Procesos. Gestión de Procesos

En este módulo se describe el concepto de proceso, cómo se representa, cuáles son las estructuras de datos y operaciones que permiten al sistema operativo gestionar procesos.

Tabla de Contenidos

Autor: Lina García Cabrera

1 Introducción

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos

Los **sistemas operativos multiprogramados** necesitan el concepto **proceso**. El sistema operativo debe entremezclar la ejecución de un número de procesos para maximizar la utilización de los recursos del ordenador. Al mismo tiempo, los sistemas de tiempo compartido deben proporcionar un tiempo de respuesta razonable.

El concepto de proceso es clave en los sistemas operativos modernos. La gestión del procesador mediante multiprogramación revolucionó la concepción de los sistemas operativos, e introdujo el término **proceso** como elemento necesario para realizar dicha gestión. Sería muy conveniente, pues, tener claro el concepto de **multiprogramación**.

El sistema operativo debe asignar recursos a los procesos y planificicarlos de acuerdo con una política específica (ciertas funciones o aplicaciones son de mayor prioridad), mientras impide los interbloqueos. Por último, el sistema operativo debe ofrecer un soporte para llevar a cabo la sincronización y la comunicación entre procesos.

> Por lo demás, este tema trata sobre la **definición de proceso**, el estudio de sus propiedades, y la gestión que realiza el sistema operativo para crear la abstracción de proceso, aunque esto último se completará en el módulo de planificación. De igual modo, en otro módulo, descubriremos que el concepto de proceso encierra, en realidad, dos características potencialmente independientes: por un lado, es una unidad a la que se le asigna y posee recursos y, por otro, es una unidad planificable. Basándonos en esta distinción introduciremos el estudio de los hilos o hebras (*threads*).

Un ***proceso*** *es un programa en ejecución*. Un proceso necesita ciertos recursos (tiempo de CPU, memoria, ficheros, dispositivos E/S) para ejecutarse. Los sistemas constan de una colección de procesos que se ejecutan de forma concurrente.

OBJETIVOS

- Conocer el concepto de proceso en el que se basa un sistema multiprogramado.
- Saber cómo se describe un proceso y cómo lo gestiona el sistema operativo.

2 ¿Qué es un proceso?

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos

Hasta ahora hemos utilizado siempre el término programa. A partir de ahora ***distinguiremos entre programa y proceso***.

Un **programa** es una secuencia de instrucciones escrita en un lenguaje dado.

Un **proceso** es una instancia de ejecución de un programa, caracterizado por su contador de programa, su palabra de estado, sus registros del procesador, su segmento de texto, pila y datos, etc.

Un programa es un concepto estático, mientras que un **proceso es un concepto dinámico**. Es posible que un programa sea ejecutado por varios usuarios en un sistema multiusuario, por cada una de estas ejecuciones existirá un proceso, con su contador de programa, registros, etc. El sistema operativo necesita el concepto de proceso para poder gestionar el procesador mediante la técnica de multiprogramación o de tiempo compartido, de hecho, **el proceso es la unidad planificable**, o de asignación de la CPU.

3 Estados de un proceso

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Durante su vida, un proceso puede pasar por una serie de **estados** discretos, algunos de ellos son:

- **En ejecución**: El proceso ocupa la CPU actualmente, es decir, se está ejecutando.
- **Listo o preparado**: El proceso dispone de todos los recursos para su ejecución, sólo le falta la CPU.
- **Bloqueado**: Al proceso le falta algún recurso para poder seguir ejecutándose, además de la CPU. Por recurso se pueden entender un dispositivo, un dato, etc. El proceso necesita que ocurra algún evento que le permita poder proseguir su ejecución.

Hay otros estados de los procesos, pero en la presente exposición se tratarán estos tres. Por sencillez, se considera un sistema con una sola CPU, aunque no es difícil la extensión a múltiples procesadores.

Solamente puede haber **un proceso en ejecución a la vez**, pero pueden existir **varios listos/preparados y varios pueden estar bloqueados**. Así pues, se forman una lista de procesos listos y otra de procesos bloqueados:

- La **lista de procesos listos** se ordena por prioridad o criterio, de manera que el siguiente proceso que reciba la CPU será el primero de la lista. *Sólo hay una lista de procesos preparados*.
- La **lista de procesos bloqueados** normalmente no está ordenada; los procesos no se desbloquean (es decir, no pasan a ser procesos listos) en orden de prioridad, sino que lo hacen en el orden de ocurrencia de los eventos que están esperando. Normalmente, *suelen existir varias listas de bloqueados*, una por cada tipo evento que esperan. Hay situaciones en las que varios procesos pueden bloquearse esperando la ocurrencia del mismo evento; en tales casos es común asignar prioridades a los procesos que esperan.

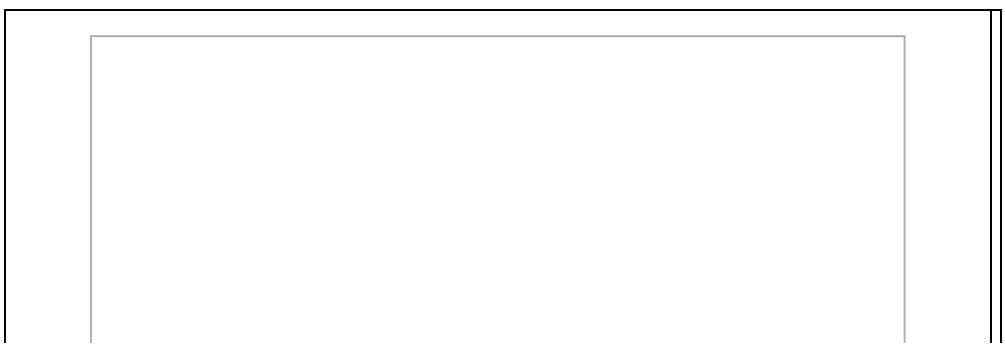
3.1 Transición de estado de los procesos

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos

A continuación se dan ejemplos de eventos que pueden provocar transiciones de estado en un proceso en este modelo de tres estados (para más detalle ver la Figura 2 - 1):

- **En ejecución --> Bloqueado**: **al iniciar una operación de E/S**, al realizar una operación WAIT sobre un semáforo a cero (en otra asignatura dedicada a la concurrencia se estudiarán los semáforos), al solicitar un recurso no compartible asignado a otro proceso.
- **En ejecución --> Listo**: por ejemplo, en un sistema de tiempo compartido, cuando el proceso que ocupa la CPU lleva demasiado tiempo ejecutándose continuamente (agota su cuanto), el sistema operativo decide que otro proceso ocupe la CPU, pasando el proceso que ocupaba la CPU a estado listo.
- **Listo --> en ejecución**: cuando lo requiere el planificador de la CPU (veremos el planificador de la CPU en el módulo de planificación de procesos).
- **Bloqueado --> Listo**: se dispone del recurso por el que se había bloqueado el proceso. Por ejemplo, termina la operación de E/S, o se produce una operación SIGNAL sobre el semáforo en que se bloqueó el proceso, no habiendo otros procesos bloqueados en el semáforo.

Obsérvese que de las cuatro transiciones de estado posibles, la única iniciada por el proceso de usuario es el bloqueo, las otras tres son iniciadas por entidades externas al proceso.



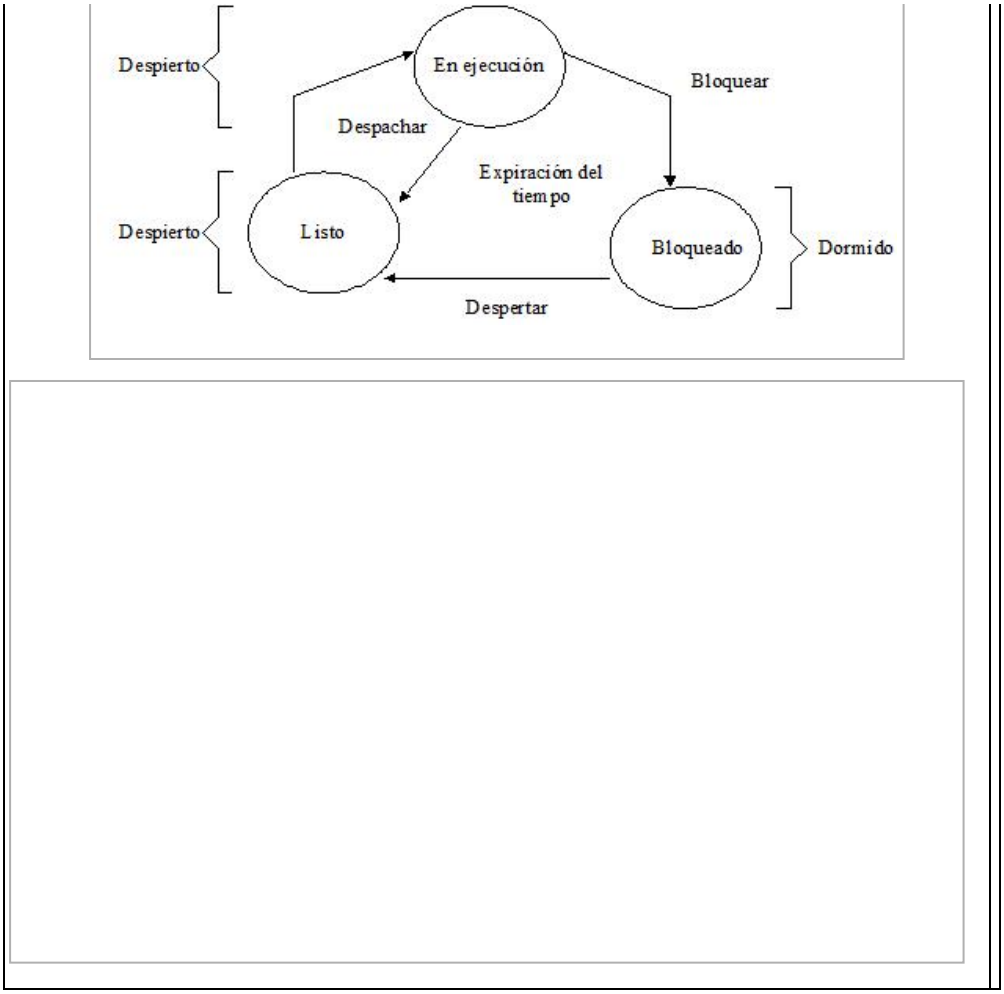


Figura 1. Transiciones de estado de los procesos.

4 Descripción de un proceso

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
De algún modo, debemos hacer una pregunta fundamental: **¿cuál es la manifestación física de un proceso?** Como mínimo debe incluir:

- un **programa o conjunto de programas** que sean ejecutados.
- Asociados a estos programas hay un conjunto de ubicaciones de datos para las **variables locales y globales**, y las **constantes definidas**. Así pues, un proceso constará, al menos, de la memoria suficiente para albergar los **programas y los datos** del proceso.
- Además, en la ejecución de un programa entra en juego normalmente una **pila**, que se utiliza para llevar la cuenta de las llamadas a procedimientos y de los parámetros que se pasan entre los procedimientos.
- Por último, asociado a cada proceso hay una serie de **atributos que utiliza el sistema operativo para el control del proceso**. Estos atributos se agrupan en una estructura de datos que se conoce como **bloque de control de proceso** (*Process Control Block*, PCB) o **descriptor de proceso**.

A la conjunción de **programa, datos, pila y atributos** (contenidos en el **bloque de control de proceso**) se le llama **imagen del proceso**.

Un **proceso** es código, datos, pila y bloque de control de proceso (estructura de datos que almacena los atributos que le permiten al sistema operativo controlar su ejecución).

4.1 El bloque de control de proceso

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
El **bloque de control de proceso** es la estructura de datos central y más importante de un sistema operativo. Cada bloque de control de proceso contiene toda la información de un proceso que necesita un sistema operativo para su control. Estos bloques son leídos y/o modificados por casi todos los módulos de un sistema operativo, incluyendo aquellos que tienen que ver con la planificación, la asignación de recursos, el tratamiento de interrupciones y el análisis y supervisión del rendimiento.

- Puede decirse que el ***conjunto de los bloques de control de procesos definen el estado del sistema operativo***.
- El conjunto de todos los PCB's se guarda en una estructura del sistema operativo llamada **tabla de procesos**, la cual se puede implementar como un vector o una lista enlazada. La **tabla de procesos** reside en memoria principal, debido a su alta frecuencia de consulta.

El sistema operativo gestiona procesos y recursos, por tanto, necesita guardar información sobre el estado de cada proceso y recurso del sistema. Esta información se guarda en **tablas** de memoria, de E/S, de ficheros y de procesos.

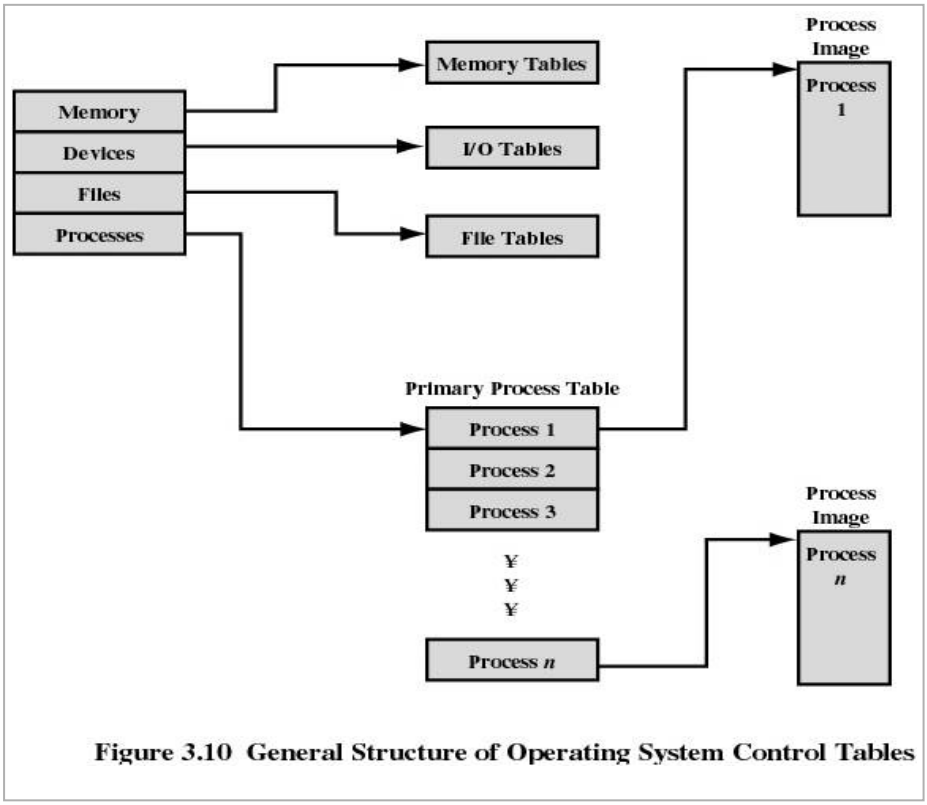


Figura 2. Tablas de control de los sistemas operativos.
En un sistema de multiprogramación, se requiere una gran cantidad de información de cada proceso para su administración. Los distintos sistemas organizarán esta información de modo diferente. En general, se puede agrupar la información de los PCB's en tres categorías:

- **Identificación del proceso**
  Con respecto a la **identificación del proceso**, en casi todos los sistemas operativos se le asigna a cada proceso un identificador numérico único (ID). Este identificador nos servirá para localizarlo dentro de la tabla de procesos. Cuando se permite que los procesos creen otros procesos, se utilizan identificadores para señalar al padre y a los descendientes de cada proceso. Además de estos, un proceso también puede tener asignado un identificador de usuario, que indica a quién pertenece el proceso (UID).
- **Información de estado del procesador**
  Básicamente, está formada por el contenido de los registros del procesador. Por supuesto, mientras el proceso está ejecutándose, la información está en los registros. Cuando se interrumpe el proceso, toda la información de los registros debe salvarse de forma que pueda restaurarse cuando el proceso reanude su ejecución. La naturaleza y número de registros involucrados depende del diseño del procesador. Normalmente, en el conjunto de registros se incluyen los registros visibles para el usuario, los registros de control y de estado (contador de programa y palabra de estado) y los punteros pila.

- **Información de control del proceso**
  Esta es la información adicional necesaria para que el sistema operativo controle y coordine los diferentes procesos activos. Como, por ejemplo, información de planificación y estado (estado del proceso, su prioridad, información de planificación, suceso), punteros a estructuras de datos (los procesos que esperan en un dispositivo, en un semáforo), punteros a zonas de memoria del proceso, recursos controlados por el proceso (ficheros abiertos), etc.

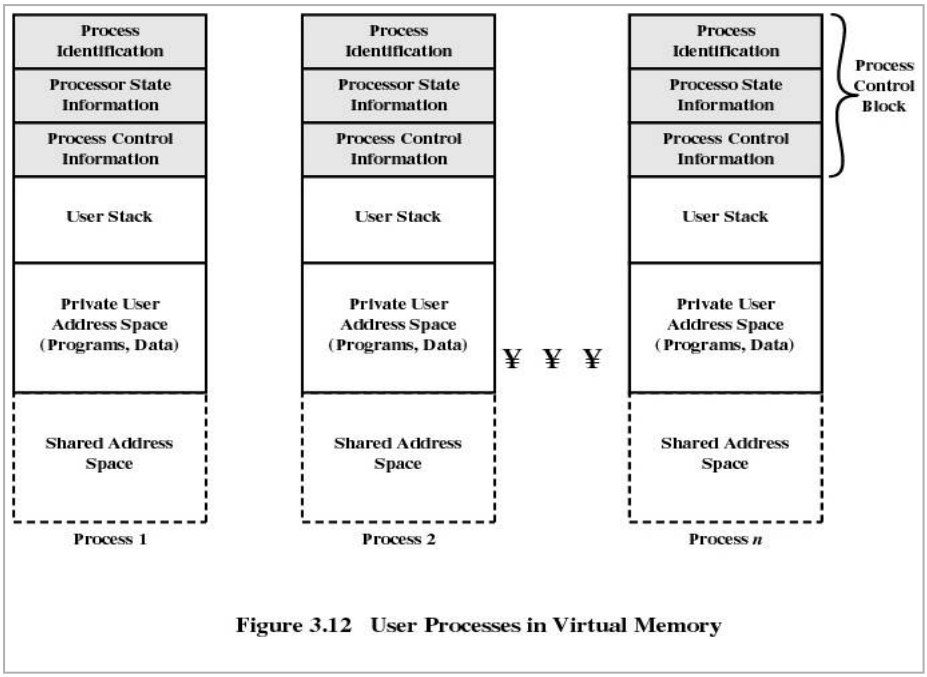


Figure 3.12 User Processes in Virtual Memory

Figura 3. Bloque de control de procesos.

Así pues, el **PCB es la entidad que define un proceso en el sistema operativo**. Dado que los PCB necesitan ser manejados con eficiencia por el sistema operativo, muchos ordenadores tienen un **registro de hardware** que siempre apunta hacia el PCB del proceso que se está ejecutando. A menudo existen **instrucciones hardware** que cargan en el PCB información sobre su entorno, y la recuperan con rapidez.

4.2 Operaciones con procesos

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Los sistemas que administran procesos deben ser capaces de realizar ciertas operaciones sobre y con los procesos. Tales operaciones incluyen:

- crear y destruir un proceso
- suspender y reanudar un proceso
- cambiar la prioridad de un proceso
- bloquear y “desbloquear” un proceso
- planificar un proceso (asignarle la CPU)
- permitir que un proceso se comunique con otro (a esto se denomina comunicación entre procesos, y se estudiará en la asignatura de concurrencia).

Crear un proceso

Crear un proceso implica muchas operaciones, tales como:

- buscarle un **identificador**
- **insertarlo en la tabla de procesos**
- determinar la **prioridad inicial** del proceso
- **crear el PCB**
- asignar los recursos iniciales al proceso

Un proceso puede crear un nuevo proceso. Si lo hace, el proceso creador se denomina **proceso padre**, y el proceso creado, **proceso hijo**. Sólo se necesita un padre para crear un hijo. Tal creación origina una **estructura jerárquica de procesos**, en la cual cada hijo tiene sólo un padre, pero un padre puede tener muchos hijos.

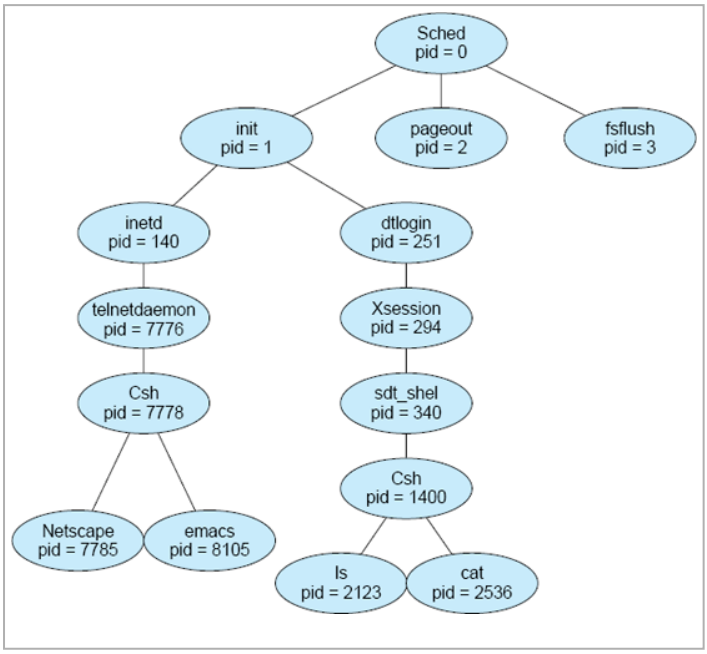


Figura 4. Árbol de procesos de un sistema Solaris (*pageout* gestiona la memoria, *fslush* gestiona sistema de ficheros, *inetd* servicios de red, *dtlogin* pantalla inicio del usuario).

En el sistema operativo UNIX la llamada al sistema `fork()` **crea un proceso hijo. El nuevo proceso consta es una copia del espacio de direcciones del proceso padre. Ambos procesos continúan su ejecución en la instrucción siguiente al `fork()`, con una diferencia: el código de retorno la llamada `fork()` del proceso nuevo (hijo) es cero, mientras que al proceso padre se le devuelve el identificador de proceso del hijo (que es distinto de cero).**

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

int main()
{
pid_t pid;

   /* fork a child process */
   pid = fork();

   if (pid < 0) {/* error occurred */
      fprintf(stderr, "Fork Failed");
      exit(-1);
   }
   else if (pid == 0) {/* child process */
      execlp("/bin/ls","ls",NULL);
   }
   else {/* parent process */
      /* parent will wait for the child to complete */
      wait(NULL);
      printf("Child Complete");
      exit(0);
   }
}
```

Normalmente, el **proceso hijo utiliza la llamada `exec()`**después de la llamada al sistema **`fork()`**, para sustituir en memoria el código del proceso con un nuevo programa. La llamada **`exec()`** carga el archivo binario en memoria (destruyendo la imagen del programa que contenía la llamada al sistema `exec()`) e inicia su ejecución.
El **proceso padre puede continuar con su ejecución o puede ejecutar la llamada al sistema `wait()`**para pasar al estado bloqueado y esperar hasta que termine la ejecución del hijo.

Eliminar un proceso

Destruir un proceso implica **eliminarlo** del sistema:

- Se le borra de las tablas o listas del sistema,
- sus recursos se devuelven al sistema y
- su PCB se borra (es decir, el espacio de memoria ocupado por su PCB se devuelve al espacio de memoria disponible).

La destrucción de un proceso es más difícil cuando éste ha creado otros procesos. En algunos sistemas un proceso hijo se destruye automáticamente cuando su padre es destruido; en otros sistemas, los procesos creados son independientes de su padre, y la destrucción de este último no tiene efecto sobre sus hijos.

Suspender y Reanudar un proceso

Al modelo anterior de tres estados (Listo, En ejecución y Bloqueado) se añaden un nuevo estado el suspendido. La **suspensión** de un proceso ***implica su desalojo de la memoria principal y su transferencia a disco***.

Un proceso suspendido no puede proseguir hasta que no lo **reanuda** otro proceso. La suspensión es una operación importante, y ha sido puesta en práctica de diferentes formas en diversos sistemas. La suspensión dura por lo normal sólo periodos breves. Muchas veces, el sistema efectúa las suspensiones para eliminar temporalmente ciertos procesos, y así reducir la carga del sistema durante una situación de carga máxima.

Cuando hay suspensiones largas se debe liberar los recursos del proceso. La decisión de liberar o no los recursos depende mucho de la naturaleza de cada recurso. La memoria principal debe ser liberada de inmediato cuando se suspenda un proceso; un dispositivo o archivo (no crítico) puede ser retenido brevemente por un proceso suspendido, pero debe ser liberada si el proceso se suspende por un periodo largo o indefinido. Reanudar (o activar) un proceso implica reiniciarlo a partir del punto en el que se suspendió.

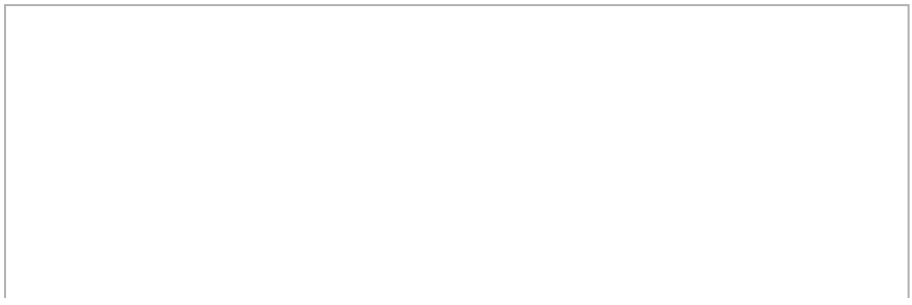
Cambiar la prioridad de un proceso normalmente no implica más que modificar el valor de la prioridad en el PCB.

4.3 Suspensión y reanudación

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Las operaciones de suspensión y reanudación de un proceso son importantes por diversas razones.

- Si un **sistema está funcionando mal**, y es probable que falle, se puede suspender a los procesos activos para reanudarlos cuando se haya corregido el problema.
- Un usuario que **desconfíe de los resultados parciales de un proceso** puede suspenderlo (en lugar de abortarlo) hasta que verifique si el proceso funciona correctamente o no.
- Algunos procesos se pueden suspender como respuesta a las **fluctuaciones a corto plazo de la carga del sistema**, y reanudarse cuando la carga vuelva a niveles normales.

La Figura 5 muestra el diagrama de transiciones de estado de los procesos, modificado para incluir las operaciones de suspensión y reanudación. Se han añadido dos estados nuevos, denominados **suspendido_listo** y **suspendido_bloqueado**; no hay necesidad de un estado suspendido_en_ejecución. Sobre la línea discontinua se encuentran los **estados activos**, y debajo de ella los **estados suspendidos**.



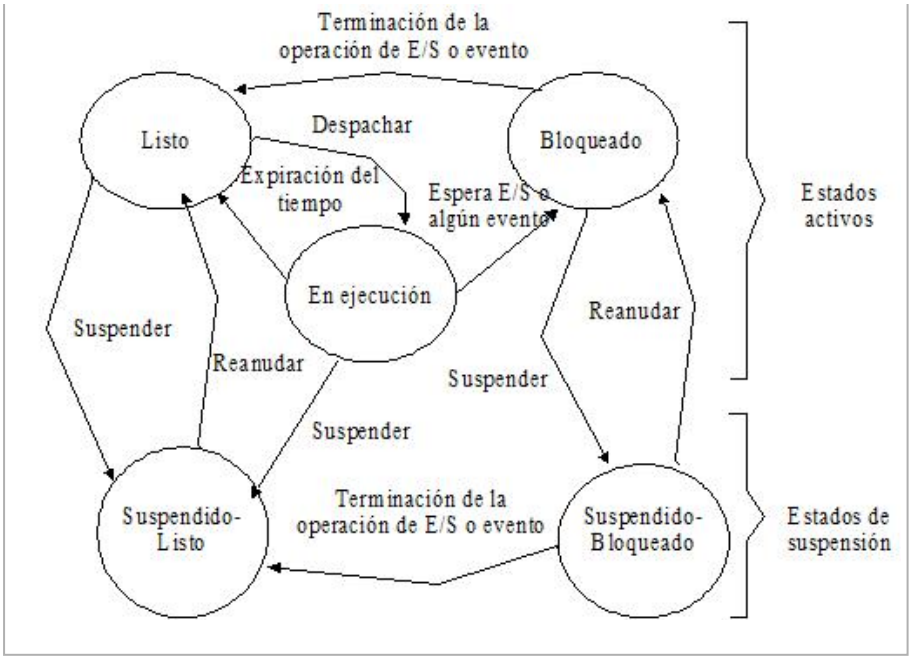


Figura 5. Transiciones de estado de los procesos.

Una **suspensión** puede ser **iniciada por el propio proceso o por otro**. En un sistema con un sólo procesador, el proceso en ejecución puede suspenderse a sí mismo; ningún otro proceso podría estar en ejecución al mismo tiempo para realizar la suspensión (aunque otro proceso sí podría solicitar la suspensión cuando se ejecute). En un sistema de múltiples procesadores, un proceso en ejecución puede suspender a otro que se esté ejecutando en ese mismo momento en un procesador diferente.

Solamente otro proceso puede suspender a un proceso listo/preparado. Un proceso puede hacer que otro proceso que se encuentre en el estado suspendido_listo pase al estado listo. Un proceso puede suspender a otro proceso que esté bloqueado, y hacerlo pasar de suspendido_bloqueado a bloqueado.

Se podría alegar que en lugar de suspender un proceso bloqueado, sería mejor esperar hasta que ocurriera el evento que esperaba; entonces el proceso podría suspenderse y pasarse al estado suspendido_listo. Por desgracia, puede ser que nunca ocurra el evento o que se postergue indefinidamente. Así pues, el diseñador debe decidir si realiza la suspensión del proceso bloqueado o establece un mecanismo mediante el cual se realice la suspensión desde el estado listo cuando ocurra el término de la operación. Como la suspensión es, por lo normal, una actividad de alta prioridad, se debe realizar de inmediato. Cuando ocurre finalmente el evento, el proceso suspendido_bloqueado pasa a suspendido_listo.

En el **módulo de planificación y de gestión de memoria** se analiza la forma en que el sistema operativo utiliza las operaciones de **suspensión y reanudación para equilibrar la carga del sistema**.

5 Control de un proceso

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Los sistemas operativos necesitan los procesos para gestionar y controlar la ejecución de los programas.

5.1 Modos de Ejecución

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Antes de continuar la discusión sobre la forma en que el sistema operativo gestiona los procesos, hace falta distinguir entre **el modo de ejecución del procesador que normalmente se asocia con el sistema operativo y el modo que normalmente se asocia con los programas de usuario**.

Ciertas instrucciones máquina pueden ejecutarse sólo en modo privilegiado. Entre éstas están la lectura o modificación de registros de control (como la palabra de estado), instrucciones primitivas de E/S e instrucciones relativas a la gestión de memoria. Y solamente se puede **acceder a ciertas zonas de memoria en el modo privilegiado**.

El modo de menor privilegio se conoce como **modo usuario**, y el de mayor privilegio como **modo de sistema, supervisor o núcleo**.

La razón por la que se usan dos modos debe quedar clara. Es necesario proteger al sistema operativo y a las estructuras de datos importantes, tales como los bloques de control de procesos, de las inferencias de los programas de usuario. En el modo núcleo, el software tiene control completo del procesador y de todas las instrucciones, registros y memoria.

Surgen dos preguntas: **¿cómo conoce el procesador en qué modo se ejecuta?**, **¿cómo se cambia de modo?**. Para la primera pregunta, hay un ***bit en la PSW que indica el modo de ejecución***. El bit se cambia como respuesta a ciertos sucesos tales como una llamada al sistema.

5.2 Cambio de proceso

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
A primera vista, la función de cambio de proceso parece sencilla. En cierto momento, un proceso que se está ejecutando se interrumpe, el sistema operativo pone a otro proceso en el estado de ejecución y pasa el control a dicho proceso. Sin embargo, surgen diversas cuestiones de diseño. En primer lugar,

- **¿qué sucesos provocan un cambio de proceso?**.
- Otra cuestión es que se debe hacer una distinción entre **cambio de contexto** y **cambio de proceso**.
- Por último, **¿qué debe hacer el sistema operativo con las diferentes estructuras de datos bajo su control para llevar a cabo un cambio de proceso?**

5.2.1 ¿Qué eventos provocan el cambio de proceso?

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos

> Un **cambio de proceso** ***puede suceder (no necesariamente ocurre)*** en cualquier instante en el que el sistema operativo gana el control de la CPU por una interrupción, una excepción o una llamada al sistema.

En primer lugar, se van a tener en cuenta las **interrupciones del sistema**. Se pueden distinguir dos clases de interrupciones del sistema:

- La primera es originada por algún tipo de suceso que es **externo** e independiente del proceso que se está ejecutando, como la culminación de una E/S.
- La segunda tiene que ver con una **condición de error o excepción generada dentro del proceso** que se está ejecutando, como un intento ilegal de acceso a un fichero, una división entre cero, una instrucción máquina con código de operación no contemplado.

En una **interrupción ordinaria**, el control se transfiere primero al **gestor de interrupciones**, quien lleva a cabo algunas tareas básicas y, después, se salta a la rutina del sistema operativo que se ocupa del tipo de interrupción que se ha producido. Algunos ejemplos de estas interrupciones son:

- **Interrupción de reloj**: Un reloj es un dispositivo que genera interrupciones periódicamente. Ante una interrupción de este tipo, un sistema operativo de tiempo compartido, entre otras cosas, determina si el proceso en ejecución ha alcanzado el máximo tiempo de ejecución que se le concedió. Si es así, el proceso pasará a estado listo, y se asignará la CPU a otro proceso.
- **Interrupción de E/S**: El sistema operativo determina exactamente qué acción de E/S ha ocurrido. Si se trata de un evento por el que esperaban uno o más procesos, entonces el sistema operativo traslada todos los procesos bloqueados en dicho evento al estado listo, y determina

(dependiendo de la política de planificación, que se verá en el módulo de planificación) si retoma la ejecución del proceso interrumpido o pasa a otro de mayor prioridad.

- **Fallo de memoria**: Un proceso hace una referencia a una dirección que no se encuentra en memoria y que debe traerse de memoria secundaria (esta posibilidad se estudiará en el módulo de gestión de la memoria). Después de hacer la solicitud de E/S para traer esa o esas direcciones de memoria, el sistema operativo lleva a cabo un cambio de contexto (próximo apartado) para retomar la ejecución de otro proceso; el proceso que cometió el fallo de memoria se pasa al estado bloqueado. Después de que las direcciones aludidas se carguen en memoria, dicho proceso se pondrá en estado listo.

En una interrupción del segundo tipo, en una **excepción**, el sistema operativo determina si el error es fatal. Si lo es, el proceso que se estaba ejecutando es eliminado, y se produce un cambio de proceso. Si no es fatal, la acción del sistema operativo dependerá de la naturaleza del error y del diseño del sistema operativo. Se puede hacer un cambio de proceso o, simplemente, retomar el mismo proceso que se estaba ejecutando.

Finalmente, el sistema operativo puede activarse mediante una **llamada al sistema** desde el programa que se está ejecutando. Por ejemplo, está ejecutándose un proceso de usuario y se llega a una instrucción que solicita una operación de E/S, tal como abrir un fichero. Esta llamada provoca la transferencia a una rutina que forma parte del código del sistema operativo. Por lo general (aunque no siempre) el uso de una llamada al sistema hace que el proceso de usuario pase al estado bloqueado.

5.3 Cambio de contexto

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
Bloque Procesos: Gestión de Procesos
Si existe una interrupción pendiente es necesario:

- **Salvar el contexto** (PC, registros del procesador, información de la pila) del programa en ejecución.
- **Poner en el PC la dirección del programa de tratamiento de la interrupción**, que suele constar de unas pocas tareas básicas.

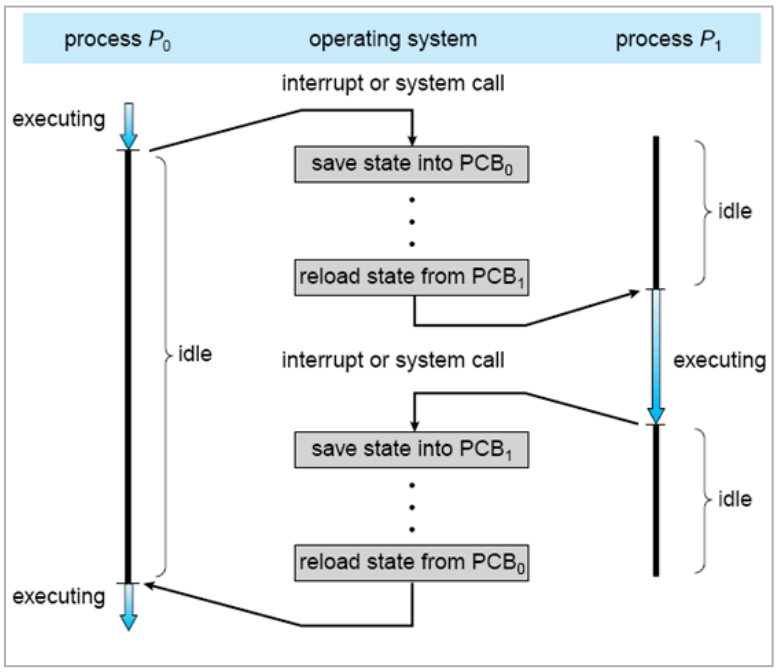


Figura 6. Cambio de contexto entre procesos.

Una pregunta que puede plantearse es: **¿qué es lo que constituye el contexto que se debe salvar?**

La respuesta es que se debe incluir información que pueda ser necesaria para reanudar el programa interrumpido. Así pues, debe guardarse la parte del bloque de control del proceso denominada información de estado del procesador. Esto incluye al contador de programa, otros registros del procesador y la información de la pila.

¿Se tiene que hacer algo más? Ello dependerá de lo que ocurra a continuación.

La rutina de tratamiento de la interrupción es normalmente un programa corto que lleva a cabo unas pocas tareas básicas relacionadas con una interrupción.

> Por ejemplo, se marca el indicador que señala la presencia de una interrupción, puede enviar un acuse de recibo a la entidad que produjo la interrupción (como un módulo de E/S) y puede hacer algunas tareas básicas relacionadas con los efectos del suceso que causó la interrupción.
> Por ejemplo, si la interrupción está relacionada con un suceso de E/S, el gestor de interrupciones comprobará condiciones de error. Si se ha producido un error, la rutina de tratamiento puede enviar una señal al proceso que solicitó originalmente la operación de E/S.

¿Hay que hacer algo más? Pues depende de si la interrupción va a venir seguida de un **cambio de proceso** o no. ***La ocurrencia de una interrupción, excepción o llamada al sistema NO siempre causa el cambio de proceso***. Es posible que después de que el gestor de interrupciones se haya ejecutado, el proceso que estaba ejecutándose retome su ejecución. En tal caso, tan sólo hay que guardar la información de estado del procesador y restaurarla para que pueda retomarse correctamente el proceso interrumpido (estas funciones son realizadas en *hardware*).

Por tanto, **el cambio de contexto es un concepto distinto al cambio de proceso**. Puede ocurrir **un cambio de contexto sin cambiar el estado del proceso** que está actualmente en estado de ejecución. En tal caso, salvar el contexto y restaurarlo posteriormente involucra un pequeño coste extra. Sin embargo, si el proceso que estaba ejecutándose tiene que pasar a otro estado (listo o bloqueado), el sistema operativo tiene que llevar a cabo cambios substanciales en su entorno.

Los pasos involucrados en **un cambio completo de proceso** son los siguientes:

1. **Salvar el contexto del procesador**, incluyendo el contador de programa y otros registros.
2. **Actualizar el PCB** que estaba en estado de ejecución. Esto implica **cambiar el estado del proceso** a alguno de los otros estados (listo, bloqueado, suspendido_listo). También se tienen que actualizar otros campos, como uno en el que se guarde la razón por la que se abandona el estado de en ejecución y otros con información de contabilidad.
3. **Mover el PCB a la cola apropiada** (listos, bloqueados por el suceso i, suspendido_listo).
4. **Seleccionar otro proceso** para su ejecución (como veremos en el módulo de planificación de procesos).
5. **Actualizar el PCB del proceso seleccionado**. Cambiar, por ejemplo, su estado a en ejecución.
6. **Actualizar las estructuras de datos de gestión de la memoria**. Esto puede hacer falta dependiendo de cómo se gestione la traducción de direcciones (lo dejaremos para los módulos sobre memoria).
7. **Restaurar el contexto del procesador** a aquél que existía en el momento en el que el proceso seleccionado dejó por última vez el estado de en ejecución, cargando los valores previos del contador de programa y de otros registros.

Así pues, **el cambio de proceso, que implica, al menos dos cambios de contexto**, requiere un esfuerzo considerablemente superior al de un cambio de contexto.