

SISTEMAS OPERATIVOS

Gestión de Memoria

Erwin Meza Vega
emezav@unicauca.edu.co

Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

GESTIÓN DE MEMORIA

Una gestión adecuada de la memoria permite que un sistema operativo administre de manera más eficiente la cantidad y el tamaño de cada uno de los procesos que puede ejecutar. Igualmente, de la gestión de memoria depende en gran medida la capacidad de respuesta del sistema al usuario y a los procesos que requieren sus servicios.

Gestión de memoria

- Componente del Sistema Operativo que se encarga de realizar las siguientes funciones (entre otras):
 - Manejar memoria libre y en uso
 - Asignar memoria para el (los) proceso(s)
 - Liberar la memoria asignada a el (los) proceso(s)

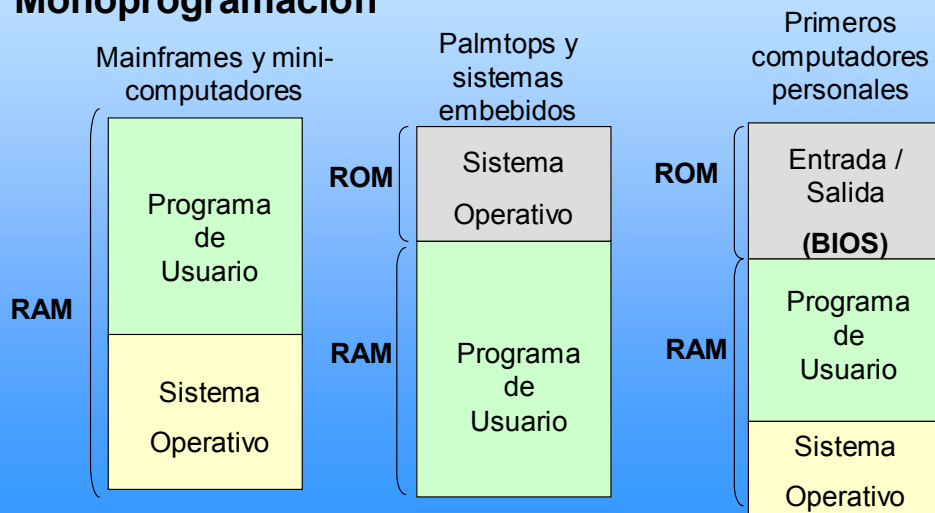
GESTIÓN DE MEMORIA

El administrador de memoria es el componente del sistema operativo que se encarga de gestionar la memoria disponible. Estas labores incluyen:

- Reservar y asignar memoria para los nuevos procesos y a aquellos procesos que en su ejecución soliciten memoria
- Mecanismos para administrar y llevar el rastro de la memoria usada y la memoria disponible
- Liberar la memoria reservada para un proceso cuando éste lo solicite, y liberar la memoria cuando un proceso termina su ejecución.

Esquemas básicos para la gestión de memoria: Monoprogramación

Monoprogramación



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

Monoprogramación

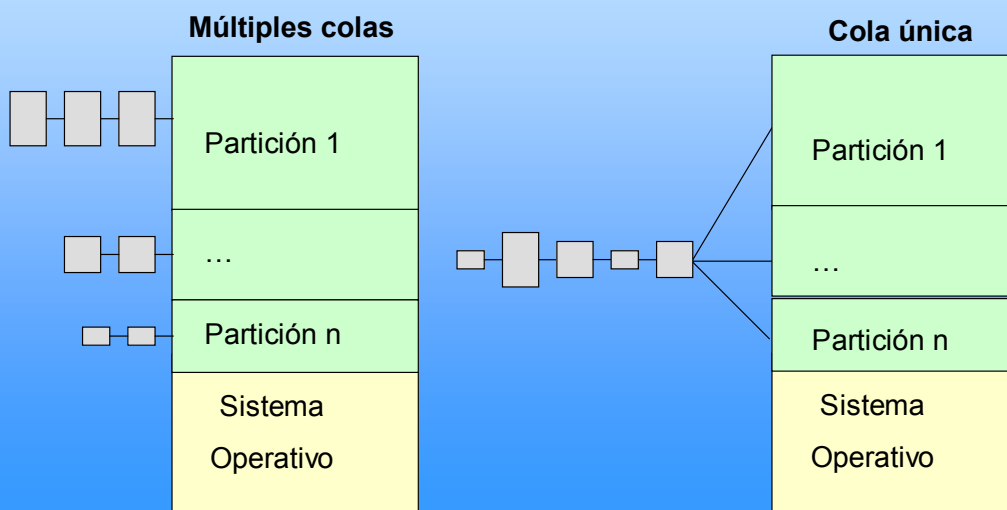
El primer esquema para el manejo de memoria consiste en ejecutar un único programa (monoprogramación), y compartir la memoria entre el programa y el sistema operativo. Este esquema presenta diferentes variaciones:

- El sistema operativo se puede ubicar en la parte baja de la memoria RAM, y el programa de usuario puede ocupar la memoria disponible. Este esquema fue utilizado en los primeros Mainframes y mini-computadores.
- El sistema operativo se ubica en la parte alta de la memoria, en una ROM (Read-Only Memory). De esta forma, el programa de usuario puede ocupar toda la memoria RAM, y no puede modificarla memoria ocupada por el sistema operativo. Este esquema es utilizado en dispositivos como palmtops y sistemas embebidos.
- El sistema operativo se encuentra en RAM, pero las rutinas básicas de manejo de entrada y salida se almacenan en una memoria ROM, llamada BIOS (Basic Input-Output System). Este esquema fue utilizado en los primeros computadores personales que utilizaban monoprogramación, como MS-DOS.

En cualquiera de los tres esquemas, el sistema operativo puede elegir el programa a ejecutar de una lista predefinida de programas, o puede solicitar al usuario el nuevo programa a ejecutar. Una vez que el programa ha finalizado, se pasa de nuevo el control al sistema operativo.

Esquemas básicos para la gestión de memoria: Multiprogramación

Multiprogramación con particiones fijas



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

Multiprogramación con particiones fijas

En el caso de la multiprogramación, la memoria RAM debe ser dividida en memoria para el Sistema Operativo y memoria para los procesos de usuario. A su vez la memoria para los procesos se divide en particiones continuas, que pueden ser de tamaño diferente de acuerdo con los procesos que se van a alojar en ellas. La disposición de estas particiones se puede establecer en el arranque del sistema.

Cuando se crea un nuevo proceso, se deberá seleccionar la partición de memoria que le será asignada, de acuerdo con su tamaño. Una vez que éste termine, se liberará la memoria de su partición, y quedará disponible para otro proceso.

Se debe tener en cuenta que el espacio sobrante dentro de la partición no podrá ser utilizado por otro proceso, y será desperdiciado. A este fenómeno se le conoce como **fragmentación interna**.

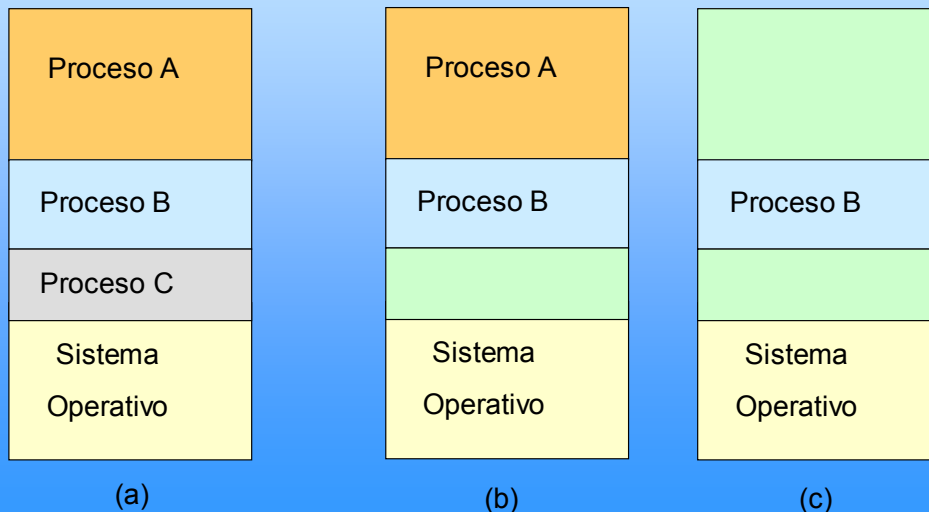
Para administrar las particiones, es posible tener una cola para cada partición de acuerdo con su tamaño, o una cola única para todas las particiones.

El problema de contar con múltiples colas reside en que mientras una de ellas puede tener varios procesos esperando para ser ejecutados, puede existir otra cola para una partición de mayor tamaño que no está siendo utilizada.

Igualmente, con una cola única se debe garantizar que los procesos reciben su oportunidad para ejecutarse. Esto se puede lograr manteniendo una partición libre, o implementando una cola de prioridad de acuerdo con las particiones libres.

Esquemas básicos para la gestión de memoria: Multiprogramación

Multiprogramación con particiones variables



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

Multiprogramación con particiones variables

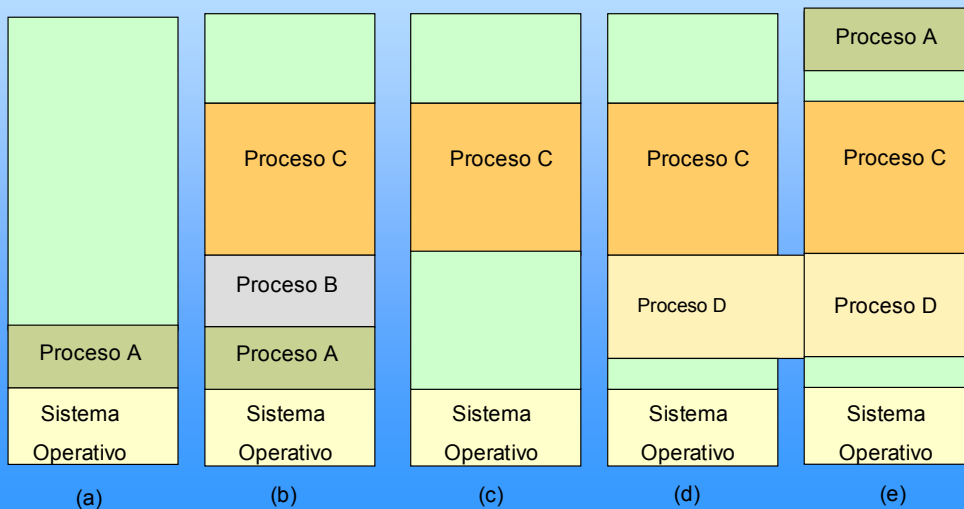
Otro esquema de gestión de memoria consiste en crear las particiones en forma dinámica, cuyo tamaño varía de acuerdo con los procesos que se están ejecutando. Cuando se crea un proceso, automáticamente se crea una partición con el tamaño mínimo que éste requiere (a). Cuando el proceso termina, se libera la memoria que se le ha asignado (b) y (c).

Este esquema presenta una desventaja: Debido a que el orden en que terminan los procesos no es necesariamente el mismo en el cual comienzan su ejecución, pueden quedar particiones de memoria vacías que no pueden ser utilizadas por procesos grandes, debido a que no son continuas (c). A este fenómeno se le conoce como **fragmentación externa**.

Una estrategia para resolver la fragmentación externa es la compactación de memoria, pero no es recomendable debido a que este proceso podría consumir una gran cantidad de tiempo de CPU.

Esquemas básicos para la gestión de memoria: Multiprogramación

Intercambio y relocalización



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

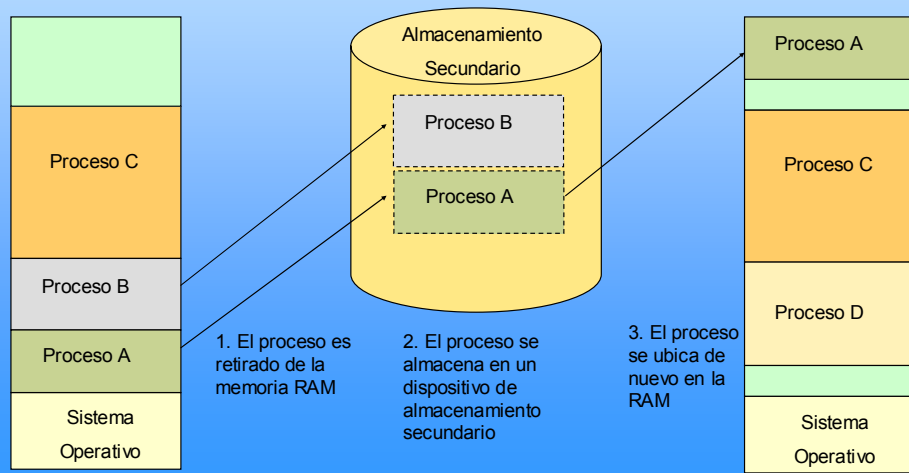
Intercambio y relocalización

En los sistemas de tiempo compartido es necesario que todas las tareas reciban tiempo de CPU. Debido a que la memoria RAM es limitada, es posible que exista la necesidad de tomar uno o más de los procesos en memoria y almacenarlos en otro sitio mientras se asigna la CPU a otro proceso. Considere el ejemplo de la diapositiva: Inicialmente sólo existe un proceso en memoria (a). Luego otros procesos entran a ejecución (b). Es posible que se requiera ejecutar otro proceso, pero por su tamaño no se puede ubicar en la memoria que se encuentra libre. En este caso, se debe tomar uno o más procesos y llevarlos a un dispositivo de almacenamiento secundario (como un disco duro) para obtener la memoria que requiere el nuevo proceso (c) y (d). Luego los procesos que se llevaron al almacenamiento secundario podrán ser ubicados de nuevo en memoria RAM para continuar con su ejecución. A este proceso se le denomina **intercambio**.

Es importante tener en cuenta que una vez que un proceso se lleva a memoria de intercambio, no se puede garantizar que cuando se regrese a la memoria RAM lo hará en la misma posición en la cual estaba originalmente, debido a que las particiones de memoria se asignan de forma dinámica. Por esta razón, es posible que el sistema operativo deba colocar el proceso en otra posición de memoria. A este proceso se le denomina **relocalización**.

Esquemas básicos para la gestión de memoria: Multiprogramación

Intercambio (swapping)



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

Intercambio

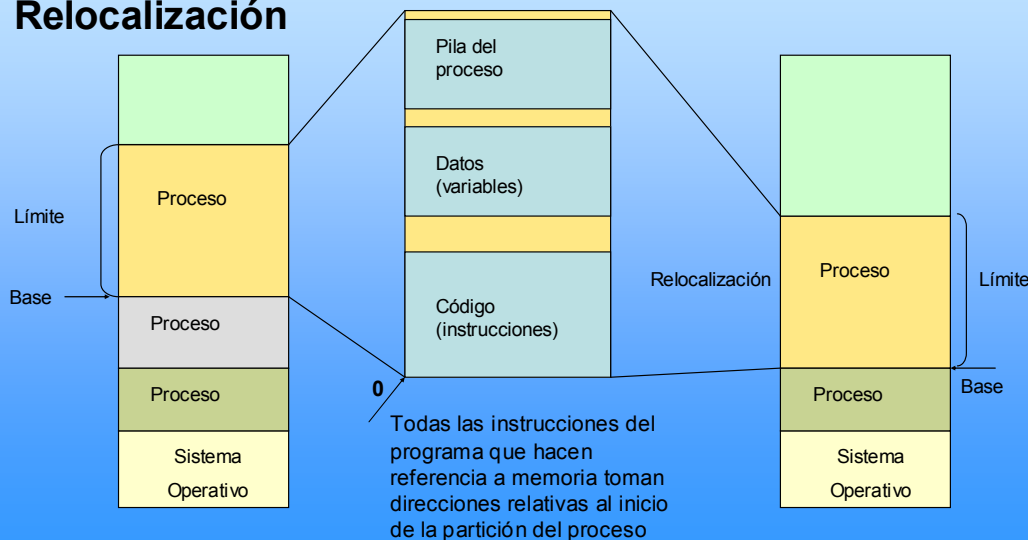
El intercambio permite que se puedan ejecutar más procesos de los que se pueden almacenar en la memoria física. En algún momento existirá una gran cantidad de procesos en memoria de los cuales sólo uno está en ejecución, y se requiere ejecutar un nuevo proceso.

Para permitir la ejecución del nuevo proceso, se deberá tomar uno o más de los procesos que se encuentran en RAM pero que no se están ejecutando y se deberán llevar a algún tipo de almacenamiento secundario (generalmente el disco duro) para crear un espacio libre. Con esto se podrá crear el nuevo proceso y asignarle la memoria que requiere para su ejecución.

En algún momento posterior se tomarán los procesos que se encuentran en el almacenamiento secundario y se llevarán a la memoria RAM, para continuar con su ejecución. Es posible que los procesos sean ubicados en una región de memoria RAM diferente a la cual tenían asignada cuando se cargaron en memoria por primera vez.

Esquemas básicos para la gestión de memoria: Multiprogramación

Relocalización



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

ESQUEMAS BÁSICOS PARA LA GESTIÓN DE MEMORIA

Relocalización

Para garantizar que un proceso pueda ser relocalizado en la memoria RAM sin inconvenientes, se debe cumplir que todas las instrucciones del programa que hacen referencia a la memoria (uso de variables, llamada a funciones, etc) sean relativas al inicio de la partición del proceso. Por ejemplo, cuando el programa realiza la siguiente operación:

$X=5$

Está asignando un valor a una dirección (lógica) dentro de la partición memoria asignada para él. Suponiendo que a la variable X le corresponde la dirección lógica 500, y que la partición del proceso inicia en la dirección física 10000, se tiene lo siguiente:

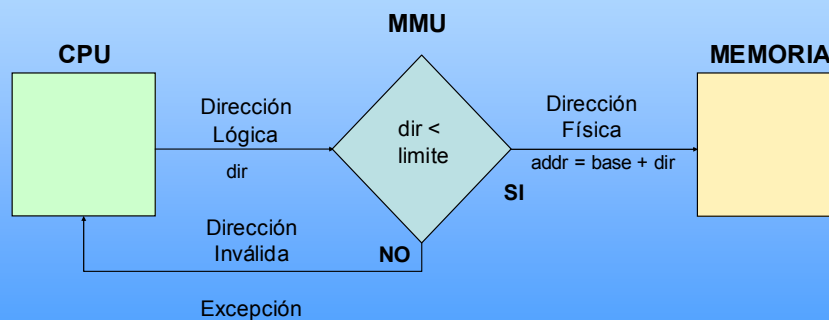
Dirección física = inicio de la partición + dirección lógica = $10000 + 500 = 10500$

Es decir que cada vez que el programa intente acceder a la variable X (dirección lógica 500), estará haciendo referencia a la dirección física 10500.

Para que la relocalización sea posible, el sistema operativo debe almacenar la dirección de inicio de la partición asignada a cada proceso (base), y el tamaño de dicha partición (límite). Con el fin de evitar que el sistema operativo tenga que realizar las operaciones aritméticas para transformar una dirección lógica en una dirección física se utilizan dos registros de la CPU (base y límite), que además permiten validar si la dirección lógica especificada se encuentre dentro de la partición de memoria del proceso.

Hardware para la gestión de memoria

ESQUEMA BÁSICO DE TRADUCCIÓN DE DIRECCIONES LÓGICAS A DIRECCIONES FÍSICAS DE MEMORIA



MMU: Unidad de gestión de memoria (Memory Management Unit)

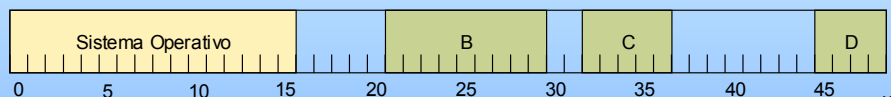
HARDWARE PARA LA GESTIÓN DE MEMORIA

El proceso de traducir una dirección lógica a una dirección física es realizado por la Unidad de Gestión de Memoria (MMU). Esta se encarga de validar si la dirección lógica especificada se encuentra dentro de la partición de memoria del proceso, al comparar el valor del registro límite con la dirección lógica. Si la dirección lógica es menor al límite, la dirección física en memoria se obtiene al sumar la base y la dirección lógica. Si por otra parte la dirección lógica se encuentra fuera de la partición asignada para el proceso, la MMU genera una excepción que generalmente ocasiona la terminación del proceso.

Esta traducción se debe repetir para cada dirección de memoria que se referencia dentro del programa, por lo cual es necesario que sea realizado por hardware. En algunos computadores la MMU es un circuito integrado independiente (como en el Motorola 68020), mientras que en otras arquitecturas la función de la MMU se encuentra integrada dentro de la CPU.

Administración de la memoria usada y disponible

DIAGRAMA DE LA MEMORIA USADA Y DISPONIBLE

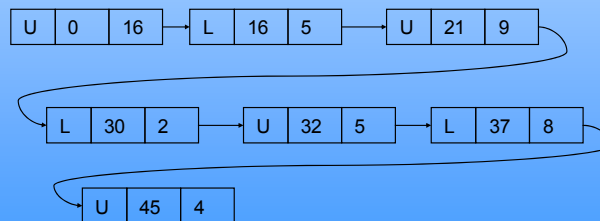


MAPA DE BITS

1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1
0	0	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1

Cada bit representa un espacio en la memoria
0 = Vacío, 1 = Ocupado.

LISTA ENLAZADA



Cada nodo representa una región de memoria continua.
U = Usado, L = Libre.

ADMINISTRACIÓN DE LA MEMORIA USADA Y DISPONIBLE

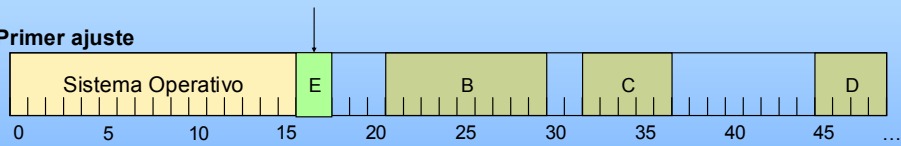
Otra de las funciones del administrador de memoria consiste en llevar el rastro de la memoria libre y ocupada. Para ello se pueden utilizar dos esquemas: Mapa de bits y lista enlazada.

- Mapa de bits: Cada unidad de almacenamiento de la memoria (byte o palabra) se representa con un bit, cuyo valor es 0 si se encuentra libre y 1 si se encuentra ocupada. La desventaja de este método radica en que el mapa de bits es proporcional al tamaño de la memoria existente, y por ser una estructura del sistema operativo se deberá almacenar también en memoria RAM.
- Lista enlazada: En este caso se almacena solamente la información de las regiones de memorias libres y utilizadas. Para cada región de memoria se almacena su dirección de inicio y su tamaño. La ventaja de este esquema consiste en que es proporcional al número de procesos que se encuentran en memoria.

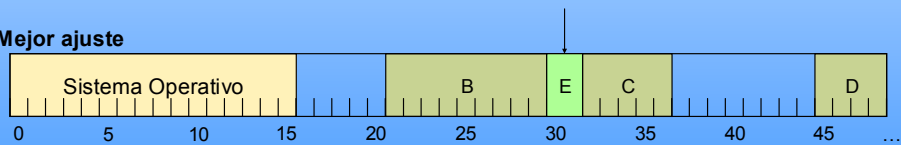
Esquemas de asignación de memoria continua

E = NUEVO PROCESO

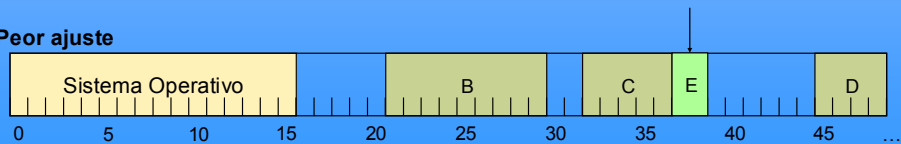
Primer ajuste



Mejor ajuste



Peor ajuste

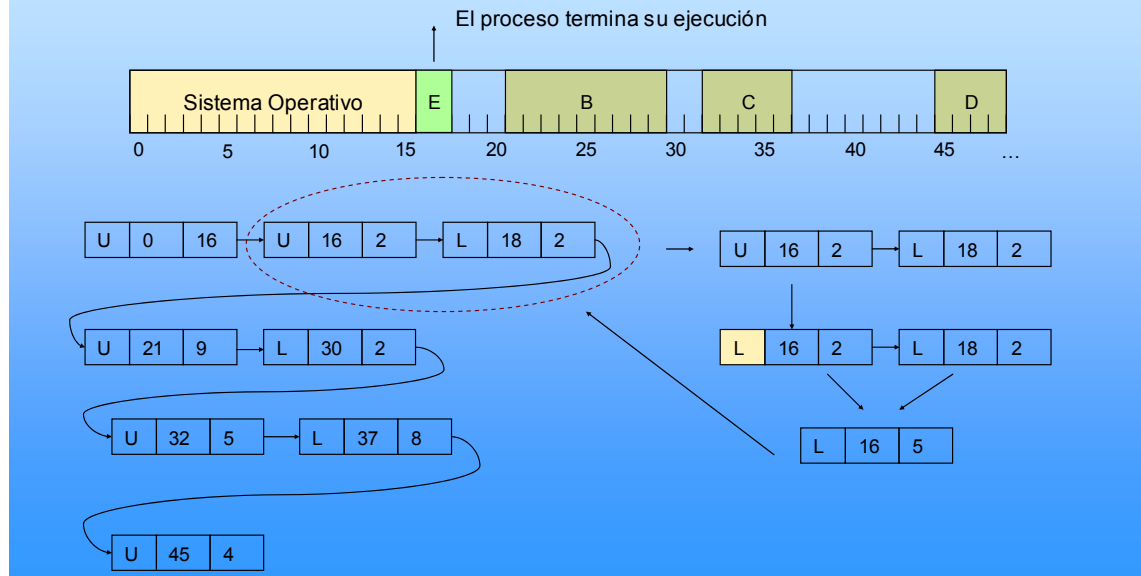


ESQUEMAS DE ASIGNACIÓN DE MEMORIA

Al crear un nuevo proceso se debe asignar una región de memoria continua para alojarlo. Este proceso se puede realizar de tres formas diferentes:

- **Primer ajuste:** Se busca la primera región de memoria con espacio disponible para contener el nuevo proceso. La búsqueda puede empezar desde el inicio de la memoria, o desde la última posición de memoria asignada.
- **Mejor ajuste:** Se recorre la lista de espacio disponible y se busca la región más pequeña en la cual se puede alojar el proceso.
- **Peor ajuste:** En este caso se elige la región de memoria más grande que existe y se reserva la memoria necesaria para alojar el proceso.

Liberación de memoria



Erwin Meza Vega (emezav@unicauca.edu.co)

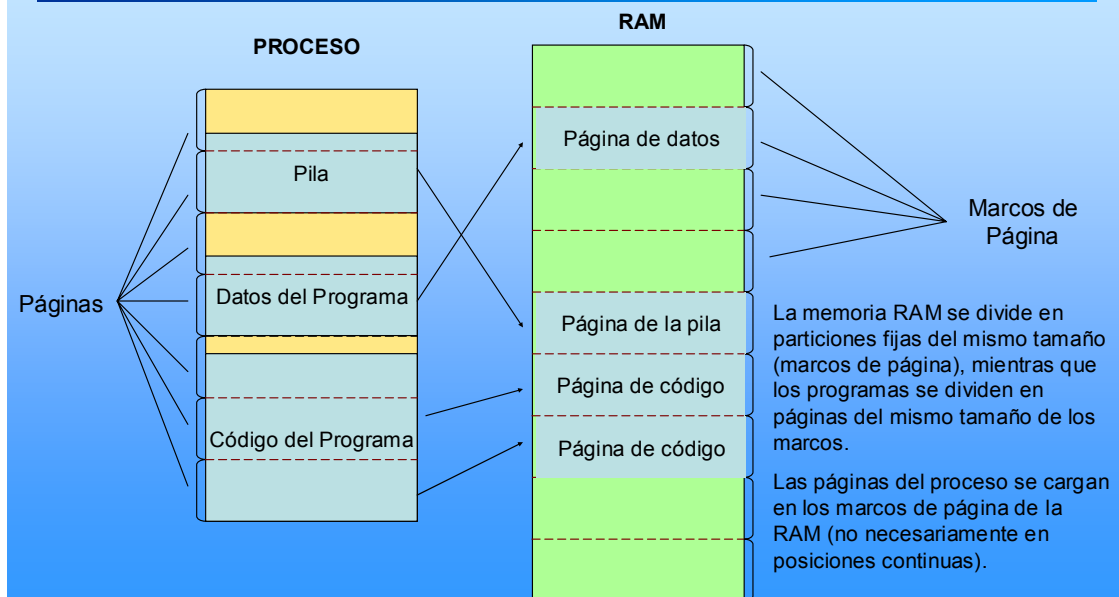
Universidad del Cauca

LIBERACIÓN DE MEMORIA

Una vez que un proceso ha terminado, es necesario que el administrador de memoria libere la partición que ha sido asignada para su ejecución. Esta tarea involucra buscar la partición que estaba siendo usada por el proceso, y marcarla como libre.

Se debe tener en cuenta que si existe una partición de memoria libre adyacente a la partición liberada, se deberán fusionar para crear una sola partición vacía.

Paginación



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

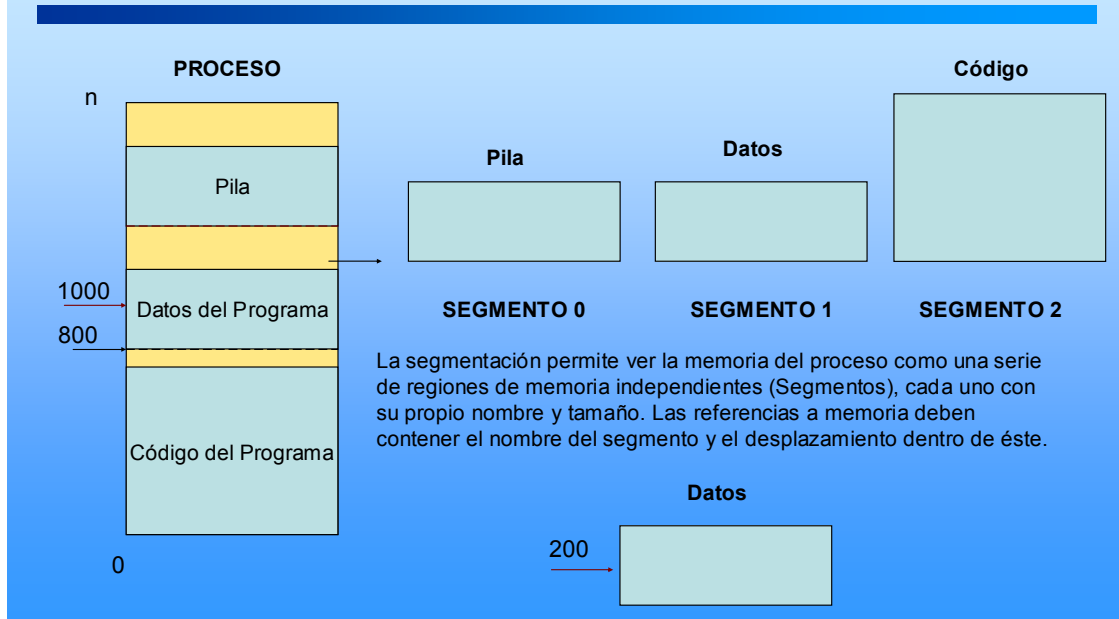
PAGINACIÓN

En los esquemas anteriores de manejo de memoria, esta tenía que ser asignada en una partición continua. La paginación evita este problema, al dividir la memoria de los procesos y la memoria física en páginas. En RAM las particiones lógicas se denominan **marcos de página**, y en la memoria del proceso se denominan **páginas**.

Cuando se prepara la ejecución de un nuevo proceso, se determina el número de páginas que ocupa en memoria. Luego se ubica el número suficiente de marcos de página en RAM y se carga el proceso en la memoria. Se debe tener en cuenta que en la forma más simple de paginación se debe garantizar que exista un número suficiente de marcos de página en RAM, pero no exige que éstos se encuentren dispuestos en forma continua. En esquemas más avanzados, se puede utilizar la memoria de intercambio para almacenar las páginas de un proceso que no estén siendo utilizadas, para así dar espacio para la ejecución de otros procesos.

A pesar de sus ventajas, la paginación exige que el administrador de la memoria deba implementar rutinas para la asignación, la liberación y el intercambio de los marcos de página, lo cual incrementa la complejidad de su programación. Igualmente, el sistema operativo deberá contar con las estructuras de datos que le permitan llevar el rastro de cuáles páginas tiene asignado cada proceso.

Segmentación



Erwin Meza Vega (emezav@unicauca.edu.co)

Universidad del Cauca

SEGMENTACIÓN

La segmentación permite separar la forma como el usuario aprecia la memoria y la forma como ésta está distribuida físicamente. En este sentido, la memoria de un proceso se puede dividir en sus diversos componentes: Código, Datos y Pila (Incluso el código puede ser dividido por módulos o funciones).

La segmentación busca dividir el programa en regiones de memoria independientes (segmentos) que pueden tener un nombre y un tamaño diferente. Desde este punto de vista, la referencias de memoria deben incluir el nombre del segmento al cual se está haciendo referencia, y el desplazamiento dentro de este segmento.

En un espacio de direcciones lineales, cualquier dirección se deberá tomar desde el inicio de la memoria del proceso. En un modelo con segmentación, esta dirección de memoria se considera a partir del inicio del segmento correspondiente. A manera de ejemplo, la dirección 1000 (que se encuentra en el segmento de datos) en un espacio de direcciones lineal corresponde a la dirección 200 en un esquema segmentado. Dado que el segmento de datos en el espacio lineal comienza en la dirección 800, la dirección 1000 se encuentra “desplazada” 200 a partir del inicio de este segmento. De esta forma, la dirección sería **SEGMENTO 1 + 200**.

Referencias

- TANENBAUM, Andrew. Modern Operating Systems. 2nd edition. Prentice Hall.
- SILBERSCHATZ, Abraham. GALVIN, Peter. GAGNE, Greg. Fundamentos de Sistemas Operativos. Séptima Edición.
- FLYNN, Ida. McHoes, Ann. Sistemas Operativos. Tercera Edición. Thomson Editores.