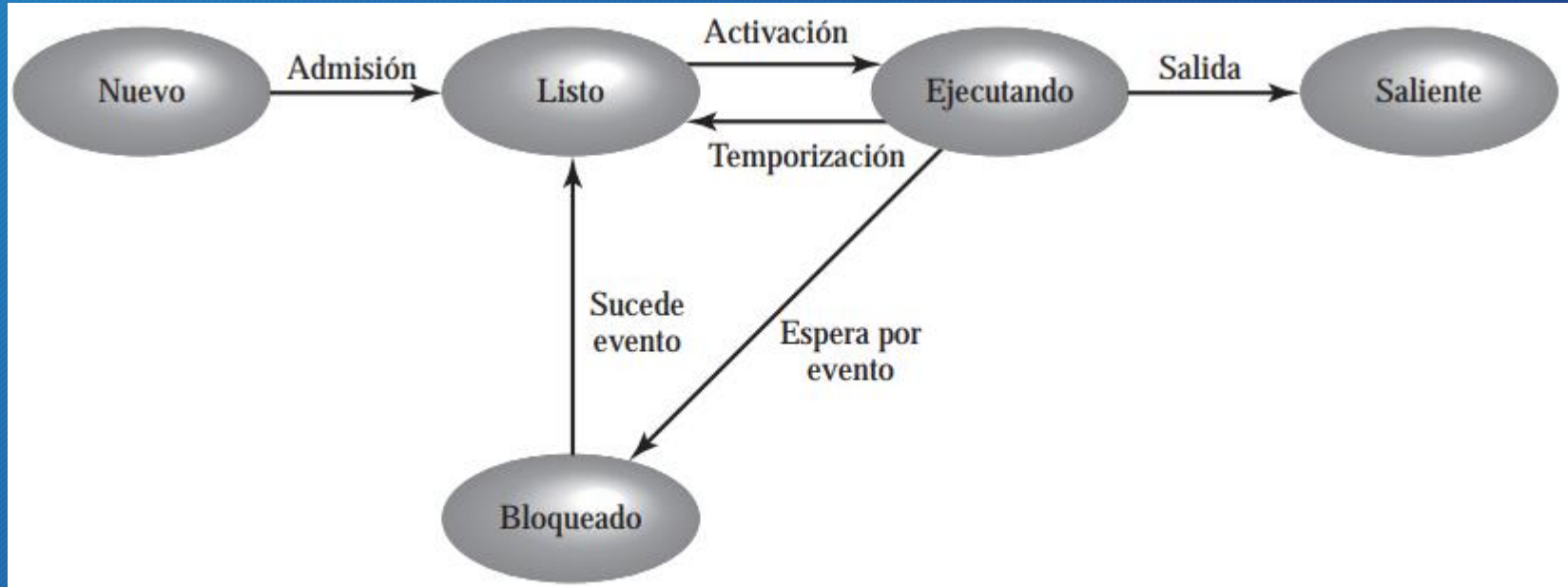
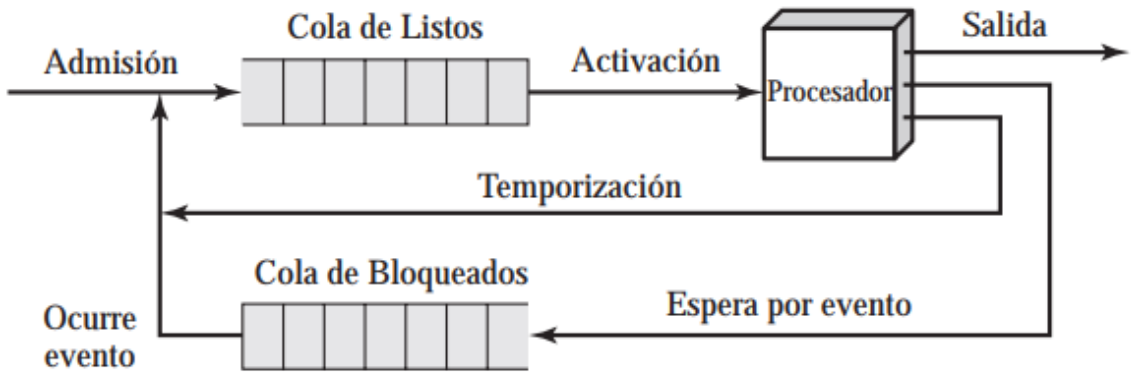


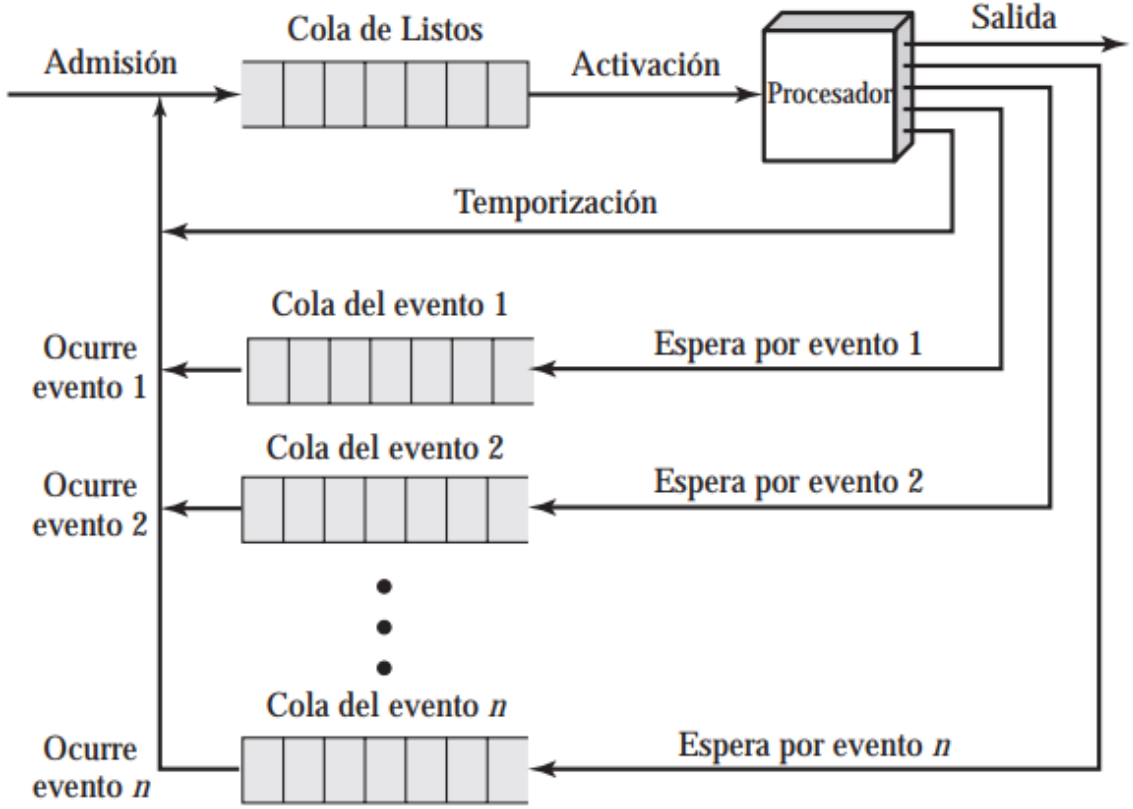
Figura 3.6. Modelo de proceso de cinco estados.

Colas de bloqueados

- La figura (a) sugiere la forma de aplicar un esquema de dos colas: la colas de Listos y la cola de Bloqueados.
- En los sistemas operativos con muchos procesos, esto puede significar cientos o incluso miles de procesos en esta lista, por lo que sería mucho más eficiente tener una cola por cada evento. De esta forma, cuando sucede un evento, la lista entera de procesos de la cola correspondiente se movería al estado de Listo (Figura (b)).



(a) Cola simple de Bloqueados



(b) Múltiples colas de Bloqueados

Procesos suspendidos

- Los tres principales estados descritos (Listo, Ejecutando, Bloqueado) proporcionan una forma sistemática de modelizar el comportamiento de los procesos y diseñar la implementación del sistema operativo. Se han construido algunos sistemas operativos utilizando únicamente estos tres estados.
- Expandir la memoria principal para acomodar más procesos. Hay dos fallos en esta solución. Primero, existe un coste asociado a la memoria principal, que, desde un coste reducido a nivel de bytes, comienza a incrementarse según nos acercamos a un almacenamiento de gigabytes. Segundo, el apetito de los programas a nivel de memoria ha crecido tan rápido como ha bajado el coste de las memorias. De forma que las grandes memorias actuales han llevado a ejecutar procesos de gran tamaño, no más procesos.

Procesos suspendidos

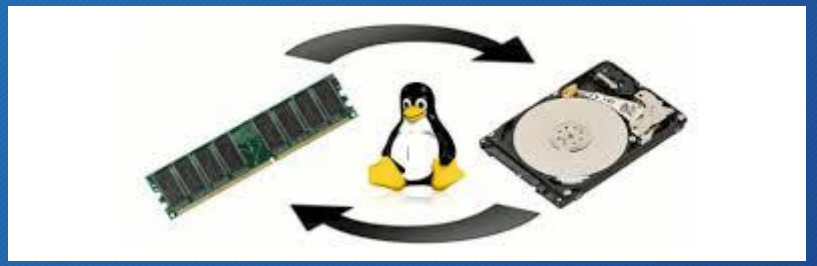
- Otra solución es el swapping (memoria de intercambio), que implica mover parte o todo el proceso de memoria principal al disco. Cuando ninguno de los procesos en memoria principal se encuentra en estado Listo, el sistema operativo intercambia uno de los procesos bloqueados a disco, en la cola de Suspendidos. Esta es una lista de procesos existentes que han sido temporalmente expulsados de la memoria principal, o suspendidos. El sistema operativo trae otro proceso de la cola de Suspendidos o responde a una solicitud de un nuevo proceso. La ejecución continúa con los nuevos procesos que han llegado.

Procesos suspendidos

- Con el uso de swapping tal y como se ha escrito, debe añadirse un nuevo estado a nuestro modelo de comportamiento de procesos, el estado Suspendido. Cuando todos los procesos en memoria principal se encuentran en estado Bloqueado, el sistema operativo puede suspender un proceso poniéndolo en el estado Suspendido y transfiriéndolo a disco. El espacio que se libera en memoria principal puede usarse para traer a otro proceso.

Procesos suspendidos

Sistema operativo

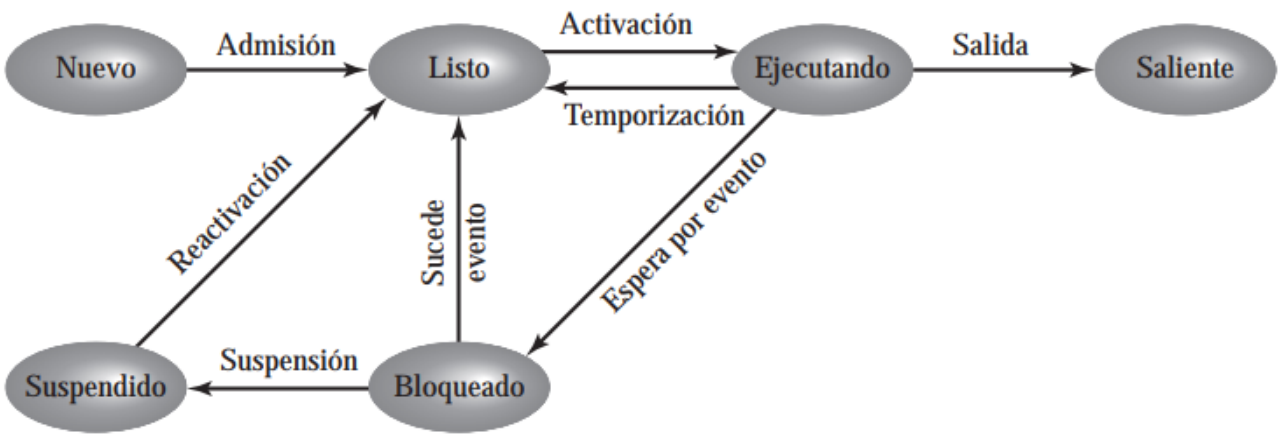


Cargar nuevo proceso

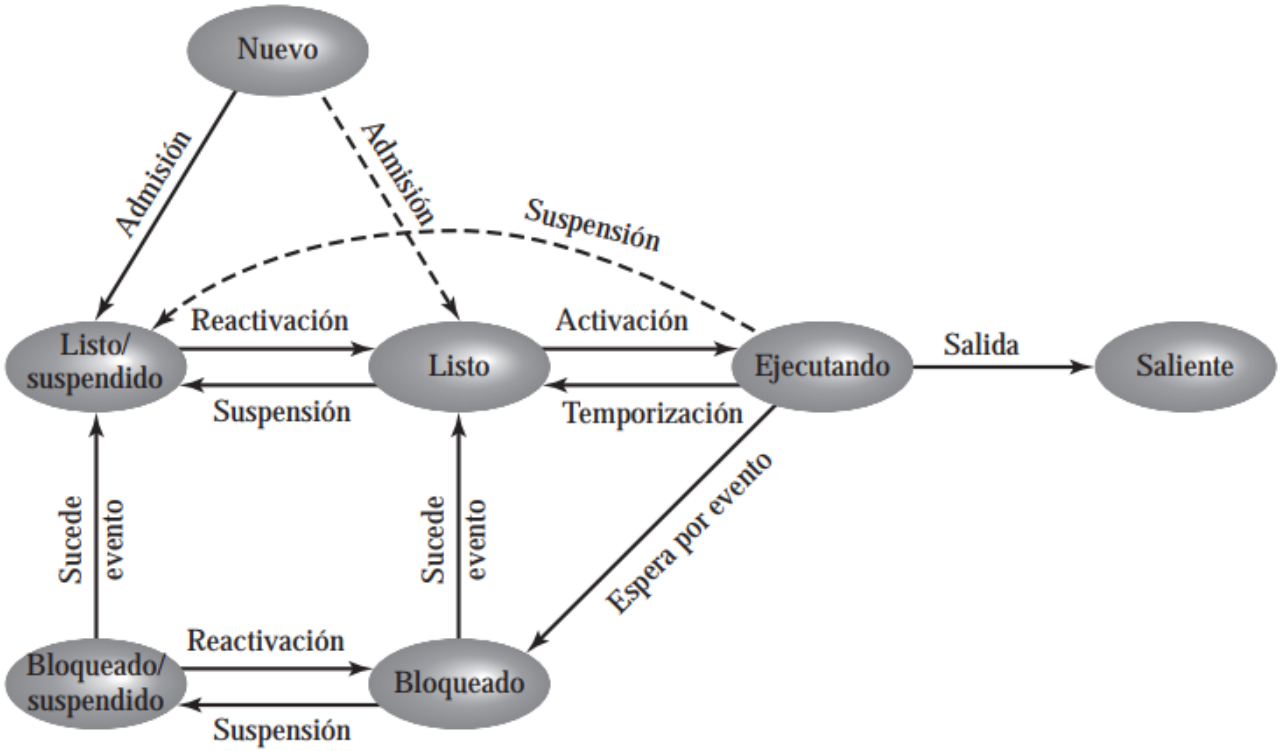
Cargar proceso suspendido con anterioridad

Procesos suspendidos

- De esta forma, necesitamos replantear este aspecto del diseño. Hay dos conceptos independientes aquí: si un proceso está esperando a un evento (Bloqueado o no) y si un proceso está transferido de memoria a disco (suspendido o no).



(a) Con un único estado Suspendido

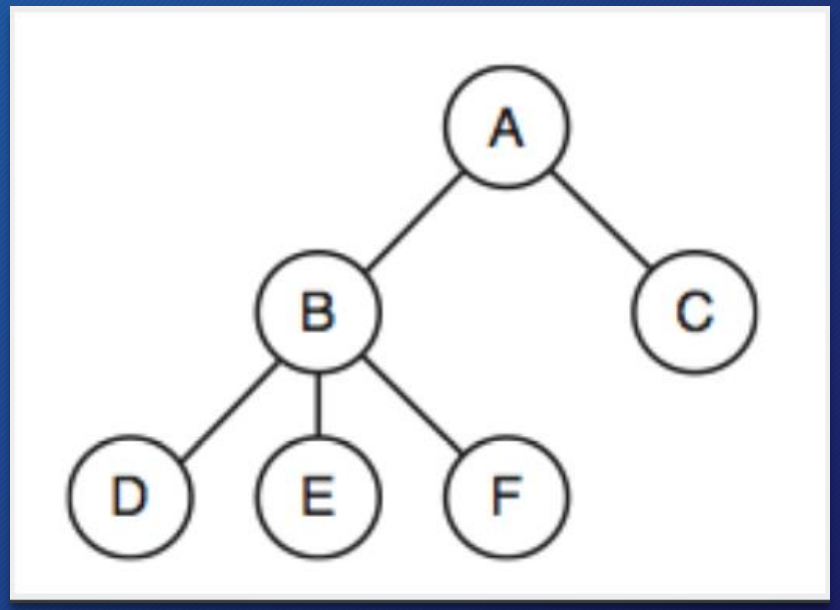


(b) Con dos estados Suspendido

En síntesis…..

- Las llamadas al sistema de administración de procesos clave son las que se encargan de la creación y la terminación de los procesos.
- Por ejemplo: El proceso llamado intérprete de comandos o shell, lee comandos de una terminal. El usuario acaba de escribir un comando, solicitando la compilación de un programa.
- El shell debe entonces crear un proceso para ejecutar el compilador. Cuando ese proceso ha terminado la compilación, ejecuta una llamada al sistema para terminarse a sí mismo.

- Un proceso puede crear uno o más procesos aparte (conocidos como procesos hijos) y estos procesos a su vez pueden crear procesos hijos, llegamos rápidamente la estructura de árbol de procesos. Los procesos relacionados que cooperan para realizar un cierto trabajo a menudo necesitan comunicarse entre sí y sincronizar sus actividades. A esta comunicación se le conoce como " comunicación entre procesos ".



- Cada persona autorizada para utilizar un sistema recibe una UID (User Identification, Identificación de usuario) que el administrador del sistema le asigna. Cada proceso iniciado tiene el UID de la persona que lo inició. Un proceso hijo tiene el mismo UID que su padre. Los usuarios pueden ser miembros de grupos, cada uno de los cuales tiene una GID (Group Identification, Identificación de grupo).
- Una UID conocida como super usuario (superuser en UNIX) tiene poder especial y puede violar muchas de las reglas de protección. En instalaciones extensas, sólo el administrador del sistema conoce la contraseña requerida para convertirse en super usuario, pero muchos de los usuarios ordinarios (en especial los estudiantes) dedican un esfuerzo considerable para tratar de encontrar fallas en el sistema que les permitan convertirse en super usuario sin la contraseña.

Planificación de procesos

- Conjunto de políticas y mecanismos incorporados al sistema operativo, a través de un módulo denominado planificador, que debe decidir cuál de los procesos en condiciones de ser ejecutado conviene ser despachado primero y qué orden de ejecución debe seguirse. Esto debe realizarse sin perder de vista su principal objetivo que consiste en el máximo aprovechamiento del sistema, lo que implica proveer un buen servicio a los procesos existentes en un momento dado.

Objetivos de la Planificación de procesos

- La Planificación de procesos tiene como principales objetivos la equidad, la eficacia, el tiempo de respuesta, el tiempo de regreso y el rendimiento.

Equidad: Todos los procesos deben ser atendidos.

Eficacia: El procesador debe estar ocupado el 100% del tiempo.

Tiempo de respuesta: El tiempo empleado en dar respuesta a las solicitudes del usuario debe ser el menor posible.

Tiempo de regreso: Reducir al mínimo el tiempo de espera de los resultados esperados por los usuarios por lotes.

Rendimiento: Maximizar el número de tareas que se procesan por cada hora.

Algoritmos de Planificación

- **Primero en llegar primero en salir:** Conocido como FIFO (First In First Out). Este algoritmo emplea una cola de procesos, asignando un lugar a cada proceso por el orden de llegada. Cuando el proceso llega es puesto en su lugar en la cola después del que llegó antes que él y se pone en estado de listo. Cuando un proceso comienza a ejecutarse no se interrumpe su ejecución hasta que termina de hacerlo.

- **Prioridad al más corto:** Su nombre es SJF (Shortest Job First). El proceso que se encuentra en ejecución cambiará de estado voluntariamente, o sea, no tendrá un tiempo de ejecución determinado para el proceso. A cada proceso se le asigna el tiempo que usará cuando vuelva a estar en ejecución, y se irá ejecutando el que tenga un menor tiempo asignado. Si se da el caso de que dos procesos tengan igual valor en ese aspecto emplea el algoritmo FIFO.

- **Round Robin:** A cada proceso se le asigna un tiempo determinado para su ejecución, el mismo tiempo para todos. En caso de que un proceso no pueda ser ejecutado completamente en ese tiempo se continuará su ejecución después de que todos los procesos restantes sean ejecutados durante el tiempo establecido. Este es un algoritmo basado en FIFO que trata la cola de procesos que se encuentran en estado de listos como una cola circular.

Algoritmos de Planificación

- **Planificación por prioridad:** En este tipo de planificación a cada proceso se le asigna una prioridad siguiendo un criterio determinado, y de acuerdo con esa prioridad será el orden en que se atienda cada proceso.

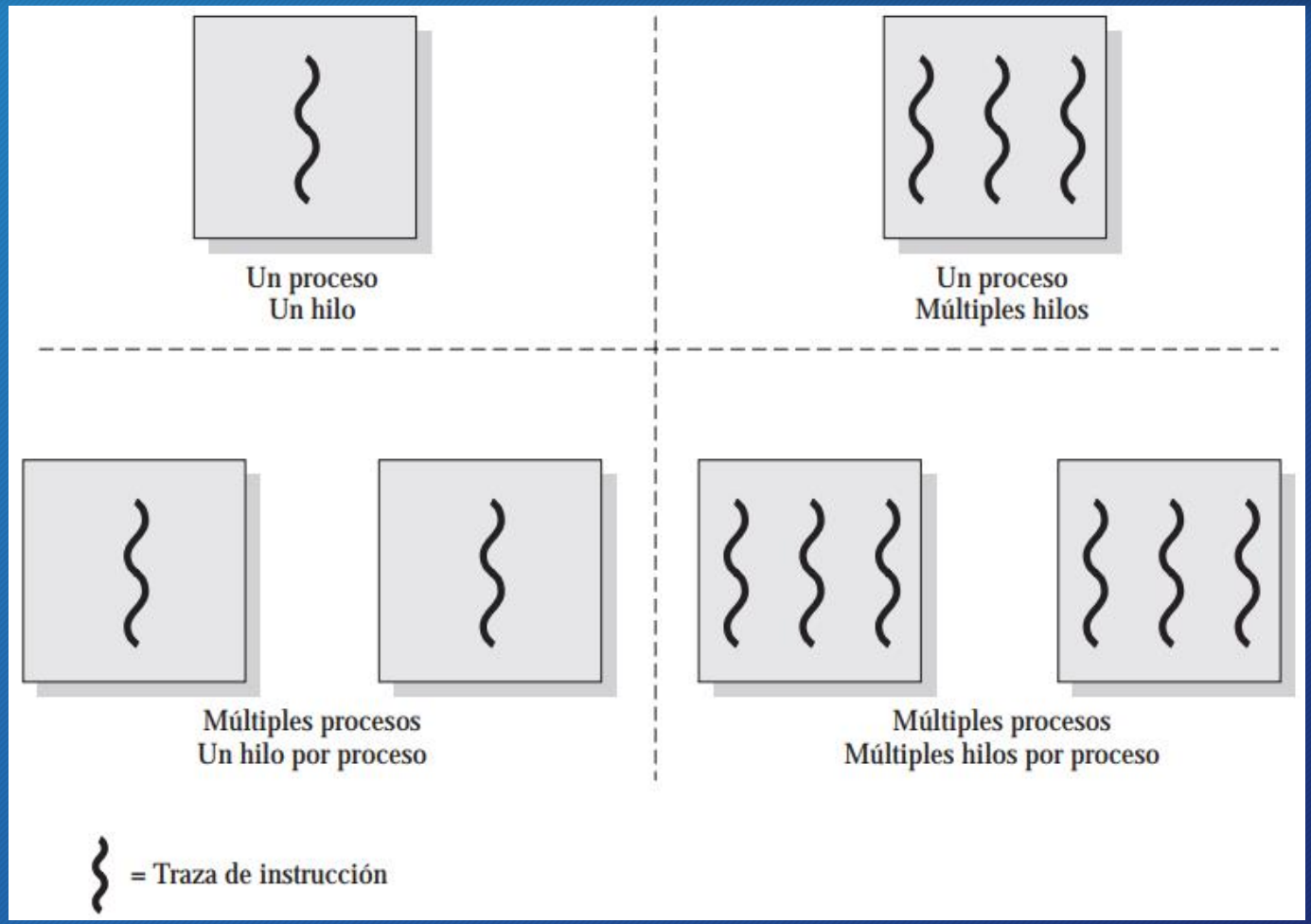
- **Planificación garantizada:** Para realizar esta planificación el sistema tiene en cuenta el número de usuarios que deben ser atendidos. Para un número "n" de usuarios se asignará a cada uno un tiempo de ejecución igual a 1/n.

- **Planificación de Colas Múltiples:** El nombre se deriva de MQS (Multilevel Queue Schedulling). En este algoritmo la cola de procesos que se encuentran en estado de listos es dividida en un número determinado de colas más pequeñas. Los procesos son clasificados mediante un criterio para determinar en qué cola será colocado cada uno cuando quede en estado de listo. Cada cola puede manejar un algoritmo de planificación diferente a las demás.

Procesos e Hilos

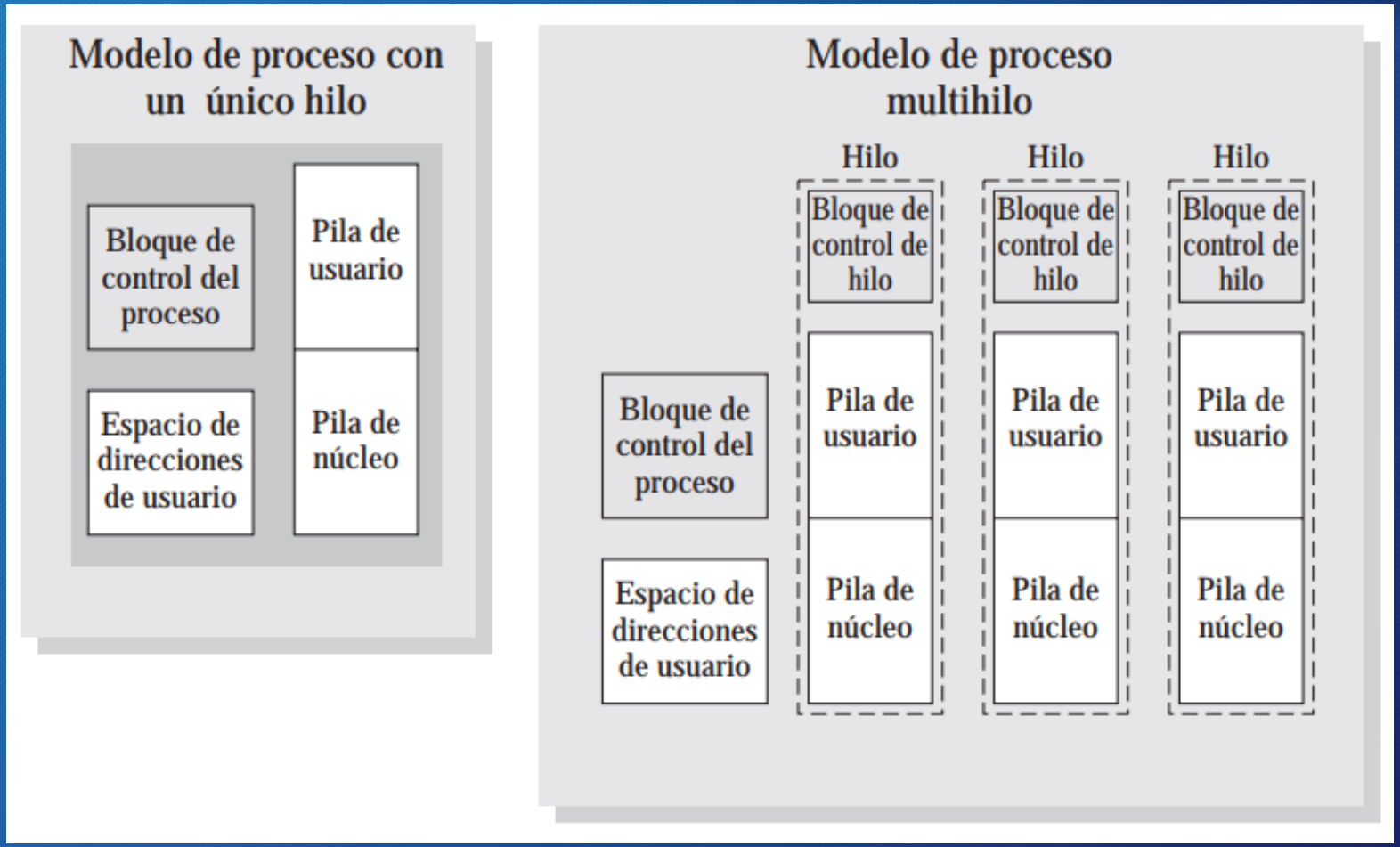
- ¿Qué es un hilo?
- Es una característica que permite a una aplicación realizar varias tareas simultáneamente.
- MULTIHILO
- Multihilo se refiere a la capacidad de un sistema operativo de dar soporte a múltiples hilos de ejecución en un solo proceso. El enfoque tradicional de un solo hilo de ejecución por proceso, en el que no se identifica con el concepto de hilo, se conoce como estrategia monohilo.

Representación de hilos y procesos



- Dentro de un proceso puede haber uno o más hilos, cada uno con:
  - Un estado de ejecución por hilo (Ejecutando, Listo, etc.).
  - Un contexto de hilo que se almacena cuando no está en ejecución; una forma de ver a un hilo es como un contador de programa independiente dentro de un proceso.
  - Una pila de ejecución.
  - Por cada hilo, espacio de almacenamiento para variables locales.
  - Acceso a la memoria y recursos de su proceso, compartido con todos los hilos de su mismo proceso.

Modelos de proceso con un único hilo y multihilo



Beneficios de los hilos

- Los mayores beneficios de los hilos provienen de las consecuencias del rendimiento:
  1. Lleva mucho menos tiempo crear un nuevo hilo en un proceso existente que crear un proceso totalmente nuevo. Los estudios realizados por los que desarrollaron el sistema operativo Mach muestran que la creación de un hilo es diez veces más rápida que la creación de un proceso en UNIX [TEVA87].
  2. Lleva menos tiempo finalizar un hilo que un proceso.
  3. Lleva menos tiempo cambiar entre dos hilos dentro del mismo proceso.
  4. Los hilos mejoran la eficiencia de la comunicación entre diferentes programas que están ejecutando. En la mayor parte de los sistemas operativos, la comunicación entre procesos independientes requiere la intervención del núcleo para proporcionar protección y los mecanismos necesarios de comunicación. Sin embargo, ya que los hilos dentro de un mismo proceso comparten memoria y archivos, se pueden comunicar entre ellos sin necesidad de invocar al núcleo.

Ejemplos de uso de hilos

- Trabajo en primer plano y en segundo plano. Por ejemplo, en un programa de hoja de cálculo, un hilo podría mostrar menús y leer la entrada de usuario, mientras otro hilo ejecuta los mandatos de usuario y actualiza la hoja de cálculo. Esta forma de trabajo a menudo incrementa la velocidad que se percibe de la aplicación, permitiendo al programa solicitar el siguiente mandato antes de que el mandato anterior esté completado.
- Procesamiento asíncrono. Los elementos asíncronos de un programa se pueden implementar como hilos. Por ejemplo, se puede diseñar un procesador de textos con protección contra un fallo de corriente que escriba el buffer de su memoria RAM a disco una vez por minuto. Se puede crear un hilo cuyo único trabajo sea crear una copia de seguridad periódicamente y que se planifique directamente a través del sistema operativo; no se necesita código adicional en el programa principal que proporcione control de tiempo o que coordine la entrada/salida.
- Velocidad de ejecución. Un proceso multihilo puede computar una serie de datos mientras que lee los siguientes de un dispositivo. En un sistema multiprocesador pueden estar ejecutando simultáneamente múltiples hilos de un mismo proceso. De esta forma, aunque un hilo pueda estar bloqueado por una operación de E/S mientras lee datos, otro hilo puede estar ejecutando.
- Estructura modular de programas. Los programas que realizan diversas tareas o que tienen varias fuentes y destinos de entrada y salida, se pueden diseñar e implementar más fácilmente usando hilos.

Estados de hilos