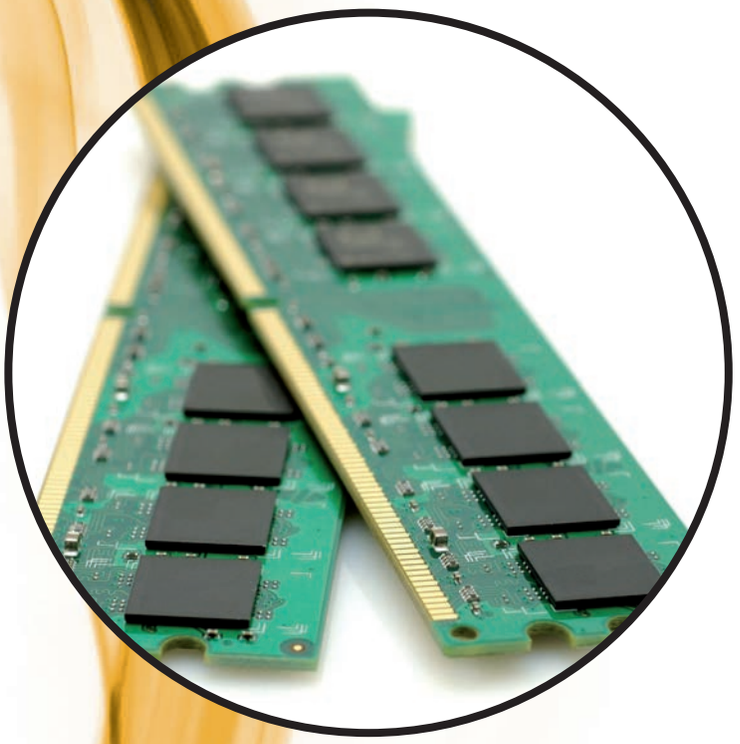
Unidad 3

Gestión de los recursos de un sistema operativo



En esta unidad aprenderemos a:

- Identificar los procesos y sus estados.
- Determinar las características y elementos de los procesos.
- Planificar la ejecución de procesos.
- Interpretar las técnicas de gestión de memoria.
- Diferenciar las técnicas de gestión de memoria.
- Conocer la gestión de entrada/salida del sistema operativo.

Y estudiaremos:

- Los procesos.
- La memoria RAM y su estructura.
- La forma de almacenar los procesos en memoria.
- Cómo se planifica la ejecución de procesos en el sistema informático.
- Los diferentes tipos de periféricos de un sistema informático.

1. Procesos y flujos

Vocabulario

Un **proceso** es un conjunto de instrucciones correspondientes a un programa que son ejecutadas por la UCP.

Un **proceso** es un concepto manejado por el sistema operativo y que referencia un programa en ejecución.

A los procesos, dependiendo especialmente del sistema operativo utilizado, se les denomina flujos de control, tareas, *threads* o hilos, según el contexto.

Cuando se ejecuta más de un proceso de forma concurrente en un sistema, todos necesitan que el propio sistema les suministre una serie de recursos. Para ello, el sistema operativo, gracias a la **UCP** (**U**nidad **C**entral de **P**roceso), se encarga de asignar estos recursos en un orden adecuado y atendiendo a unas prioridades. También realiza funciones de sincronización de todos los procesos, para que se ejecuten en el orden adecuado y según la prioridad decidida.

Cada vez que un programa se convierte en proceso, es decir, cada vez que se ejecuta un programa, además de ubicar en memoria las instrucciones que lo componen y sus datos asociados, a dicho proceso se le asocia una **estructura de datos.**

Esta estructura de datos, que es única para cada proceso, identifica el proceso respecto de los demás y sirve para controlar su correcta ejecución. Es lo que se llama el **bloque de control del proceso** o **BCP,** y contendrá para cada proceso la siguiente información: estado actual del proceso, identificador del proceso, prioridad del proceso, ubicación en memoria y recursos utilizados.

2. Hebras y estados de los procesos

Una **hebra** es un punto de ejecución de un proceso. Un proceso tendrá siempre una hebra, en la que corre el propio programa, pero puede tener más hebras.

Las hebras representan un método software para mejorar el rendimiento y eficacia de los sistemas operativos. Las hebras de un mismo proceso compartirán recursos, como memoria, archivos, recursos hardware, etc.

Vocabulario

Una **hebra** o **hilo** es un subproceso de un proceso que consume recursos propios pero que depende del proceso padre que lo ha ejecutado.

Un proceso clásico será aquel que solo posea una hebra. Pongamos un ejemplo. Si ejecutamos el procesador de textos Word, con un solo documento abierto, el programa Word convertido en proceso estará ejecutándose en un único espacio de memoria, tendrá acceso a determinados archivos (galerías de imágenes, corrector ortográfico, etc.), tendrá acceso al hardware (impresora, disquetera), etc. En definitiva, este proceso, de momento, solamente tiene una hebra.

Si en esta situación, sin cerrar Word, abrimos un nuevo documento, Word no se vuelve a cargar como proceso. Simplemente el programa, convertido en proceso, tendrá a su disposición dos hebras o hilos diferentes, de tal forma que el proceso sigue siendo el mismo (el original).

Word se está ejecutando una sola vez y el resto de documentos de texto que abramos en esta misma sesión de trabajo no serán procesos propiamente dichos. Serán hilos o hebras del proceso principal, que es el propio procesador de textos.

Antes de hablar de prioridades, y teniendo muy en cuenta lo comentado anteriormente, vamos a ver los diferentes estados en los que pueden estar los procesos.

Hoy en día existen gran cantidad de programas diseñados en multihilo o multihebra. De esta forma, si un programa puede realizar varias cosas, como analizar el registro del equipo, desfragmentar el disco duro y realizar copias de seguridad, todas ellas se podrán ejecutar a la vez. En programas convencionales, solamente se podría ejecutar una tras otra, pero no todas a la vez.

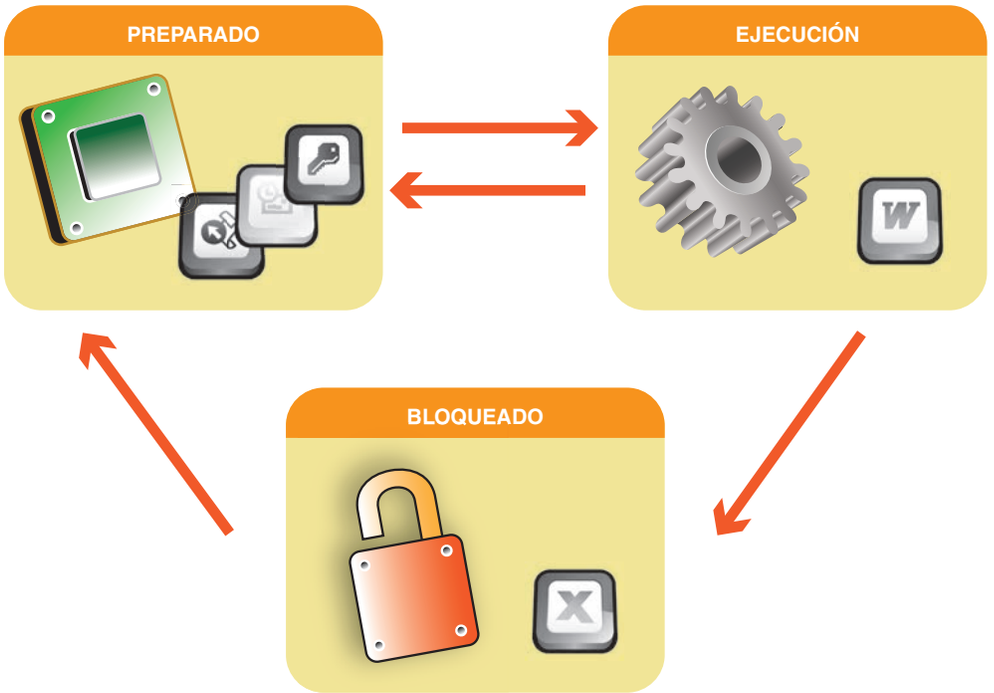


Fig. 3.1. *Estados de los procesos.*

Básicamente los estados posibles de un proceso, que podemos ver en la Figura 3.1, son los siguientes:

- **En ejecución.** El procesador está ejecutando instrucciones del programa que lo compone y tiene concedido el tiempo de uso de la UCP en un instante concreto.
- **Preparado, en espera o activo.** Un proceso está preparado para ser ejecutado; es decir, está esperando turno para poder utilizar su intervalo de tiempo y poner en funcionamiento sus instrucciones accediendo a los recursos del sistema.
- **Bloqueado.** El proceso está retenido; es decir, está bloqueado debido a causas múltiples. Una de estas causas puede ser que dos procesos utilicen el mismo fichero de datos. Otra puede ser que dos procesos necesiten utilizar la misma unidad de CD-ROM para cargar determinados datos, etc.

En general, todos los procesos dentro de cualquier sistema operativo tienen unas características que los identifican. En primer lugar, indicaremos que cada programa en ejecución, es decir, cada proceso, tiene un identificador que lo discrimina de los demás. Cada proceso tiene un número asignado por el sistema operativo que sirve precisamente para identificar el proceso, lanzarlo a ejecución, detenerlo, cancelarlo, reanudarlo, etc. Este identificador de proceso se nombra con la abreviatura **PID.**

También veremos que en cada sistema operativo, los procesos los lanzan normalmente otros procesos. Es decir, que cada proceso que se lanza a ejecución depende, en la mayoría de los casos, de otro proceso denominado **proceso padre.** Así, al nuevo proceso lanzado se le denomina **proceso hijo.**

Ten en cuenta

Un programa se convierte en proceso cuando se ejecuta y reside completamente en memoria RAM.

Actividades

1. ¿Puede pasar un programa de estado bloqueado a ejecución?
2. Cuando se lanza un nuevo proceso, ¿pasa este directamente a ejecución?

3. Transición de los procesos

Una vez que un programa se ha lanzado y se ha convertido en proceso, puede atravesar varias fases o **estados** hasta que finaliza o termina.

Cuando un proceso se lanza, nunca se ejecuta directamente, sino que se coloca en la **cola de procesos** en un estado denominado **preparado.** Cuando la UCP le asigna su tiempo, el proceso pasa de preparado a **ejecución.** Estos dos estados se alternarán en caso de que se esté ejecutando más de un proceso en el sistema.

Los cambios de estado en los que se puede encontrar un proceso se denominan **transiciones.** En la Figura 3.2 se recogen las transiciones o cambios de estado que pueden experimentar los procesos.

- **Transición A.** Ocurre cuando el programa que está en ejecución necesita algún elemento, señal, dato, etc., para continuar ejecutándose.
- **Transición B.** Ocurre cuando un programa o proceso ha utilizado el tiempo asignado por la UCP (procesador) para su ejecución y tiene que dejar paso al siguiente proceso.
- **Transición C.** Ocurre cuando el proceso que está preparado pasa al proceso de ejecución, es decir, cuando al proceso le llega una nueva disposición de tiempo de la UCP para poder ejecutarse.
- **Transición D.** Ocurre cuando el proceso pasa de estar bloqueado a estar preparado, es decir, cuando el proceso recibe una orden o señal que estaba esperando para pasar al estado de preparado y, posteriormente, tras la transición, a estado de ejecución.

En un sistema multiproceso o multihebra, cuando un proceso o hilo pasa de un estado a otro (por ejemplo, de espera a ejecución), lo que se producirá es un **cambio de contexto.**

El cambio de contexto puede ser **parcial** si se realiza entre hilos del mismo proceso. En caso de que el cambio de contexto sea entre hilos de diferentes procesos, se producirá un **cambio de contexto completo,** ya que el cambio afectará a memoria, hardware, ficheros comunes, etc.

Veamos en la Figura 3.3 un ejemplo de cambio de contexto entre dos procesos:

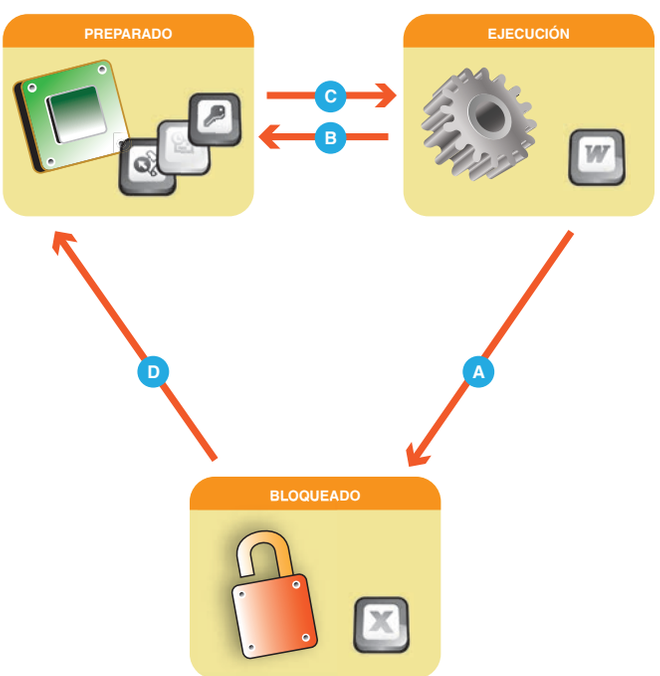


Fig. 3.2. *Transición de los procesos.*

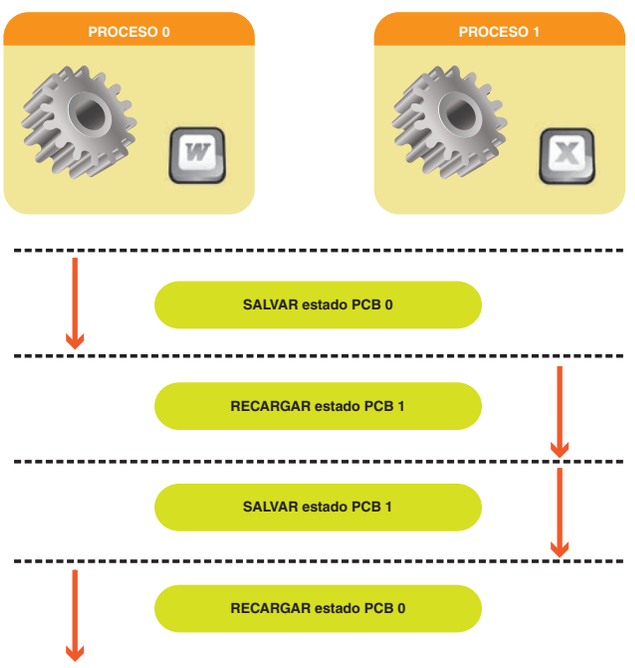


Fig. 3.3. *Cambio de contexto.*

Ejemplos

En la Figura 3.4 podemos apreciar de qué forma se ejecutan tres procesos (o hilos en sistemas operativos multihilo o multihebra), pasando de estar activos a estar en espera, según se asignen tiempos de ejecución de UCP a unos u otros.

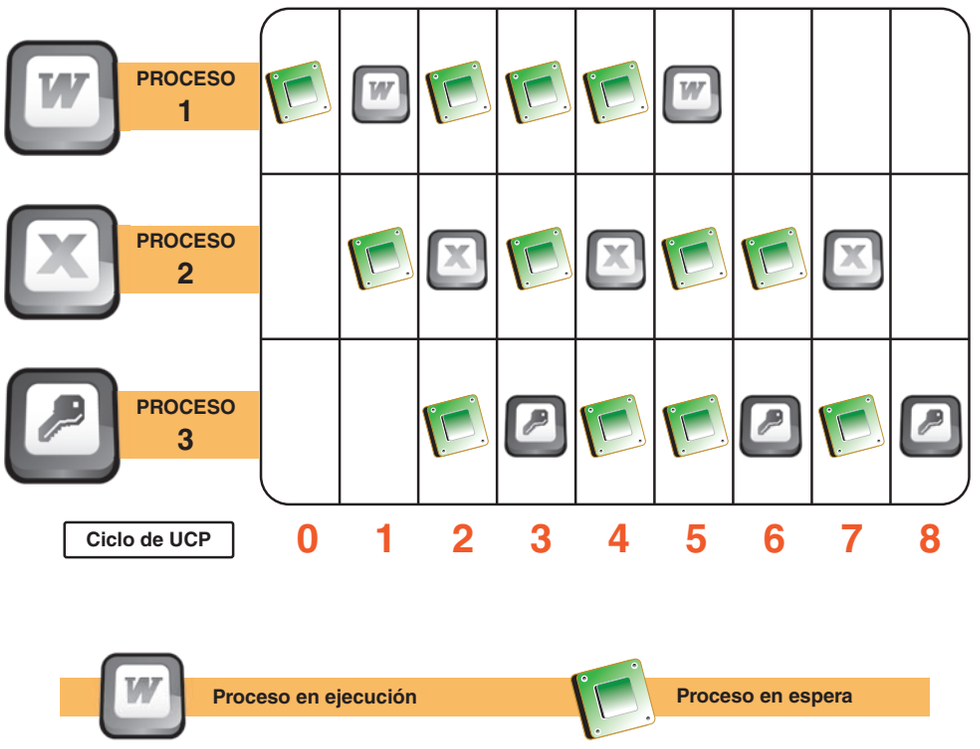


Fig. 3.4. *Esquema de ejecución de tres procesos.*

Los diferentes estados tienen una relación directa con lo que vamos a denominar **prioridades,** que son aquellas que el administrador del sistema, o el propio sistema, asignan a cada proceso. De ello dependerá que un proceso se ejecute en más o menos tiempo.

Se pueden establecer prioridades en función de la necesidad de ejecución de algunos programas. Los programas que más se ejecutan, es decir, los más necesarios, tendrán prioridad de ejecución sobre aquellos que se ejecutan muy de cuando en cuando.

Es ahora cuando hemos de hablar de la **planificación.** Con esta técnica conseguimos indicar al ordenador los procesos que deben ejecutarse y los estados que estos deben adoptar. Gracias a los **algoritmos de planificación** podemos decidir qué proceso ha de ejecutarse en cada momento y por qué. Algunas características de estos algoritmos son la imparcialidad, la equidad, la eficiencia, el tiempo de respuesta y el rendimiento.

Para ejecutar un proceso, introducimos en el indicador de comandos (***shell*** en UNIX, **command.com** en Windows) el nombre de un fichero ejecutable o hacemos doble clic con el ratón sobre el icono que representa un programa ejecutable (por ejemplo: **Explorer** en Windows). De esta forma, el sistema operativo prepara el programa a través del **cargador** para lanzarlo a ejecución.

Ten en cuenta

La prioridad de un proceso determina la cantidad de ciclos de UCP que consumirá respecto de otros procesos en ejecución.

Actividades

3. ¿Cuántas instrucciones puede procesar en un instante de tiempo la UCP?
4. ¿Quién asigna las prioridades a los procesos en un sistema multiproceso?

Vocabulario

Se denomina **planificador** a aquella parte del sistema operativo encargada de asignar los recursos del sistema, de manera que se consigan unos objetivos de comportamiento especificados.

Una vez cargado el proceso, el sistema operativo asigna a través del **planificador** la prioridad del nuevo proceso respecto de los que hay en ejecución.

De esta forma, cada proceso atraviesa varias fases. En un momento dado, el proceso se estará ejecutando; posteriormente estará en espera, mientras la UCP ejecuta otro; otros procesos estarán preparados para ser lanzados; otros podrán estar bloqueados, etc. Pues bien, en estos cambios de proceso, el sistema operativo tiene que saber qué ficheros están abiertos en cada proceso, qué periféricos se están utilizando, etc.

Cuando se están ejecutando varias tareas a la vez (procesos), es necesario compartir el tiempo de trabajo de la UCP. El tiempo compartido consiste en dividir el tiempo de ejecución del procesador en minúsculos intervalos de tiempo ***(quantum)*** e ir asignando cada uno de esos intervalos de ejecución a cada proceso que está en ejecución.

4. Bloque de Control de Procesos

Ten en cuenta

Todo proceso consume recursos hardware de un sistema informático, y es el sistema operativo el que determina, mediante el planificador, de qué forma se asignan los recursos a cada proceso.

Los sistemas operativos disponen de los servicios necesarios para la gestión de los procesos, tales como su creación, terminación, ejecución periódica, cambio de prioridad, etc. Además, durante su existencia, los procesos pasan por distintos estados cuyas transiciones están controladas por el sistema operativo. Los diferentes estados de los procesos y sus posibles transiciones ya los hemos visto antes.

Toda la información de un proceso que el sistema operativo necesita para controlarlo se mantiene en una estructura de datos vista anteriormente: el **bloque de control de procesos** o **BCP.** En sistemas operativos multiproceso, el sistema operativo mantiene listas de bloques de control de procesos para cada uno de los estados del sistema.

El BCP de cada proceso almacena información como:

- **Estado actual del proceso.** Ejecución, preparado o bloqueado.
- **Identificador del proceso.** Dependiendo del sistema operativo, a cada proceso se le asigna un PID.
- **Prioridad del proceso.** La asignada por el planificador.
- **Ubicación en memoria.** Dirección de memoria en la que se carga el proceso.
- **Recursos utilizados.** Recursos hardware y software para poder ejecutarse.

Ten en cuenta

A una posición de memoria solamente puede acceder un proceso en un determinado momento.

Gracias a los algoritmos de planificación, el cargador, planificador, BCP, recursos hardware y software se sincronizan mediante el sistema operativo para la ejecución ordenada de los procesos.

En programas multihilo o multihebra, el BCP puede contener además el PPID, o ***P**rocess **P**arent **ID**entification*. Este dato referencia el PID del proceso padre dentro del BCP, de tal forma que desde el propio BCP se pueden identificar todos los procesos que son hijos de otro, siempre y cuando tengan el mismo PPID.

Es evidente que en procesos convencionales, este dato en el BCP no existirá.

Actividades

5. ¿Desde dónde y cómo se pueden lanzar los procesos en los sistemas Windows y Linux?

6. En un sistema operativo monoproceso, ¿cómo se planifica la ejecución de varios procesos?

5. Algoritmos de planificación

Gracias a los algoritmos de planificación, especialmente en sistemas operativos multiproceso o en sistemas operativos en red, siempre y cuando se ejecuten varios procesos en el mismo equipo, la CPU se encarga de asignar tiempos de ejecución a cada proceso según el tipo de algoritmo y la prioridad de cada proceso.

A continuación vamos a dar una breve descripción de algunos de los algoritmos de planificación más habituales en sistemas multiproceso y multiusuario.

Veamos dos de los algoritmos de planificación, actualmente utilizados en sistemas Windows y Linux.

- **Algoritmo de rueda.** Asigna rotativamente tiempos de ejecución a los diferentes procesos. También se llama algoritmo de **Round-Robin** y en él la asignación de tiempos de ejecución a los procesos es la misma y de forma secuencial. A cada uno se le asigna el mismo ***quantum*** o intervalo de tiempo de ejecución. La selección entre los procesos se realiza mediante una cola **FIFO** (***F**irst **I**n **F**irst **O**ut,* el primero en entrar es el primero en salir o ser servido). Es el algoritmo utilizado normalmente en la asignación de tiempos en sistemas operativos multiusuario y multiproceso, y en la actualidad se puede decir que es el utilizado en sistemas operativos monousuario y que trabajan en multitarea.
- **Algoritmo FIFO** ***(F**irst **I**n **F**irst **O**ut)* **o FCFS** ***(F**irst **C**ome **F**irst **S**erve).* Los ciclos de UCP asignados a cada proceso se asignan en función de una cola FIFO. Al primer proceso que llega se le asignan tiempos o ciclos de UCP hasta que termina completamente. A continuación, se ejecuta completo el siguiente proceso que hay en la cola FIFO y así sucesivamente hasta terminar con el último proceso. Este algoritmo de planificación normalmente se utiliza para la gestión de trabajos en colas de impresión, respecto de los trabajos que van llegando a la impresora.

Actividades

7. ¿Cuál es el algoritmo utilizado normalmente para gestionar los trabajos que llegan a una impresora?

8. Cuando todos los procesos tienen más o menos la misma duración, ¿cuál es el algoritmo que mejor optimiza el tiempo de la UCP para varios procesos?

CEO

En la Web del CEO podemos encontrar una ampliación de este punto y, sobre todo, ejemplos de estos y otros algoritmos de planificación.

6. Memoria RAM y memoria virtual

Sabemos que el ordenador cuenta con la memoria central o principal, pero esta es limitada y, en grandes sistemas, insuficiente.

Al principio, para ubicar los procesos en memoria y solucionar este problema, se adoptaron técnicas tales como dividir el programa en partes denominadas **capas.** Cada una de las capas se iba ejecutando (cargando en memoria) según fuera necesario; es decir, primero se pasaría parte del programa del disco duro (o soporte de almacenamiento) a la memoria, y cuando fuera necesario utilizar otra parte del programa que no estuviese en memoria central o principal (RAM), se accedería de nuevo al disco para cargar la siguiente capa en memoria central.

Esta labor de dividir el programa en capas la puede realizar el mismo programador mediante la división del programa en módulos que se irán ejecutando según sea necesario, si bien esto supone un elevado esfuerzo para él.

Fotheringam diseñó un método conocido como de **memoria virtual.** Este diseñador pensó en la posibilidad de que al ubicar un programa en memoria, este fuera demasiado grande para el tamaño físico de aquella y creó una técnica para hacer que en memoria permaneciera solo la parte del programa que se estuviera ejecutando y que el resto quedara en el disco.

Ampliación

Fotheringam diseñó en 1961 una técnica revolucionaria para dividir los programas en partes y poderlos ubicar en equipos que tuvieran menos memoria real que el tamaño del programa.

Este concepto, aplicado hoy en día en la mayoría de los sistemas operativos, considera el espacio libre de disco como si se tratase de memoria RAM (memoria virtual). Así, para el usuario el programa estará cargado en RAM, pero en realidad solo se cargará en RAM la parte del programa que se esté ejecutando en ese instante. Entre tanto, el resto del programa en ejecución permanecerá temporalmente almacenado en disco para su posterior utilización, si fuera necesario.

Si en un momento dado necesitamos ejecutar una parte del programa almacenada en memoria virtual (en el disco duro), esta pasará a RAM para su ejecución real, y la parte del programa que estaba en RAM pasará al disco. Así, siempre habrá más RAM libre para realizar cálculos o ejecutar otros programas, sobre todo en sistemas operativos multiusuario y multitarea.

En la Figura 3.5 podemos ver los elementos que entran en juego a la hora de utilizar memoria virtual.

Para la ubicación de programas en memoria, se puede utilizar la técnica de memoria virtual para que siempre haya RAM libre para todos los programas que queramos ejecutar, es decir, para los procesos. Eso sí, cuando cargamos demasiados procesos a la vez, el sistema se ralentiza, ya que tiene que pasar información continuamente desde el disco duro a la RAM o viceversa.

Los sistemas operativos multiusuario y multitarea son especialistas en esta gestión. Casi todas las versiones de Windows realizan una gestión muy eficaz de la memoria virtual.

Es obvio que para realizar esta gestión se ha de disponer de un espacio determinado en el disco duro. Concretamente, para sistemas de Microsoft es recomendable asignar un 2,5 % del tamaño total de la RAM de espacio en disco para la gestión de memoria virtual, y un 5 % como máximo.

En sistemas operativos Windows, respecto de otros como Linux, existe un gran problema a la hora de gestionar la memoria virtual, y es la fragmentación de los archivos que se almacenan en la zona de intercambio. Esto en Linux no pasa. El que esta zona se fragmente, si es que se hace un uso considerable de esta zona de intercambio, implica que el equipo cada vez sea más lento ya que los archivos no están contiguos y eso implica que el acceso a ellos sea mucho más lento.

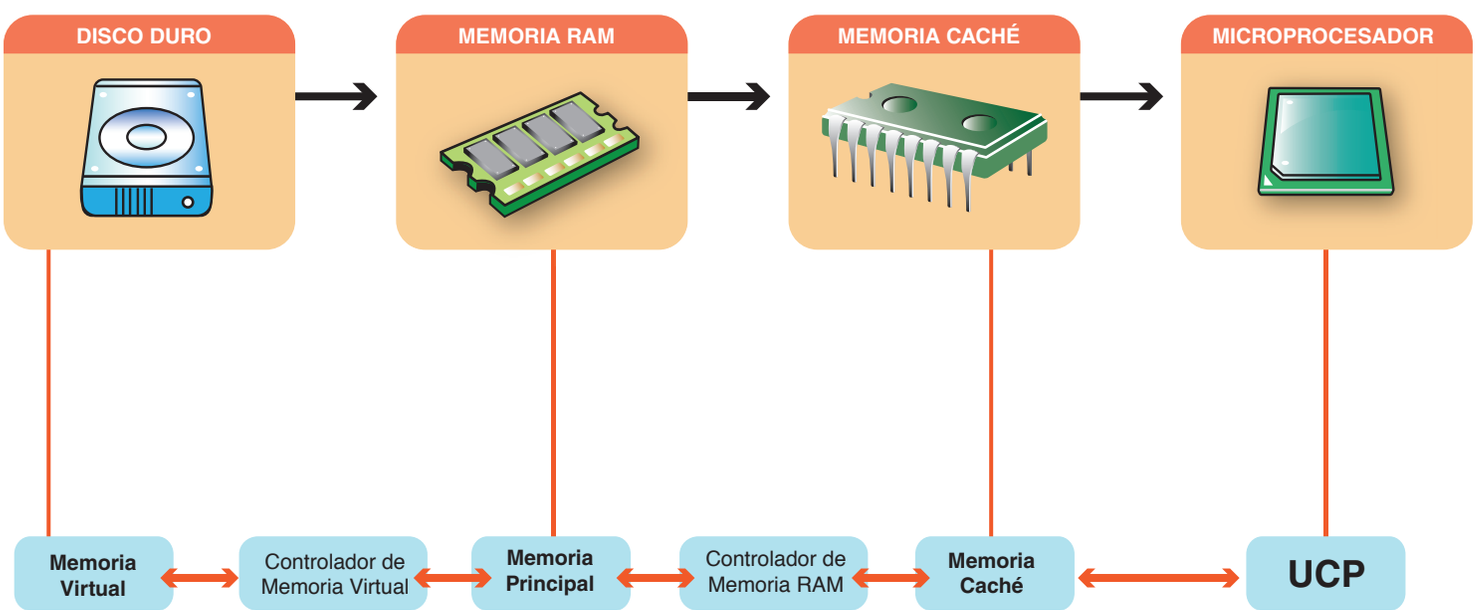


Fig. 3.5. *Gestión de memoria virtual.*

7. Intercambio

Al principio, en los sistemas operativos monousuario y monoproceso, la gestión de memoria era muy sencilla. Las memorias tenían poca capacidad y solo se reservaba una parte de ellas para el sistema operativo (véase Fig. 3.6).

Con la aparición de los sistemas operativos multiusuario y multitarea, la gestión de memoria se convierte en una de las funciones más importantes del sistema operativo.

La parte del sistema operativo que administra la memoria es el **administrador de memoria.** Su labor es clara: llevar un registro de las partes de memoria que se están utilizando y de las que no. De esta forma, reserva espacio de memoria para los nuevos procesos y libera el espacio de los procesos que finalizan.

El administrador de memoria también se encarga de gestionar el intercambio de datos entre memoria y disco, siempre y cuando los procesos sean tan grandes que no quepan de una sola vez en la memoria.

La gestión de memoria es importante cuando trabajamos en sistemas operativos multiproceso, y aún más en sistemas operativos multihilo, ya que se comparten espacios de memoria en los que se alojan las variables compartidas y a los que acceden varios procesos o hilos de un proceso.

En este caso, la memoria tiene que ser gestionada y controlada por el sistema operativo de tal forma que cada proceso utilice el espacio de memoria sin afectar a otros espacios de memoria en los que pueda haber datos o registros con información para otros procesos o hilos de un proceso.

Para gestionar la memoria en sistemas operativos multitarea, esta se divide en particiones fijas. Así, el sistema operativo dispone de una cola de los procesos que solicitan entrar en memoria. El **planificador** tiene en cuenta los requerimientos de memoria de cada uno de los procesos y las particiones de memoria disponibles. Estos requerimientos de uso de memoria se almacenan en el **BCP.**

La mayor dificultad de diseño de las particiones fijas es la adecuada selección de los tamaños de las mismas, puesto que puede derivar en un desaprovechamiento o fragmentación de la memoria. Esta fragmentación puede ser interna, cuando la parte de la memoria que no se está usando es interna a una partición asignada a un proceso, o externa, cuando una partición disponible no se emplea porque es muy pequeña para cualesquiera de los procesos que esperan.

Con un conjunto dinámico de procesos ejecutándose no es posible encontrar las particiones de memoria adecuadas. La opción es disponer de particiones variables. El problema que se plantea ahora es disponer de un registro con información de las particiones libres y ocupadas, que sea eficiente tanto en el tiempo de asignación como en el aprovechamiento de la memoria. No obstante, se siguen presentando problemas de fragmentación externa. Una solución es permitir que los procesos puedan utilizar memoria no contigua, lo que se consigue mediante técnicas de **paginación.** En esta situación hay un mecanismo de traducción de las direcciones lógicas a las físicas mediante una **tabla de páginas.**

La tabla de páginas presenta dos cuestiones a tener en cuenta: el tamaño de la tabla (que puede ser demasiado grande) y el tiempo de asignación (que debe ser de corta duración).

Vocabulario

La **zona de intercambio** es una zona de un disco duro utilizada para almacenar procesos que actualmente no están en ejecución y así dejar memoria RAM libre para los procesos que sí lo están.

Actividades

9. ¿En qué sistemas es más importante la gestión de memoria?
10. ¿Con qué tipo de soportes se realiza el intercambio de memoria de un ordenador?

Fig. 3.6. *Gestión básica de memoria.*

Ten en cuenta

En la **segmentación,** la RAM se divide en espacios que no tienen que ser del mismo tamaño y que estarán en función del tamaño de los procesos que se vayan a ejecutar.

En contraposición a la visión de la memoria como un *array* o lista unidimensional, está la concepción por parte del usuario de considerar la memoria como un conjunto de segmentos de diferentes tamaños, sin ninguna ordenación entre ellos. Este esquema corresponde a la **segmentación.** En este caso, el espacio de direcciones lógicas es un conjunto de segmentos con diferentes nombres y tamaños. En el esquema de segmentación no se produce fragmentación interna, pero sí externa, que ocurre cuando todos los bloques de memoria libres son muy pequeños para acomodar un trozo o bloque de proceso.

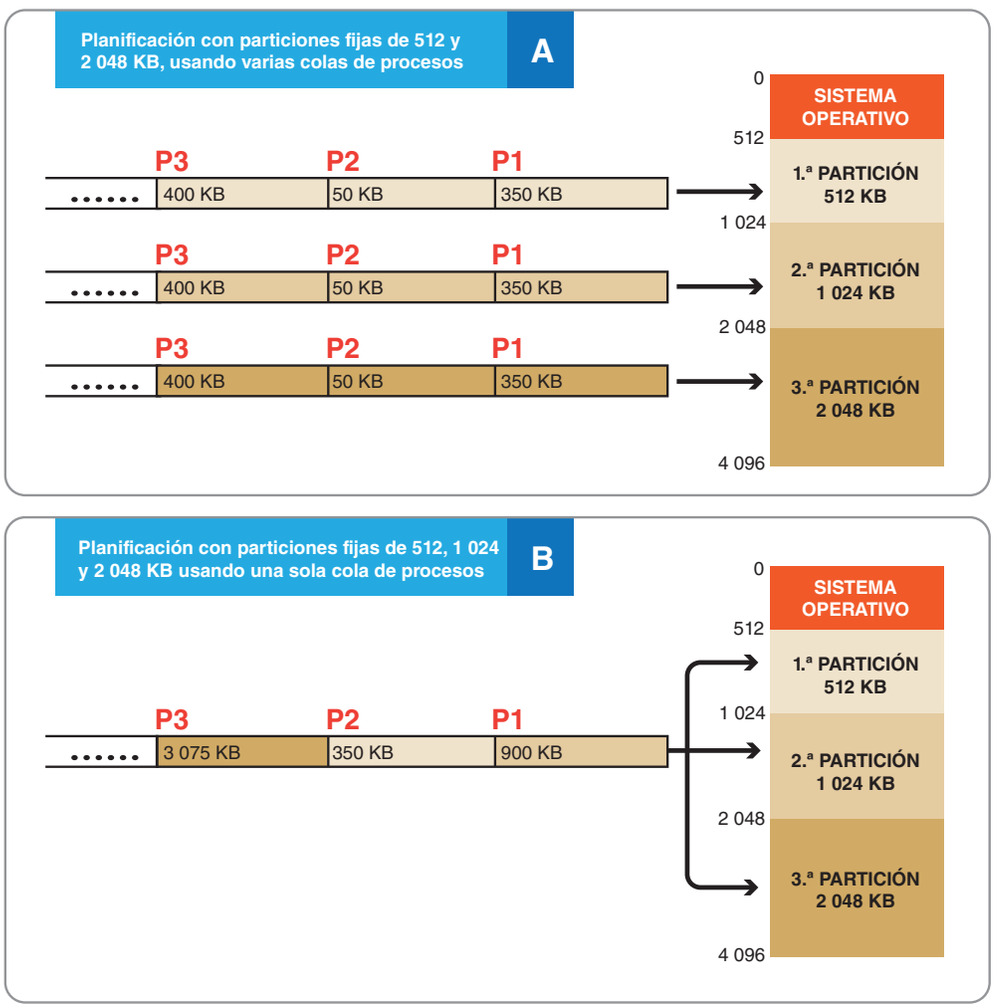
Aunque la segmentación y la paginación son esquemas diferentes de gestión de la memoria, se pueden considerar estrategias combinadas, ya que la única diferencia es que la paginación utiliza bloques de memoria de tamaño fijo.

En todos estos esquemas, se supone que el proceso que se va a ejecutar está cargado totalmente en memoria. La idea de permitir ejecutar procesos que no están cargados totalmente en memoria, e incluso que sus tamaños superen al de la memoria física instalada, da lugar al concepto de **memoria virtual.**

En los sistemas operativos actuales se puede configurar el área de intercambio, de tal forma que podemos indicar el tamaño en disco destinado a tal fin, e incluso se puede indicar en qué disco se puede realizar el intercambio.

El gran inconveniente de modificar «a la ligera» lo que el sistema administra automáticamente, es que podemos provocar que el rendimiento del equipo sea menor.

Ejemplos



En la Figura 3.7 vemos un ejemplo con dos formas de planificar la ubicación de los procesos que esperan para ser ejecutados.

La primera técnica («A» en el gráfico) consiste en dividir la memoria RAM, que tiene 4 MB de capacidad total, en zonas de tamaño fijo pero no iguales, en nuestro caso de 512, 1 024 y 2 048 KB, de tal forma que se puedan crear varias colas de procesos cuyos tamaños sean inferiores al tamaño de la partición en la que esperan ejecutarse.

La segunda técnica («B» en el gráfico) consiste en mantener esas divisiones de memoria con tamaños fijos y diferentes, pero ahora solo habrá una cola de espera para los procesos, que se irán alojando en una de las particiones libres que sea capaz de alojarlos.

En ambos casos existe una zona reservada, como siempre, para que se aloje el sistema operativo. Es la zona de memoria más baja que corresponde a los primeros 512 KB de los 4 MB (4 094 KB) totales de memoria RAM de los que se compone nuestro ejemplo.

Fig. 3.7. *Gestión de particiones.*

8. Paginación, segmentación y *swapping*

Segmentación, paginación y *swapping* son técnicas de gestión de memoria, que en general permiten ejecutar programas de un tamaño superior a la capacidad de la memoria RAM utilizando el disco duro como una «ampliación» de la memoria principal del equipo. La ventaja es que se puede ejecutar cualquier programa; el inconveniente es la pérdida de rendimiento.

8.1. Paginación

La paginación es una técnica que consiste en dividir la memoria interna o RAM en zonas iguales llamadas ***frames,*** y los programas en partes del mismo tamaño denominadas **páginas.**

Para ubicar un programa en memoria, el sistema operativo buscará en memoria física los *frames* que tenga libres. El tamaño de estos *frames* se diseña mediante hardware.

Si utilizamos un sistema de multiprogramación y solo hay un trabajo, este tendrá asignados todos los *frames* necesarios para él. Esta asignación de *frames* la realiza el sistema operativo.

Mediante la **tabla de páginas,** la UCP asigna las direcciones físicas de los *frames* a las páginas en las que se ha dividido el programa. La asignación de los *frames* no tiene que ser necesariamente consecutiva. Un proceso se puede ubicar en memoria interna en *frames* no contiguos, ya que estos pueden estar ocupados por otros procesos.

Vocabulario

Un **marco de página** es una división de la memoria en zonas del mismo tamaño utilizadas para intercambiar procesos con los espacios de almacenamiento.

La técnica de paginación es similar a la de memoria virtual. La gran diferencia es que aquí no existe disco duro para intercambiar parte de los procesos. Concretamente, el sistema operativo DOS utiliza una técnica parecida a la paginación.

Como ejemplo, veamos el sistema operativo DOS. Solo sirve de almacenamiento para parte del núcleo del sistema operativo y para almacenar temporalmente parte de los procesos que tengan un tamaño superior a 640 KB.

DOS divide la memoria extendida (por encima del primer MB) en páginas de 64 KB para realizar el intercambio de información con la memoria convencional.

Un programa de 1 MB ocupará lo que pueda de memoria convencional y el resto se almacenará temporalmente en memoria extendida. Este programa se paginará a través del llamado **marco de página,** del que hablaremos más adelante. Se intercambian las páginas desde memoria convencional a extendida y viceversa, dependiendo de la parte del proceso que se vaya a ejecutar. Esta gestión de memoria se conoce como **memoria expandida.**

En resumen, la paginación es una técnica de reasignación o redireccionamiento dinámico, con la consideración de que la tabla de páginas se puede almacenar en registros especiales destinados a tal efecto o en una parte de la propia memoria.

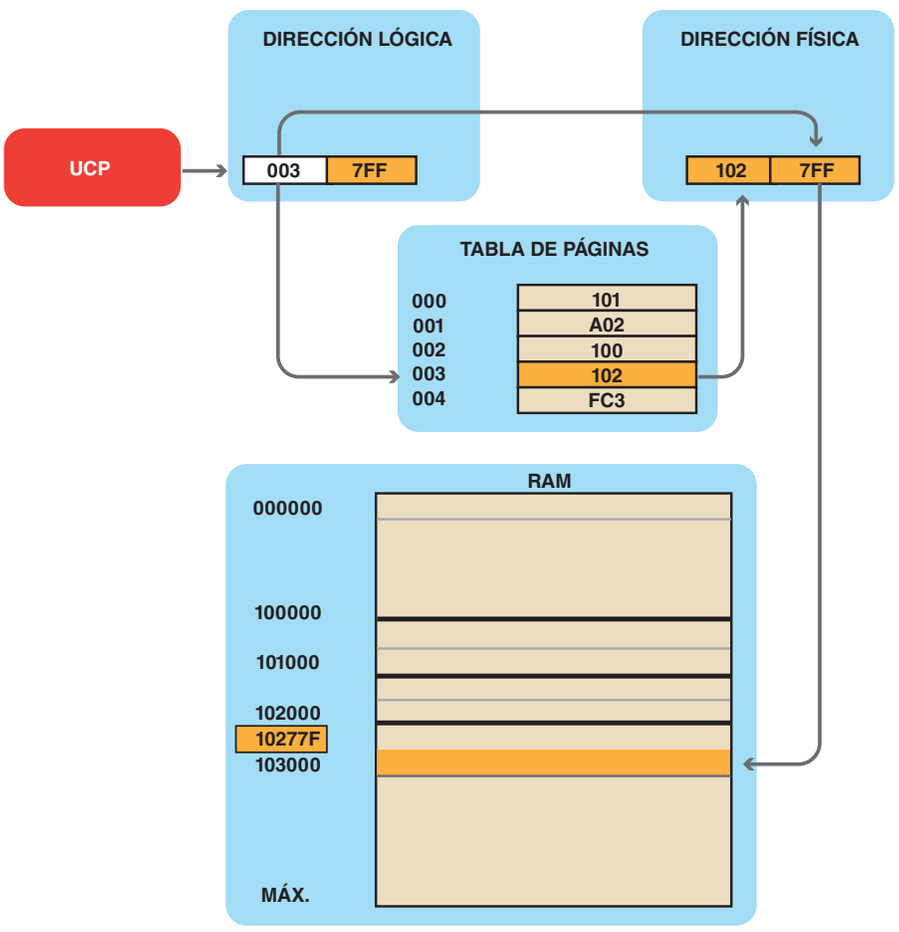


Fig. 3.8. *Paginación.*

Ten en cuenta

Todo proceso consume recursos hardware de un sistema informático, y es el sistema operativo el que determina, mediante el planificador, de qué forma se asignan los recursos a cada proceso.

La transformación de las direcciones lógicas en físicas la realiza la **unidad de administración de memoria** o ***M**anagement **M**emory **U**nit* (MMU).

Ejemplos

Supongamos que tenemos un proceso que necesita dividirse en cinco páginas: 000, 001, 002, 003 y 004. A cada página habrá que asignarle un *frame* o marco de página de la memoria física. Para hacer esa asignación se usa la tabla de páginas, que se construye cuando se carga el proceso y que contiene tantas entradas como páginas tenga el proceso, en nuestro caso cinco.

Cada dirección lógica tendrá dos campos, el primero accede a la página dentro de la tabla de páginas y el segundo indica el desplazamiento que hay que realizar dentro de esa página para acceder a la información deseada.

En nuestro ejemplo (véase Fig. 3.8), la UCP indica como dirección lógica 003 7FF. Esto quiere decir que debemos acceder a la cuarta página (003), donde está la dirección del marco de página o *frame* de la memoria física que contiene la información a ejecutar, que en nuestro caso es la 102.

Una vez ubicados en dicho *frame,* debemos desplazarnos 7FF posiciones, con lo que se obtiene a partir de los dos campos la dirección física a la que se quiere acceder, que en este caso es 102 7FF.

Truco

Nunca ejecutes demasiados programas a la vez con una cantidad de memoria pequeña, ya que el intercambio entre RAM y disco ralentizará mucho el sistema.

8.2. Segmentación

Es una técnica similar a la paginación que permite definir los bloques de memoria de tamaño variable. Cada **segmento** puede variar desde 0 hasta un máximo permitido.

Estos segmentos pueden tener longitudes distintas. Además, la longitud de un segmento puede variar según las necesidades del programa.

Supongamos que realizamos un programa y, para que se ejecute, necesita utilizar tablas (estructuras de datos) en memoria. Si tenemos en cuenta que una tabla puede asignarse de forma estática o dinámica según las necesidades del programa, habrá veces en que esta tabla necesitará un espacio determinado en memoria, mientras que otras, este espacio será mayor o menor según la necesidad. Gracias a la segmentación podemos ubicar en memoria estas estructuras de datos, independientemente del tamaño que tengan.

El ordenador, a través del sistema operativo, puede organizar la memoria en bloques concretos y tener partes de ella destinadas a almacenar las estructuras de datos, que pueden crecer o menguar según las necesidades del usuario o del programa. Para ello se utilizarán las pilas de memoria o ***stacks,*** en las que se gestionan las estructuras de datos necesarias.

La paginación difiere de la segmentación en que las páginas son de tamaño fijo y los segmentos no. El uso de la técnica de paginación o segmentación dependerá del sistema operativo utilizado y de la máquina en la que lo usemos, además de las necesidades del software.

8.3. *Swapping*

El *swapping* es una técnica similar a la de memoria virtual. Cuando varios usuarios están ejecutando procesos en un mismo ordenador, este se ve obligado a cargarlos en RAM. Según el estado en el que se encuentre el proceso de cada usuario, la memoria se irá liberando de su proceso y pasará a la zona de ***swap*** mediante la técnica llamada ***swap-out.*** De esta forma, la memoria interna queda liberada para que en ella se pueda almacenar otro proceso del mismo usuario o de otro.

Si el usuario vuelve a solicitar su proceso para seguir ejecutándolo, se produce el denominado *swap-in,* que consiste en pasar el programa de la zona de *swap* a la memoria interna. Esta zona de *swap* se suele utilizar en sistemas operativos como UNIX y Linux. Está formada por un espacio físico del disco en el que tenemos el sistema operativo y las aplicaciones que se van a ejecutar. Los fabricantes de estos sistemas operativos recomiendan que esta zona sea del 20% aproximadamente del espacio en disco o el doble de la capacidad de RAM del ordenador.

La diferencia entre la gestión de memoria virtual y el *swapping* es que, mediante la primera, puede llegar a ocurrir que el disco esté tan lleno que la gestión sea difícil o imposible, ya que el espacio destinado al intercambio suele ser espacio del disco duro en el que está instalado tanto el sistema operativo como el software de aplicaciones y los datos del usuario.

En el *swapping* no puede ocurrir esto, ya que esta zona siempre estará reservada y disponible para el intercambio de programas con la memoria principal. Normalmente, al estar esta zona en un dispositivo físico diferente, todo el espacio estará disponible cada vez que encendamos el ordenador.

Actividades

11. ¿Qué sistemas operativos del mercado utilizan la técnica de paginación para la ubicación de los procesos en memoria?

12. ¿Qué sistemas operativos del mercado utilizan la técnica de ***swapping*** para la ubicación de los procesos en memoria?

13. ¿Qué técnica es más efectiva, la paginación, la segmentación o el ***swapping?***

9. Programas reubicables, reentrantes, residentes y reutilizables

Según cómo, dónde y cuándo se ubiquen en memoria, los programas pueden ser de varios tipos:

A. Reubicables

Son aquellos que, una vez cargados en RAM para ejecutarse, pueden variar de situación, ya que la parte de RAM que ocupan puede ser necesaria para ubicar otro proceso. Estos procesos o programas cambian de posición cuando se está realizando una operación sobre el ordenador. Esta operación suele ser de configuración interna del propio ordenador.

Recuerda

Un programa es un conjunto de instrucciones que el sistema operativo ejecuta para realizar determinados procesos dentro de un sistema informático.

B. Reentrantes

Son aquellos programas que, si no se están ejecutando, dejan la memoria libre para otros procesos. Estos procesos, cuando se liberan, se suelen almacenar temporalmente en el disco duro. Son los procesos gestionados mediante la técnica de memoria virtual.

Recuerda

En un programa se ejecutan instrucciones aritméticas y lógicas, estas últimas determinadas por el álgebra de **Boole.**

C. Residentes

Son aquellos que, una vez cargados en memoria, permanecerán en ella hasta que se apague el ordenador. No cambian su ubicación en ningún momento. Suelen ser programas de antivirus, de análisis de sistema, de monitorización, etc. Los más comunes son los llamados **centinelas,** que incorporan los antivirus para que analicen continuamente lo que se carga en memoria. De esta forma, si se ejecuta un proceso, el programa residente lo analiza y, si detecta algo raro o extraño, envía un mensaje de alerta.

La ubicación de estos programas en memoria dependerá, fundamentalmente, del sistema operativo y de la propia aplicación que lance el programa residente. Suelen ubicarse en esos 64 KB de memoria, aunque no necesariamente.

Actividades

14. ¿Es el procesador de textos Word un programa residente?

15. ¿Es un antivirus un programa residente?

D. Reutilizables

Son programas que normalmente son utilizados por varios usuarios a la vez en memoria, independientemente del número de usuarios que los vayan a utilizar. Con ello se consigue un mejor aprovechamiento de la memoria.

10. Gestión de entrada/salida: tipos de periféricos

Una de las funciones principales de un sistema operativo es el control de los periféricos de entrada/salida del ordenador. El sistema operativo se encarga de enviar órdenes, determinar el dispositivo que necesita la atención del procesador, eliminar posibles errores, etc.

En primer lugar, es necesario hacer una clasificación de los periféricos. Esta clasificación no corresponde a si son periféricos de entrada o de salida, sino a si gestionan la información por bloques o por caracteres:

- **Periféricos tipo bloque.** Son aquellos en los que la información que se maneja es de tamaño fijo. La información entra o sale de memoria en forma de bloque. Un ejemplo son los registros de ficheros de datos almacenados en discos o disquetes, ya que cada registro contiene información referente a un bloque homogéneo.
- **Periféricos tipo carácter.** Son los que sirven para introducir datos dentro de la memoria del ordenador en forma de caracteres, sin ningún orden concreto, por ejemplo los teclados. También analizaremos los periféricos que sirven para ver los resultados obtenidos de nuestra gestión en forma de cadena de caracteres: pueden ser el monitor, la impresora, etc.

Cada periférico está compuesto por un componente mecánico y por otro, u otros, componentes electrónicos. Por ejemplo, un disco duro estará compuesto por los propios discos de aluminio recubiertos de material magnético, las cabezas de lectura, el motor que los hace girar, etc., y por la denominada controladora o adaptador, encargado de conectar el dispositivo físico al ordenador.

El sistema operativo se encarga de acceder a la información de la memoria principal, extraerla en forma de impulsos eléctricos y enviarla a los diferentes dispositivos periféricos. Si la información se envía a un disco duro, los impulsos se transformarán en señales de tipo magnético; si se envía a una impresora, se transformarán en caracteres, etc.

Ten en cuenta

Los sistemas operativos actuales trabajan normalmente mediante **interfaz de tipo gráfico,** aunque permiten la ejecución de determinados comandos y programas mediante la interfaz de tipo texto.

11. Comunicación con el sistema: interfaces de usuario

Hay que destacar las interfaces como medio de comunicación entre hardware y software a través del sistema operativo. Las interfaces se pueden clasificar en:

- **Interfaz tipo texto.** Si el sistema operativo es de tipo texto, todas las órdenes que el usuario introduzca y las respuestas que el sistema operativo dé se introducirán o visualizarán mediante cadenas de caracteres. Un ejemplo de sistemas operativos tipo texto son: DOS, UNIX (en versiones inferiores a la System V Release 4), las primeras versiones de Linux, etc. Todas las órdenes se introducen por teclado y se visualizan en la pantalla. La pantalla, cuando se gestiona en tipo texto, tiene un tamaño de 80 columnas por 24 filas; es decir, puede mostrar hasta 1 920 caracteres de una sola vez.
- **Interfaz tipo gráfico.** Hoy en día, la mayoría de los sistemas operativos utilizan medios de comunicación entre máquina y ordenador de tipo gráfico. En este tipo de interfaces, el uso del ratón es casi imprescindible. La información en pantalla se muestra en bloques o en pantallas independientes. A estos bloques se les denomina **ventanas,** y en ellas aparece una serie de componentes y objetos que nos sirven para enviar o recibir información sin tener que teclear nada.

Actividades

16. ¿Disponen todos los sistemas operativos de interfaz tipo texto y tipo gráfico?

17. ¿Crees que existe algún tipo de sistema operativo que no tenga interfaz gráfica?

12. Clasificación de los periféricos

La clasificación más usual de los periféricos es la que se muestra en la Figura 3.9:

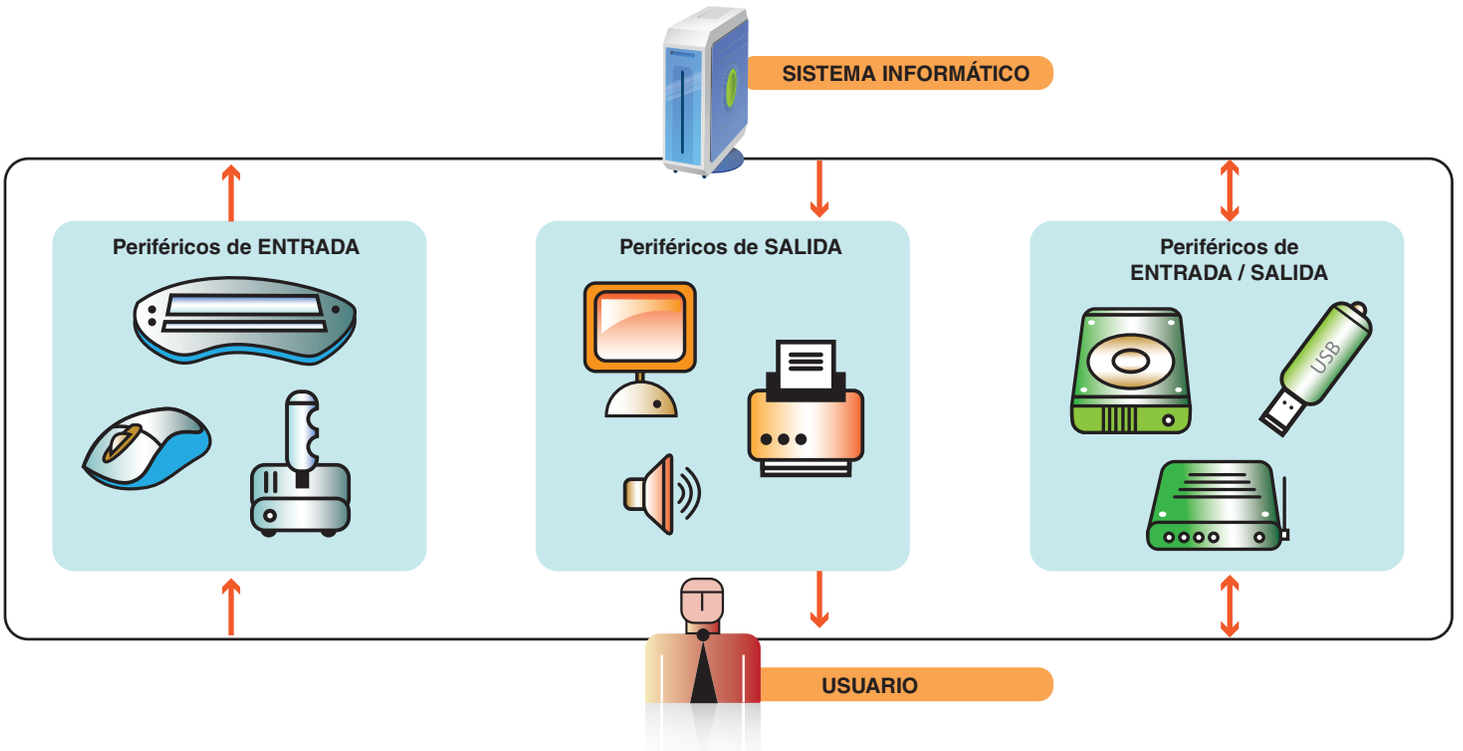


Fig. 3.9. *Esquema de periféricos de entrada/salida.*

- **De entrada.** Son los que sirven para introducir información (datos o programas) en el ordenador. La información va desde ellos hacia la memoria y el resto de componentes internos para ser procesada. Son periféricos de entrada el teclado, el escáner, la unidad lectora de CD-ROM, el ratón, etc.
- **De salida.** Son los que se utilizan para extraer la información (datos en forma de resultados, programas, etc.) desde la memoria y resto de componentes internos del ordenador, y mostrar los datos. Son periféricos de salida la impresora, la pantalla, el plóter, etc.
- **De entrada/salida (E/S).** Son los que se utilizan para introducir o extraer datos desde y hacia el ordenador. Por ejemplo, los dispositivos de almacenamiento como los discos duros (Fig. 3.11). En ellos se puede escribir información (salida) al igual que leerla (entrada). Hay otros muchos periféricos dentro de esta categoría, como los monitores táctiles, el módem, el *router,* las tarjetas de red, el *pen drive,* las impresoras multifunción, etc.

No se deben confundir los periféricos de E/S con los soportes de información. Los periféricos son, por ejemplo, las unidades de disquete. El disquete en sí se denomina **soporte,** ya que es el que almacena la información. El periférico no almacena información, pues es el medio físico que sirve para almacenarla. Pongamos un ejemplo: un radiocasete es un periférico, y la cinta en la que están grabadas las canciones es un soporte.

En definitiva, el soporte de información es la parte del periférico extraíble (disquete, CD-ROM) o no (platos del disco duro) en la que se almacena la información.

Alguna de las principales características de los soportes es que son reutilizables, que tienen elevada capacidad de almacenamiento, que son no volátiles y que son más económicos que la memoria principal (RAM).

Recuerda

Para que un mismo periférico de entrada/salida pueda funcionar en diferentes sistemas operativos, es necesario instalar los controladores o *drivers* que el fabricante suministra con el periférico.

Algunos de los periféricos de E/S más importantes son:

A. Teclado y ratón

Son los periféricos de entrada por excelencia. Los teclados pueden ser de varios modelos, dependiendo del número de teclas que lo compongan (84, 102 o 104). Normalmente, se utilizan los de 102 teclas. Veamos detalladamente cómo es un teclado en la Figura 3.10.

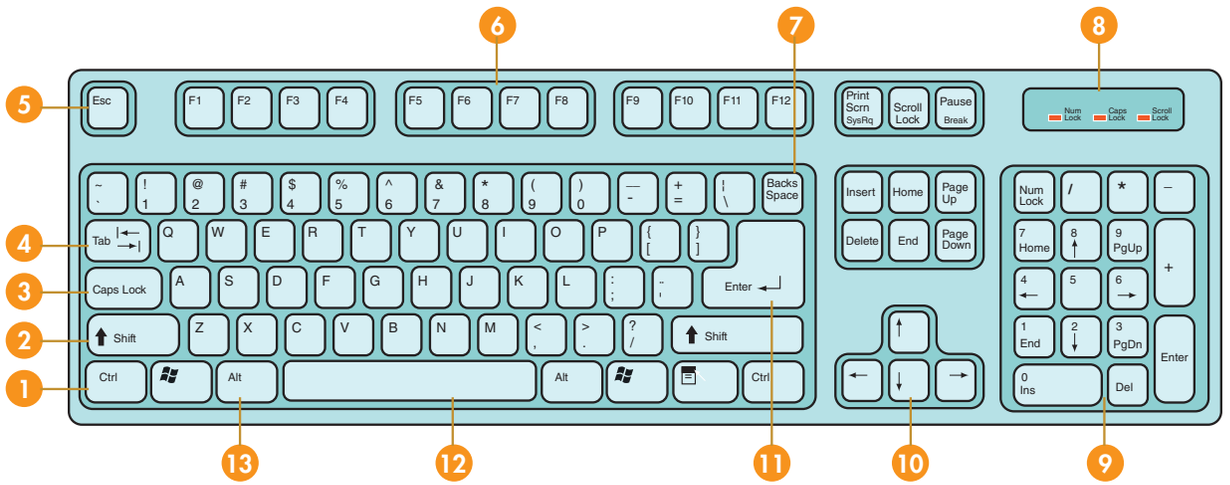


Fig. 3.10. *Esquema de un teclado.*

1. **Ctrl.** Se utiliza con otra combinación de teclas para realizar determinadas funciones o para seleccionar múltiples archivos y/o carpetas.
2. **Shift.** Se utiliza para escribir letras en mayúsculas o el símbolo de la parte superior del resto de teclas.
3. **Caps Lock.** Se utiliza para dejar activada la escritura de teclas en mayúsculas.
4. **Tab.** Se utiliza para tabulación en procesadores de texto y movimiento en campos de entrada en formularios.
5. **Esc.** Se utiliza normalmente para finalizar procesos o acciones.
6. **Teclas de función (F1 a F12).** Se utilizan para abreviar acciones en herramientas ofimáticas o realizar determinadas acciones sobre el sistema operativo.
7. **Retroceso.** Se utiliza para borrar el carácter a la izquierda de la posición del cursor.
8. **Panel identificador.** Indica si tenemos activadas las mayúsculas o el teclado numérico.
9. **Teclado numérico.** Se utiliza como tal o como teclado de edición en teclados que no disponen de teclas para este fin.
10. **Teclado de edición.** Se utiliza para moverse por documentos, por gráficos e incluso en los juegos.
11. **Enter.** Tecla que sirve para hacer efectivas las operaciones de confirmar alguna acción o para insertar líneas en procesadores de textos u otras herramientas ofimáticas.
12. **Espaciador.** Se utiliza para insertar espacios en blanco o seleccionar casillas de verificación en cuadros de diálogo.
13. **Alt.** Tecla que utilizada en combinación con otras sirve para realizar determinadas acciones del sistema operativo.

Investigación

Analiza en la Web los tipos de monitores de entrada/salida más comercializados y averigua en qué sistemas operativos se pueden utilizar.

B. Monitor

Es un periférico de salida. Puede ser monocromo o color, y sus prestaciones dependerán, en gran medida, de la **tarjeta gráfica** y de la memoria apropiada que incorpore el fabricante, de la frecuencia de refresco, del tamaño en pulgadas, etc. Estas tarjetas son las que comunican el ordenador con el monitor.

C. Impresora

Es un periférico de salida que permite la salida en papel de la información deseada. La gama de impresoras va desde las de *impacto* hasta las más modernas denominadas *sin impacto* (térmicas, de inyección de tinta, láser y electromagnéticas).

D. Otros periféricos

- **Escáner.** Es un dispositivo de entrada que permite transformar imágenes o texto impreso en datos digitales.
- **Módem.** Es un periférico de E/S que se conecta a la entrada estándar del teléfono y permite la comunicación remota con otros equipos.
- **Unidades de disquete.** Son periféricos de E/S que permiten almacenar o extraer información de los soportes (disquetes).
- **Unidades de disco duro.** Son de elevada capacidad y alta velocidad. Se utilizan para instalar en ellas el software de los sistemas operativos y la mayor parte del software de aplicaciones. Su capacidad se mide en GB.
- **Tableta digitalizadora y lápiz óptico.** Son periféricos utilizados normalmente para la confección de gráficos y esquemas en los que el uso del teclado y el ratón resulta tedioso. Ambos son dispositivos periféricos de entrada.
- **DVD** *(**D**igital **V**ideo **D**isk).* Es un periférico de entrada. Para acceder a la información se aplica tecnología láser. Su capacidad es superior a los 4 GB y goza de gran difusión en la actualidad.
- ***Blue-ray.*** Similar a los anteriores pero con mucha más capacidad. Suele almacenar hasta 50 GB de información.
- **HDVD.** Similar al DVD, tiene una capacidad de hasta 30 GB por soporte.

Actividades

18. ¿Un ***pen drive*** es un dispositivo de entrada, de salida o de entrada/salida?

19. ¿Los teclados son dispositivos de entrada?

Fig. 3.11. *Unidades lectoras y disco duro.*

13. Gestión de la información

Cuando trabajamos con sistemas operativos multiusuario, la gestión de datos que se hace dentro del ordenador y su ubicación en memoria y en los soportes de almacenamiento externo, pueden plantear algunos problemas.

Ya hemos visto que, para la ubicación en memoria, el sistema operativo dispone de sus medios. En cuanto al almacenamiento en soportes externos, la gestión que haga el sistema operativo tiene que responder a varias características: se podrá almacenar una gran cantidad de información, se almacenará de forma correcta una vez terminado el procesamiento y existirá la posibilidad de que varios procesos o programas accedan a la misma información sin interferencias.

Para todo esto, después de ser procesada, la información tiene que almacenarse de forma permanente en los soportes externos de almacenamiento a través de archivos. Cada sistema operativo utiliza su propio **sistema de archivos.** El sistema operativo gestiona cada archivo almacenado en el soporte indicando el nombre, el tamaño, el tipo, la fecha y hora de grabación, el lugar del soporte en el que se encuentra, etc.

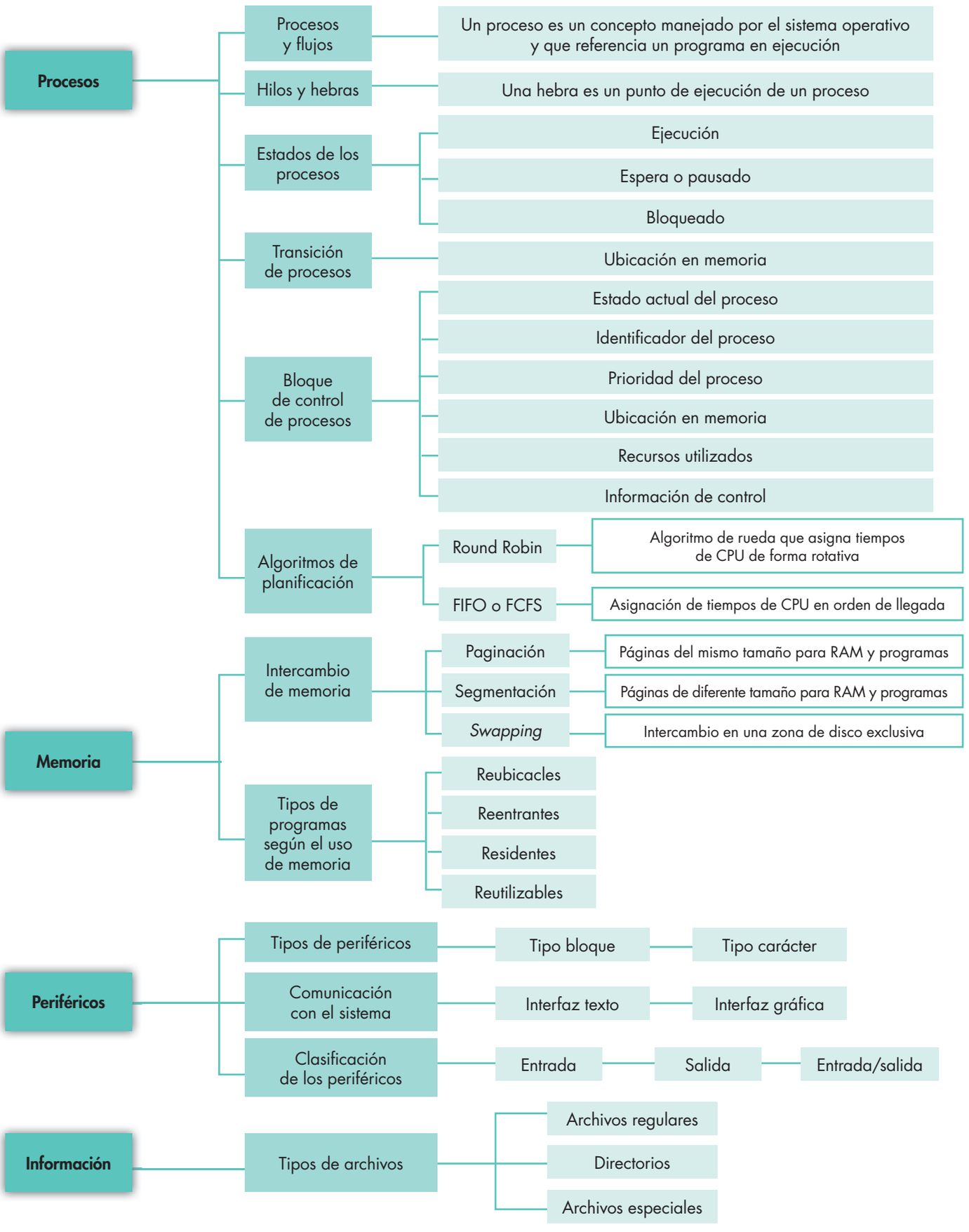
Ya iremos viendo en cada sistema operativo cuál es su sistema de archivos o ***File System.*** Cada uno de ellos hace una gestión diferente del espacio de almacenamiento, lo cual dependerá de si el sistema es multiusuario o monousuario, multitarea o monotarea, multiprocesador o monoprocesador, etc. En general, los tipos de archivos que gestiona todo sistema operativo son tres:

- **Archivos regulares o estándares.** Son los que contienen información del usuario, programas, documentos, texto, gráficos, etc.
- **Directorios.** Son archivos que contienen referencias a otros archivos regulares o a otros directorios.
- **Archivos especiales.** Los que no son de ninguno de los dos tipos anteriores.

Ten en cuenta

Un sistema de archivos determina de qué forma se almacena la información en un soporte y qué se puede hacer con ella.

Síntesis


Procesos
Procesos y flujos
Un proceso es un concepto manejado por el sistema operativo y que referencia un programa en ejecución
Hilos y hebras
Una hebra es un punto de ejecución de un proceso
Estados de los procesos
Ejecución
Espera o pausado
Bloqueado
Transición de procesos
Ubicación en memoria
Bloque de control de procesos
Estado actual del proceso
Identificador del proceso
Prioridad del proceso
Ubicación en memoria
Recursos utilizados
Información de control
Algoritmos de planificación
Round Robin
Algoritmo de rueda que asigna tiempos de CPU de forma rotativa
FIFO o FCFS
Asignación de tiempos de CPU en orden de llegada
Memoria
Intercambio de memoria
Paginación
Páginas del mismo tamaño para RAM y programas
Segmentación
Páginas de diferente tamaño para RAM y programas
Swapping
Intercambio en una zona de disco exclusiva
Tipos de programas según el uso de memoria
Reubicacles
Reentrantes
Residentes
Reutilizables
Periféricos
Tipos de periféricos
Tipo bloque
Tipo carácter
Comunicación con el sistema
Interfaz texto
Interfaz gráfica
Clasificación de los periféricos
Entrada
Salida
Entrada/salida
Información
Tipos de archivos
Archivos regulares
Directorios
Archivos especiales

Test de repaso

1. Cuando un programa o proceso se encuentra en estado bloqueado este puede pasar a ejecución:

a) Sí, directamente tras solucionar el conflicto que provoca el bloqueo.

b) Sí, pero previamente tendremos que colocar el proceso en la cola de trabajos pendientes.

c) No directamente, ya que previamente tiene que pasar por el estado en espera o preparado.

d) No, debido a que el bloqueo, si es de hardware, implica siempre el bloqueo definitivo del proceso.

2. Un proceso está preparado para ser ejecutado:

a) Si está esperando el turno para poder usar su intervalo de tiempo de ejecución de CPU.

b) Si está retenido por cualquier causa.

c) Si se están ejecutando instrucciones en ese momento de ese proceso.

d) Son correctas b y c.

3. Cuáles de las siguientes afirmaciones son ciertas respecto de los trabajos que llegan a una impresora en un sistema multiusuario:

a) Los trabajos se encolan en el denominado *spool*.

b) Los trabajos normalmente se gestionan mediante un algoritmo de planificación FIFO.

c) Se pueden asignar prioridades a los trabajos de impresión según estime el administrador.

d) Son correctas a, b y c.

4. Si utilizamos la técnica del *swapping*, el proceso mediante el cual el programa en ejecución pasa de memoria a la zona de *swap* se denomina:

a) Swap-in.

b) Swap-out.

c) Paginación.

d) Memoria virtual.

e) Segmentación.

5. La paginación se diferencia de la segmentación en que:

a) En la paginación la memoria se divide en *frames* del mismo tamaño y en la segmentación no.

b) En la paginación las partes en las que se divide el programa se llaman páginas y en la segmentación segmentos.

c) La paginación utiliza la denominada tabla de páginas y en la segmentación el *stack*.

d) Todas son ciertas.

6. La paginación es una técnica que consiste en dividir la memoria en zonas denominadas:

a) Páginas.

b) Frames.

c) Tabla de páginas.

d) Swap.

e) Todas son falsas.

7. Los directorios son:

a) Archivos para gestionar la entrada y salida de archivos regulares hacia o desde los periféricos.

b) Información del usuario, programas, documentos, texto, gráficos, etc.

c) Archivos que contienen referencias a otros archivos regulares o a otros directorios.

d) Son correctas a y b.

8. ¿Cuál de las siguientes afirmaciones es falsa?

a) La paginación utiliza segmentos de tamaño fijo y la segmentación marcos de tamaño variable.

b) En la paginación la memoria se divide en bloques del mismo tamaño y en la segmentación no.

c) En la paginación las zonas de memoria se llaman marcos y en la segmentación segmentos.

d) La paginación utiliza la denominada tabla de páginas y la segmentación la pila de memoria.

9. El cambio de contexto, puede producirse…

a) Entre diferentes procesos.

b) Entre diferentes hilos de un mismo proceso o entre distintos hilos de diferentes procesos.

c) Solamente entre hilos de un mismo proceso.

d) Son correctas a y b.

10. Los programas residentes:

a) Son utilizados por varios usuarios a la vez.

b) Una vez cargados en memoria permanecen en ella hasta que se apague el ordenador.

c) Se denominan vigilantes.

d) Son correctas a y b.

Solución: 1:c; 2:a; 3:d; 4:b; 5:d; 6:b; 7:c; 8:a; 9:d; 10:b.

Comprueba tu aprendizaje

1. Rellena la tabla de la derecha respecto de los estados de los procesos y las transiciones de los mismos. Utiliza las referencias a las transiciones explicadas en los puntos 2 y 3 de la unidad. Recuerda que las transiciones serán A, B, C o D. Si existe más de una opción, se indicarán ambas y se justificarán.

2. ¿Qué contiene el BCP o Bloque de Control de Procesos?

3. ¿Todo proceso tiene una entrada en el BCP?

4. ¿Qué algoritmo es el que mejor se puede utilizar para la ejecución de procesos en un sistema informático multiusuario?

Proceso	Estado	Transición	Nuevo Estado
P1	Ejecución		Bloqueado
P2	Pausado	C	
P3	Ejecución		Pausado
P4	Bloqueado	D	

5. Rellena la siguiente tabla:

GESTIÓN DE MEMORIA	DIVISIÓN DE LA MEMORIA	GESTIÓN DE DISCO	FRAGMENTACIÓN
Paginación			
Segmentación			
Swapping			

6. ¿Cómo se denominan los programas que pueden ser utilizados por varios usuarios y están cargados una sola vez en memoria?

7. Comenta alguna característica de los siguientes elementos:

- Archivos regulares o estándares.
- Directorios.
- Archivos especiales.

8. Rellena la siguiente tabla:

PERIFÉRICO	ENTRADA/SALIDA	INTERNO/EXTERNO	RÁPIDO/LENTO
Escáner			
Pizarra digital			
HDVD			
DVD			
Impresora			
Discos duros			
Monitor			
Router			