

8 ADMINISTRACIÓN DE MEMORIA

- 1 ANTECEDENTES
- 2 ASIGNACIÓN CONTIGUA
- 3 PAGINACIÓN
- 4 SEGMENTACIÓN
- 5 SEGMENTACIÓN CON PAGINACIÓN

8.1 ANTECEDENTES

- La Memoria (Almacenamiento Principal) es un aspecto central en la operación de un Sistema Computacional.
- La Memoria consiste en un gran arreglo de posiciones (palabras o bytes), donde cada una tiene una dirección única.

8.1 ANTECEDENTES

Evolución requerimientos de memoria de Windows

Versión	Aparición	Mínimo	Recomendado
<i>Operating System</i>	<i>Release Date</i>	<i>Minimum Memory Requirement</i>	<i>Recommended Memory</i>
Windows 1.0	November 1985	256KB	
Windows 2.03	November 1987	320KB	
Windows 3.0	March 1990	896KB	1MB
Windows 3.1	April 1992	2.6MB	4MB
Windows 95	August 1995	8MB	16MB
Windows NT 4.0	August 1996	32MB	96MB
Windows 98	June 1998	24MB	64MB
Windows ME	September 2000	32MB	128MB
Windows 2000 Professional	February 2000	64MB	128MB
Windows XP Home	October 2001	64MB	128MB
Windows XP Professional	October 2001	128MB	256MB

8.1 ANTECEDENTES

- Un programa debe ser traído a la memoria y puesto en un proceso para su ejecución.
- La CPU busca (fetch) cada instrucción en la memoria, de acuerdo al contenido del Contador de Programas.
- La ejecución de cada instrucción puede tener como consecuencia el acceso a posiciones adicionales de memoria.

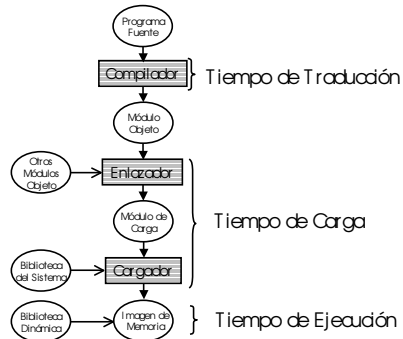
8.1 ANTECEDENTES

- Hay distintos aspectos a considerar en la Administración de Memoria y su evolución:
 - 1 Vinculación de Direcciones
 - 2 Espacio de Direcciones Lógico/Físico
 - 3 Carga Dinámica
 - 4 Enlace Dinámico
 - 5 Overlays
 - 6 Intercambio (swapping)

8.1.1 Vinculación de Direcciones

- La Cola de Entrada contiene procesos en disco esperando ser traídos a memoria para su ejecución.
- Un programa de usuario pasa por varias etapas (o tiempos) antes de ser ejecutado:
 - **de Traducción:** el Programa Fuente es traducido a un equivalente (Módulo Objeto), escrito generalmente en lenguaje de máquina convendand.
 - **de Carga:** El módulo Objeto contiene *referencias no resueltas*, que el Enlazador obtiene de otros módulos, generándose el Módulo de Carga.
 - **de Ejecución:** El proceso está ocupando una posición de la Memoria, requisito indispensable para ejecutar sus instrucciones.

8.1.1 Vinculación de Direcciones



8.1.1 Vinculación de Direcciones

- Las direcciones, al igual que las instrucciones y los datos, van sufriendo transformaciones al pasar un tiempo a otro.
- En cada paso, debe existir una correspondencia de la ubicación de las instrucciones y datos respecto de su ubicación en el paso anterior.
- Esto se refleja en que tanto el traductor como el enlazador y el cargador asocian (vinculan) a cada dato/instrucción una dirección, de naturaleza correspondiente al tiempo en que ello ocurre.

8.1.1 Vinculación de Direcciones

- La unión de la dirección de las instrucciones y los datos a las direcciones de memoria puede ocurrir en tres instantes (tiempos) diferentes.
 - **Traducción:** las posiciones de memoria se conocen a priori, se genera código absoluto; si la posición cambia, se debe recompilar. Ej. .COM en DOS.
 - **Carga:** las posiciones de memoria no se conocen a priori, se genera código reubicable. La vinculación se pospone hasta el instante de carga. Ej. .EXE en DOS.
 - **Ejecución:** el proceso puede ser movido de una posición de memoria a otra durante su ejecución, la unión se pospone hasta la ejecución (run-time). Ej. DLLs en Windows.

8.1.2 Espacio de Direcciones Lógico/Físico

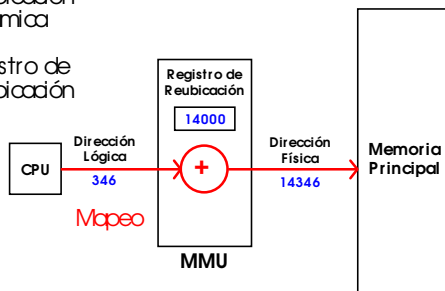
- El concepto de un Espacio de Direcciones Lógico vinculado a un Espacio de Direcciones Físico separado, es central para la Administración apropiada de la Memoria.
- El conjunto de todas las Direcciones Lógicas generadas por un programa es conocido como *Espacio de Direcciones Lógicas*.
- Algo análogo ocurre con las Direcciones Físicas.

8.1.2 Espacio de Direcciones Lógico/Físico

- Una Dirección Lógica es generada por la CPU, como consecuencia de la ejecución de una instrucción; también es referida como Dirección Virtual.
- Una Dirección Física es una dirección vista por la Unidad de Administración de Memoria (MMU).
- Las Direcciones Lógicas y Físicas son las mismas en los esquemas de vinculación de direcciones en tiempo de compilación y carga; pero difieren en los esquemas de vinculación de direcciones en tiempo de ejecución.
- La traducción de una Dirección Lógica (generada por un programa en ejecución) a una Dirección Física se denomina **Mapeo**.

8.1.2 Espacio de Direcciones Lógico/Físico

Reubicación
Dinámica
con
Registro de
Reubicación



8.1.2 Espacio de Direcciones Lógico/Físico

- Unidad de Administración de Memoria (MMU)
 - Dispositivo de hardware que mapea direcciones lógicas a físicas.
 - En la figura anterior, en la MMU el valor del registro de reubicación es sumado a cada dirección lógica, en el momento que es enviada a memoria física.
 - El programa de usuario solamente trata con direcciones lógicas; nunca ve las direcciones físicas reales.

8.1.3 Carga Dinámica

- Principio: una rutina no es cargada hasta que es invocada.
- En la Carga Dinámica hay una mejor utilización del espacio de memoria, pues una rutina no usada nunca es cargada.
- Es útil cuando se necesitan grandes cantidades de código para manejar casos poco frecuentes.
- No se necesita soporte especial del sistema operativo; es implementado a través del diseño de programas.

8.1.4 Enlace Dinámico

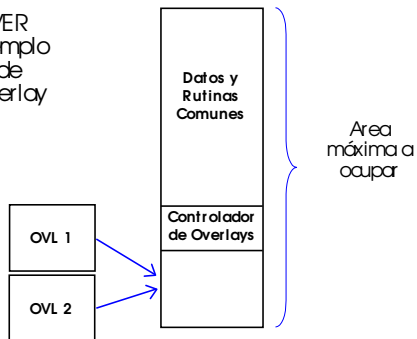
- Algunos Sistemas Operativos sólo soportan Enlace Estático.
- En el Enlace Dinámico, el enlace de un módulo puede ser pospuesto hasta tiempo de ejecución.
- Una pequeña pieza de código (un *stub*), es usada para ubicar la biblioteca de rutinas residente en memoria correspondiente. De no estarlo, es cargada (la primera vez que es referenciada).
- Luego, el *stub* se reemplaza a sí mismo con la dirección de la rutina y luego ejecuta la rutina.
- El Sistema Operativo verifica si la rutina está en el espacio de direcciones del proceso.

8.1.5 Overlays

- Principio: mantener en memoria sólo instrucciones y datos a ser referenciados posteriormente.
- Esto es necesario cuando el proceso es de mayor tamaño que la memoria asignada a éste.
- No se necesita soporte especial del Sistema Operativo, pues es implementado por el programador.
- El diseño y la programación de la estructura del overlay son complejos.

8.1.5 Overlays

VER
Ejemplo
de
Overlay



8.1.6 Intercambio (Swapping)

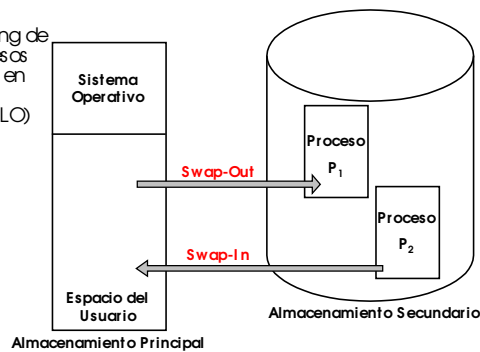
- Un proceso puede ser sacado (swap out) temporalmente de la memoria a un almacenamiento de respaldo (backing store) y luego ser traído de vuelta a la memoria (swap in) para continuar su ejecución.
- El almacenamiento de respaldo es un disco rápido, suficientemente grande para acomodar copias de todas las imágenes de memoria para todos los usuarios; debe proveer acceso directo a las imágenes de memoria.

8.1.6 Intercambio (Swapping)

- Roll out/in es una variante del swapping, usada para algoritmos de itineración basada en prioridad; el proceso de menor prioridad es sacado para que el proceso de mayor prioridad sea cargado y ejecutado.
- La mayor parte del tiempo de swapping es de transferencia; el tiempo total de transferencia es directamente proporcional a la cantidad de memoria intercambiada.
- Hay versiones modificadas del swapping en varios sistemas operativos.

8.1.6 Intercambio (Swapping)

Swapping de 2 procesos basado en disco (EJEMPLO)



8.2 ASIGNACIÓN CONTIGUA

- La Memoria Principal está dividida usualmente en dos particiones:
 - El Sistema Operativo residente está almacenado generalmente en memoria baja, junto con el vector de interrupciones.
 - Los procesos de usuario, se cargan en memoria alta.
- Cada proceso ocupa una porción única y contigua de memoria.
- La pregunta es cómo asignar la Memoria Principal disponible a los procesos que se encuentran en la Cola de Entrada.

8.2 ASIGNACIÓN CONTIGUA

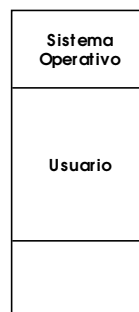
- Se analizará:
 - 1 Asignación con Partición Única
 - 2 Asignación con Partición Múltiple
 - 3 Fragmentación

8.2.1 Asignación con Partición Única

- Se usa el esquema basado en registro de reubicación para proteger los procesos de usuarios unos de otros y de cambiar el código y/o datos del sistema operativo.
- El registro de reubicación contiene el valor de la dirección física inferior; el registro límite contiene el rango de direcciones lógicas; toda dirección lógica debe ser menor o igual que el registro límite.

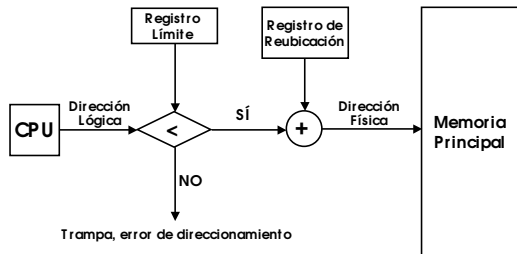
8.2.1 Asignación con Partición Única

Partición Única de Memoria



8.2.1 Asignación con Partición Única

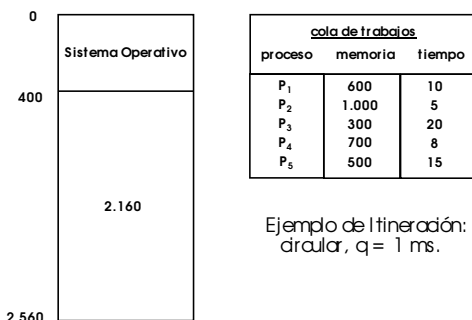
Soporte de Hardware para Registros de Reubicación y Límite



8.2.2 Asignación con Particiones Múltiples

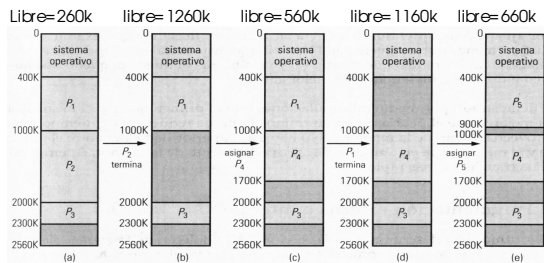
- Las particiones pueden ser:
 - Fijas (MFT). No varían ni en tamaño ni en número.
 - Variables (MVT). Varían en tamaño y número.
- Se pueden generar cavidades (hole); las hay de varios tamaños, esparidas a lo largo de la memoria.
- Cuando un proceso arriba, se le asigna una cavidad suficientemente grande para acomodarlo.
- El Sistema Operativo mantiene información acerca de:
 - particiones asignadas
 - particiones libres (holes)

8.2.2 Asignación con Particiones Múltiples



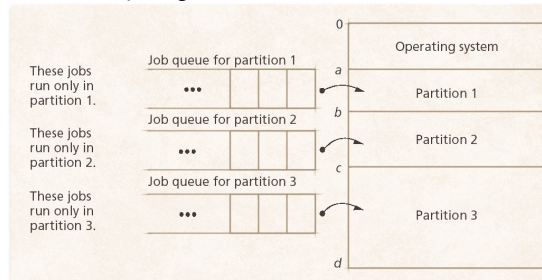
8.2.2 Asignación con Particiones Múltiples

Asignación de Memoria e Itineración a Largo Plazo
Cada uno es un mapa de memoria



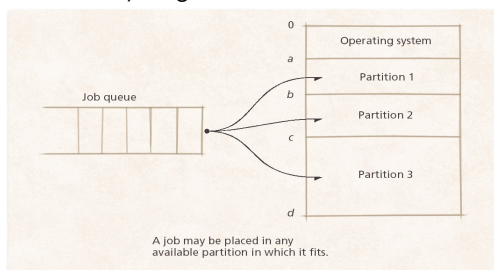
8.2.2 Asignación con Particiones Múltiples

Multiprogramación en particiones fijas con traducción y carga absoluta.



8.2.2 Asignación con Particiones Múltiples

Multiprogramación en particiones fijas con traducción y carga reubicable.



8.2.2 Asignación con Particiones Múltiples

Problema de la Asignación Dinámica de Memoria

Cómo satisfacer un requerimiento de tamaño N de una lista de cavidades libres.

- **Primer calce (first-fit):** Asigna la primera cavidad suficientemente grande.
- **Mejor calce (best-fit):** Asigna la cavidad más pequeña suficientemente grande; debe buscar en la lista completa, a menos que esté ordenada por tamaño (ascendente). Genera el menor sobrante.
- **Peor calce (worst-fit):** Asigna la cavidad más grande; también debe buscar en la lista completa, a menos que esté ordenada por tamaño (descendente). Genera el sobrante más grande.
- ¿cuál es mejor? ¿cuál es más rápido?

8.2.3 Fragmentación

Hasta lo que se ha visto no hay fragmentación, existen dos tipos:

- **Fragmentación externa:** espacio total de memoria que existe para satisfacer un requerimiento, pero que no es contigua. La memoria está fragmentada en un gran número de cavidades pequeñas. (ver 8.2.2 tercer mapa, P5 = 500k, hay libre=560k pero no contiguas)
- **Fragmentación interna:** la memoria asignada puede ser de tamaño un poco mayor que la memoria requerida; esta diferencia en tamaño se denomina memoria interna respecto de la partición, pero que no está siendo usada y no puede ser reasignada.

8.2.3 Fragmentación

Interna

Proceso
no reassignable (misma partición)

Externa

Proceso
reassignable (otra partición)

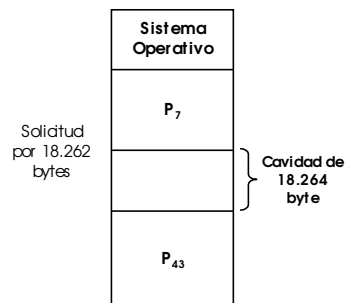
8.2.3 Fragmentación

Comparación de algoritmos de asignación:

- **First-fit** es mejor en términos de velocidad.
- En MFT, **best-fit** es mejor en términos de utilización del almacenamiento (disminuye la fragmentación interna).
- En MVT, **worst-fit** es mejor en términos de utilización del almacenamiento (disminuye la fragmentación externa).

8.2.3 Fragmentación

Asignación de Memoria en múltiplos de un N° de bytes, lo cual genera Fragmentación Interna



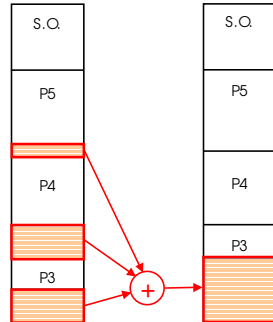
8.2.3 Fragmentación

- La fragmentación puede ser evitada o pospuesta, pero no eliminada.
- Un método para reducir la fragmentación externa es la **compactación**.
 - Arrastra los contenidos de la memoria ocupada para poner toda la memoria libre junta, en un bloque único.
 - La compactación es posible sólo si la reubicación es dinámica y es hecha en tiempo de ejecución.
 - Problema de E/S
 - * Mover un Job en memoria mientras éste está llevando a cabo E/S
 - * Hacer la E/S sólo en los Buffers del S.O.

8.2.3 Fragmentación

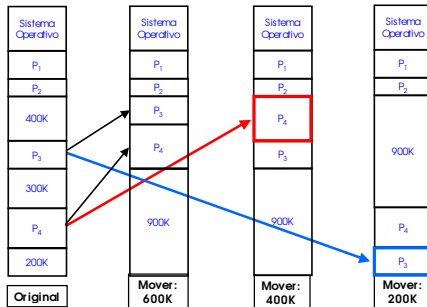
Compactación

Se desplazan los procesos P4 y P3, para esto se reubicaron todas las direcciones internas (cambiar de lugar programa, datos y modificar el registro base con la nueva dirección) de estos procesos.



8.2.3 Fragmentación

Diferentes alternativas de Compactación: Depende del costo. P1=200k; P2=100K; P3=200K; P4=400K. Si sólo necesito 450k para un proceso nuevo a partir del original ¿qué proceso podría mover? Se crea uno con el tamaño suficiente, pero no una sola cavidad grande.



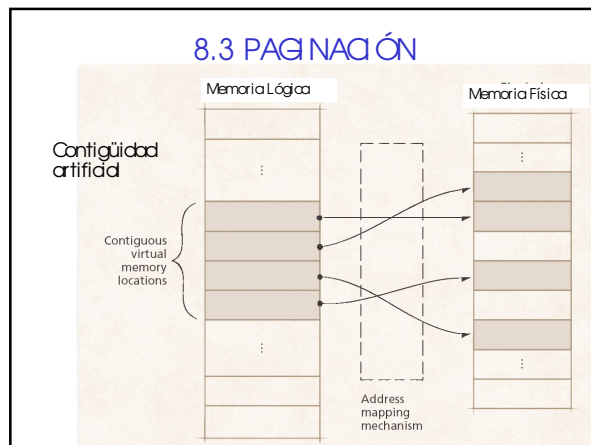
8.2.3 Fragmentación

- El intercambio se puede combinar con la compactación.
- Proceso en memoria -> disco, luego proceso en disco -> a memoria
- Restricciones:
 - Reubicación estática: mismas posiciones de memoria que ocupaba antes. ¿se puede compactar?
 - Reubicación dinámica: se podrá colocar en otra posición. ¿se puede compactar?
 - Si el intercambio forma parte del sistema, el costo por compactación es mínimo.

8.3 PAGINACIÓN

- La Paginación es un esquema de gestión de memoria que permite que el espacio de direcciones físico de un proceso sea no contiguo.
- La paginación evita el problema de hacer calzar el tamaño variable de los programas con la capacidad de la memoria disponible, eliminándose la fragmentación externa.
- Hoy en día, la paginación se hace de manera combinada entre el hardware y el SO.

8.3 PAGINACIÓN



8.3 PAGINACIÓN

- 1 Método Básico
- 2 Estructura de la Tabla de Páginas
- 3 Paginación Multinivel
- 4 Tabla de Páginas Invertida
- 5 Páginas Compartidas
