

Sistemas Operativos

Administración de la memoria

Clase 8 - Unidad III


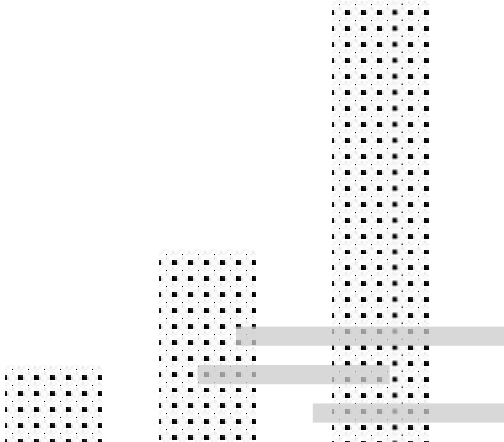
Lic. Alexis Sostersich
sostersich.alexis@uader.edu.ar



Teoría – Sistemas Operativos

Administración de la memoria

Unidad III – Clase 8

- Reubicación
 - Paginación
 - Segmentación
- 
- 

Reubicación

Antes de considerar formas de tratar los problemas del particionamiento, se va a aclarar un aspecto relacionado con la colocación de los procesos en la memoria. Cuando se utiliza el esquema de particionamiento fijo, se espera que un proceso siempre se asigne a la misma partición.

Es decir, sea cual sea la partición seleccionada cuando se carga un nuevo proceso, ésta será la utilizada para el intercambio del proceso entre la memoria y el área de swap en disco. En este caso, se utiliza un cargador sencillo: cuando el proceso se carga por primera vez, todas las referencias de la memoria relativas del código se reemplazan por direcciones de la memoria principal absolutas, determinadas por la dirección base del proceso cargado.

Reubicación

En el caso de particiones de igual tamaño, y en el caso de una única cola de procesos para particiones de distinto tamaño, un proceso puede ocupar diferentes particiones durante el transcurso de su ciclo de vida. Cuando la imagen de un proceso se crea por primera vez, se carga en la misma partición de memoria principal.

Más adelante, el proceso podría llevarse a disco; cuando se trae a la memoria principal posteriormente, podría asignarse a una partición distinta de la última vez. Lo mismo ocurre en el caso del particionamiento dinámico. Observe que cuando un proceso ocupa regiones diferentes de memoria en las ocasiones que se trae a la memoria. Más aún, cuando se utiliza la compactación, los procesos se desplazan mientras están en la memoria principal.

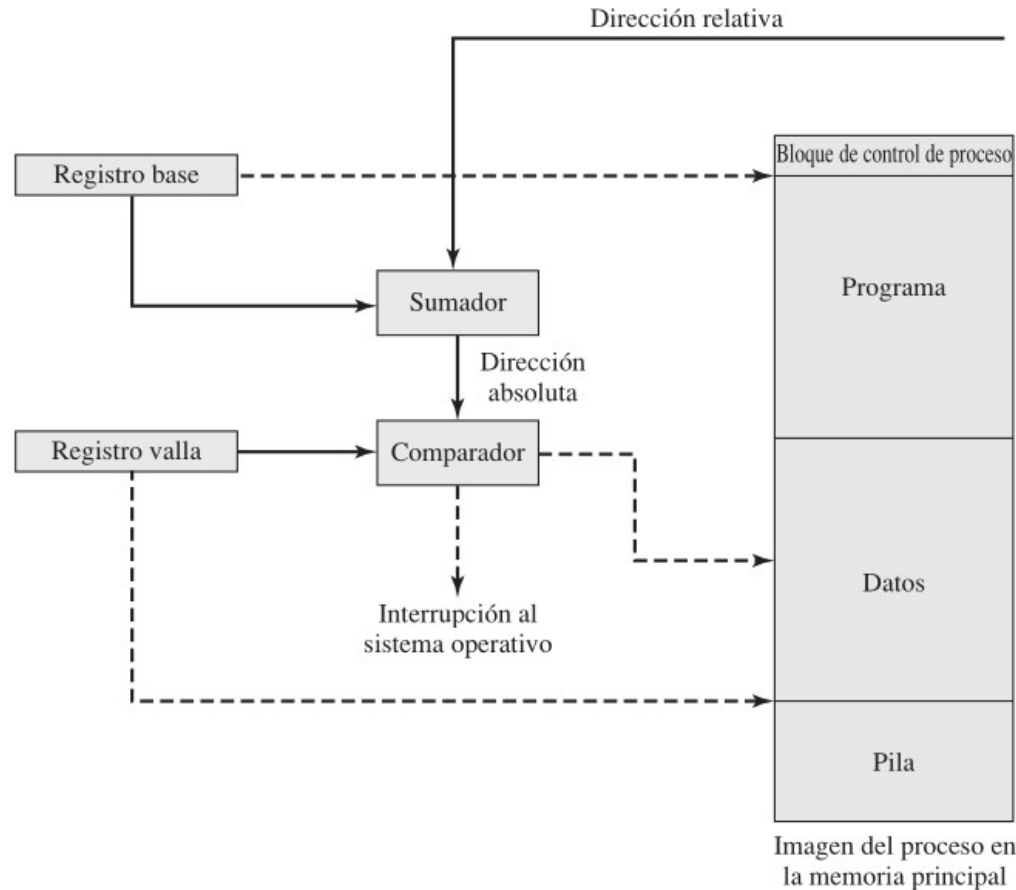
Por tanto, las ubicaciones (de las instrucciones y los datos) referenciadas por un proceso no son fijas. Cambiarán cada vez que un proceso se intercambia o se desplaza. Para resolver este problema, se realiza una distinción entre varios tipos de direcciones.

Reubicación

Una dirección lógica es una referencia a una ubicación de memoria independiente de la asignación actual de datos a la memoria; se debe llevar a cabo una traducción a una dirección física antes de que se alcance el acceso a la memoria. Una dirección relativa es un ejemplo particular de dirección lógica, en el que la dirección se expresa como una ubicación relativa a algún punto conocido, normalmente un valor en un registro del procesador. Una dirección física, o dirección absoluta, es una ubicación real de la memoria principal.

Los programas que emplean direcciones relativas de memoria se cargan utilizando carga dinámica en tiempo de ejecución. Normalmente, todas las referencias de memoria de los procesos cargados son relativas al origen del programa. Por tanto, se necesita un mecanismo hardware para traducir las direcciones relativas a direcciones físicas de la memoria principal, en tiempo de ejecución de la instrucción que contiene dicha referencia.

Soporte hardware para la reubicación



Paginación

Tanto las particiones de tamaño fijo como variable son ineficientes en el uso de la memoria; los primeros provocan fragmentación interna, los últimos fragmentación externa.

Supóngase, sin embargo, que la memoria principal se divide en porciones de tamaño fijo relativamente pequeños, y que cada proceso también se divide en porciones pequeñas del mismo tamaño fijo. A dichas porciones del proceso, conocidas como páginas, se les asigna porciones disponibles de memoria, conocidas como marcos, o marcos de páginas. Esta sección muestra que el espacio de memoria malgastado por cada proceso debido a la fragmentación interna corresponde sólo a una fracción de la última página de un proceso. No existe fragmentación externa.

Paginación

Vemos el uso de páginas y marcos. En un momento dado, algunos de los marcos de la memoria están en uso y algunos están libres. El sistema operativo mantiene una lista de marcos libres.

| Marco número | Memoria principal |
|--------------|-------------------|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| 10 | |
| 11 | |
| 12 | |
| 13 | |
| 14 | |

(a) Quince marcos disponibles

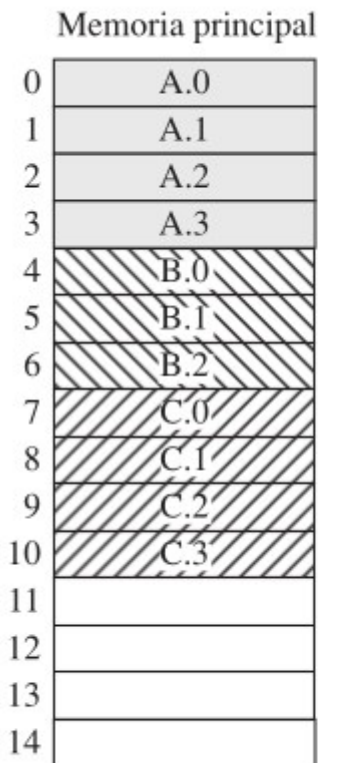
| Marco número | Memoria principal |
|--------------|-------------------|
| 0 | A.0 |
| 1 | A.1 |
| 2 | A.2 |
| 3 | A.3 |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| 10 | |
| 11 | |
| 12 | |
| 13 | |
| 14 | |

(b) Cargar proceso A

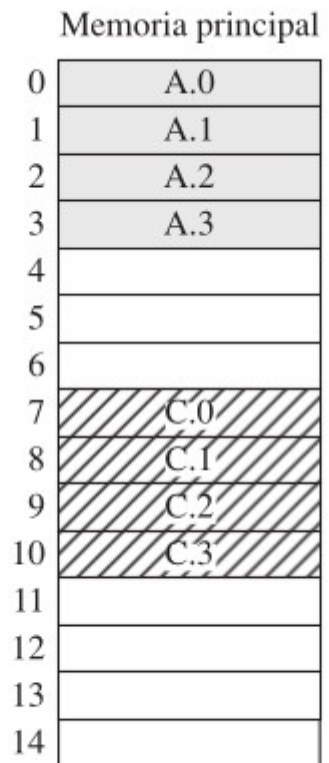
| Marco número | Memoria principal |
|--------------|-------------------|
| 0 | A.0 |
| 1 | A.1 |
| 2 | A.2 |
| 3 | A.3 |
| 4 | B.0 |
| 5 | B.1 |
| 6 | B.2 |
| 7 | |
| 8 | |
| 9 | |
| 10 | |
| 11 | |
| 12 | |
| 13 | |
| 14 | |

(c) Cargar proceso B

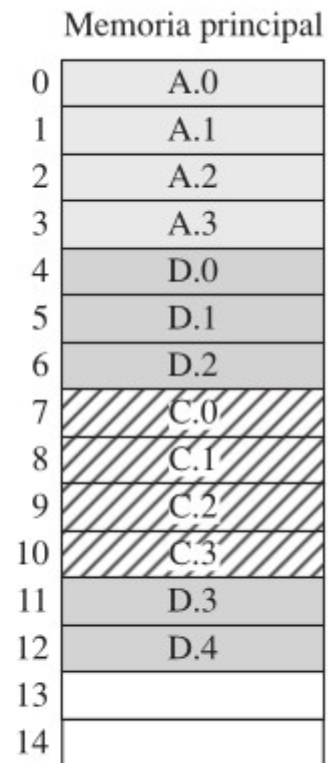
Paginación



(d) Cargar proceso C



(e) Intercambiar B



(f) Cargar proceso D

Paginación

Suponga, que no hay suficientes marcos contiguos sin utilizar para ubicar un proceso. ¿Esto evitaría que el sistema operativo cargara el proceso D?

La respuesta es no, porque se puede utilizar el concepto de dirección lógica. Un registro base sencillo de direcciones no basta en esta ocasión. En su lugar, el sistema operativo mantiene una tabla de páginas por cada proceso. La tabla de páginas muestra la ubicación del marco por cada página del proceso. Dentro del programa, cada dirección lógica está formada por un número de página y un desplazamiento dentro de la página. Es importante recordar que en el caso de una partición simple, una dirección lógica es la ubicación de una palabra relativa al comienzo del programa; el procesador la traduce en una dirección física. Con paginación, la traducción de direcciones lógicas a físicas las realiza también el hardware del procesador. Ahora el procesador debe conocer cómo acceder a la tabla de páginas del proceso actual. Presentado como una dirección lógica (número de página, desplazamiento), el procesador utiliza la tabla de páginas para producir una dirección física (número de marco, desplazamiento).

Paginación

Las cinco páginas del proceso D se cargan en los marcos 4, 5, 6, 11 y 12. Vemos las diferentes tablas de páginas en este momento. Una tabla de páginas contiene una entrada por cada página del proceso, de forma que la tabla se indexe fácilmente por el número de página (iniciando en la página 0). Cada entrada de la tabla de páginas contiene el número del marco en la memoria principal, si existe, que contiene la página correspondiente. Adicionalmente, el sistema operativo mantiene una única lista de marcos libres de todos los marcos de la memoria que se encuentran actualmente no ocupados y disponibles para las páginas.

Vemos que la paginación simple, tal y como se describe aquí, es similar al particionamiento fijo. Las diferencias son que, con la paginación, las particiones son bastante pequeñas; un programa podría ocupar más de una partición; y dichas particiones no necesitan ser contiguas.

| | |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

Tabla de
páginas del
proceso A

| | |
|---|---|
| 0 | — |
| 1 | — |
| 2 | — |

Tabla de
páginas del
proceso B

| | |
|---|----|
| 0 | 7 |
| 1 | 8 |
| 2 | 9 |
| 3 | 10 |

Tabla de
páginas del
proceso C

| | |
|---|----|
| 0 | 4 |
| 1 | 5 |
| 2 | 6 |
| 3 | 11 |
| 4 | 12 |

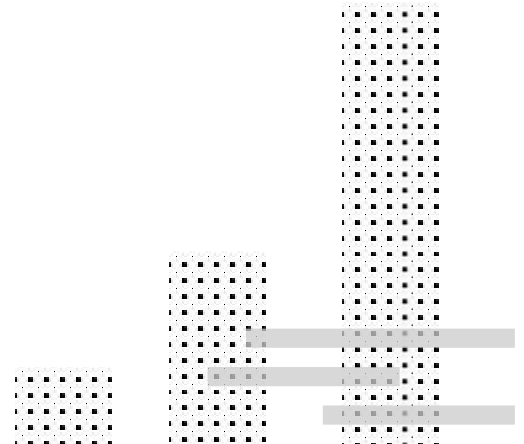
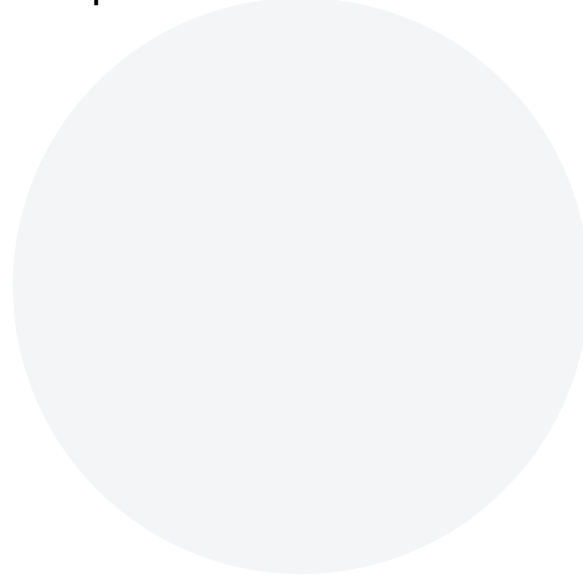
Tabla de
páginas del
proceso D

| |
|----|
| 13 |
| 14 |

Lista de
marcos libre

Paginación

Resumiendo, con la paginación simple, la memoria principal se divide en muchos marcos pequeños de igual tamaño. Cada proceso se divide en páginas de igual tamaño; los procesos más pequeños requieren menos páginas, los procesos mayores requieren más. Cuando un proceso se trae a la memoria, todas sus páginas se cargan en los marcos disponibles, y se establece una tabla de páginas. Esta técnica resuelve muchos de los problemas inherentes en el particionamiento.

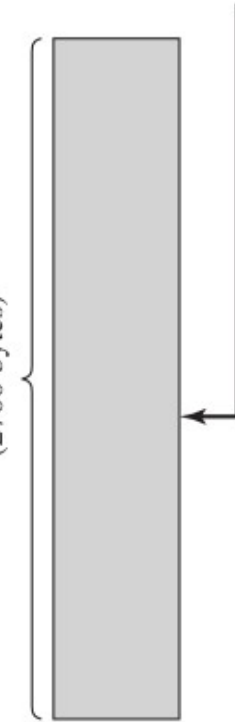


Paginación

Dirección relativa = 1502

0000010111011110

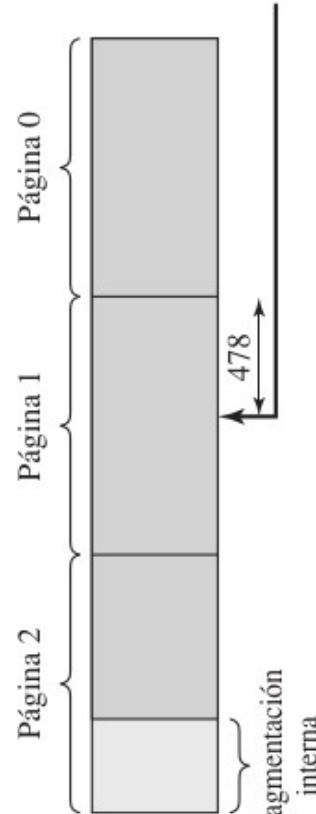
Proceso de usuario
(2700 bytes)



(a) Particionado

Dirección lógica =
Página# = 1, Desplazamiento = 478

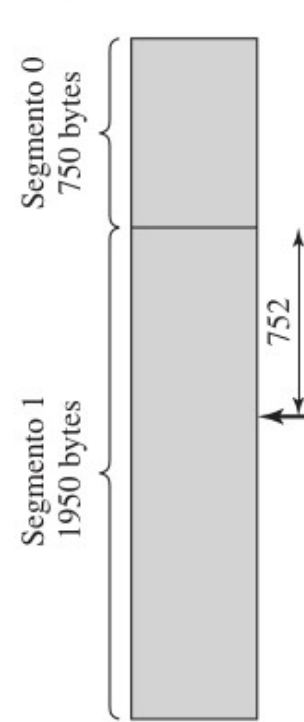
0000010111011110



(b) Paginación
(tamaño página = 1K)

Dirección lógica =
Segmento# = 1, desplazamiento = 752

0001001011110000



(c) Segmentación

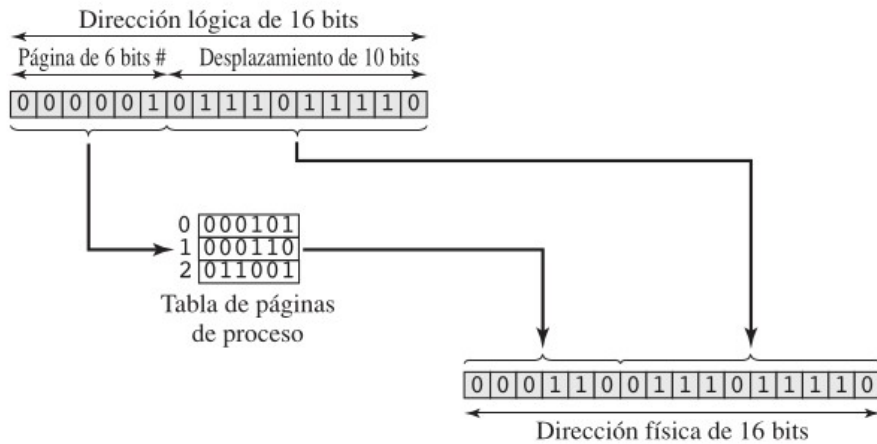
Segmentación

Un programa de usuario se puede subdividir utilizando segmentación, en la cual el programa y sus datos asociados se dividen en un número de segmentos. No se requiere que todos los programas sean de la misma longitud, aunque hay una longitud máxima de segmento. Como en el caso de la paginación, una dirección lógica utilizando segmentación está compuesta por dos partes, en este caso un número de segmento y un desplazamiento.

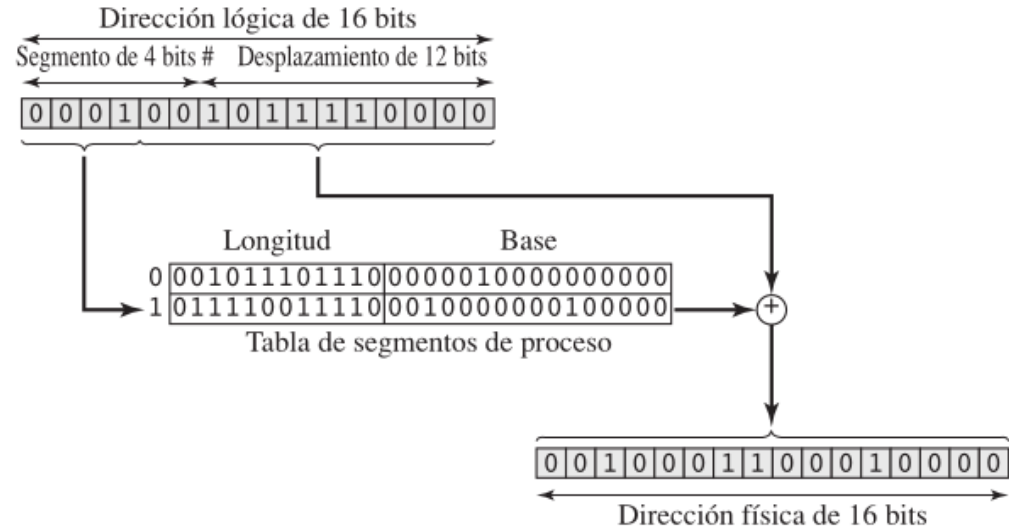
Debido al uso de segmentos de distinto tamaño, la segmentación es similar al particionamiento dinámico. En la ausencia de un esquema de overlays o el uso de la memoria virtual, se necesitaría que todos los segmentos de un programa se cargaran en la memoria para su ejecución. La diferencia, comparada con el particionamiento dinámico, es que con la segmentación un programa podría ocupar más de una partición, y estas particiones no necesitan ser contiguas. La segmentación elimina la fragmentación interna pero, al igual que el particionamiento dinámico, sufre de fragmentación externa.

Sin embargo, debido a que el proceso se divide en varias piezas más pequeñas, la fragmentación externa debería ser menor.

Segmentación



(a) Paginación



(b) Segmentación

Segmentación

Mientras que la paginación es invisible al programador, la segmentación es normalmente visible y se proporciona como una utilidad para organizar programas y datos. Normalmente, el programador o compilador asignará programas y datos a diferentes segmentos. Para los propósitos de la programación modular, los programas o datos se pueden dividir posteriormente en múltiples segmentos. El inconveniente principal de este servicio es que el programador debe ser consciente de la limitación de tamaño de segmento máximo.

Otra consecuencia de utilizar segmentos de distinto tamaño es que no hay una relación simple entre direcciones lógicas y direcciones físicas. De forma análoga a la paginación, un esquema de segmentación sencillo haría uso de una tabla de segmentos por cada proceso y una lista de bloques libre de memoria principal. Cada entrada de la tabla de segmentos tendría que proporcionar la dirección inicial de la memoria principal del correspondiente segmento. La entrada también debería proporcionar la longitud del segmento, para asegurar que no se utilizan direcciones no válidas. Cuando un proceso entra en el estado Ejecutando, la dirección de su tabla de segmentos se carga en un registro especial utilizado por el hardware de gestión de la memoria.

Segmentación

Considérese una dirección de $n+m$ bits, donde los n bits de la izquierda corresponden al número de segmento y los m bits de la derecha corresponden al desplazamiento. En el ejemplo, $n = 4$ y $m = 12$. Por tanto, el tamaño de segmento máximo es $2^{12} = 4096$. Se necesita llevar a cabo los siguientes pasos para la traducción de direcciones:

- Extraer el número de segmento como los n bits de la izquierda de la dirección lógica.
- Utilizar el número de segmento como un índice a la tabla de segmentos del proceso para encontrar la dirección física inicial del segmento.
- Comparar el desplazamiento, expresado como los m bits de la derecha, y la longitud del segmento. Si el desplazamiento es mayor o igual que la longitud, la dirección no es válida.
- La dirección física deseada es la suma de la dirección física inicial y el desplazamiento.

Segmentación

En el ejemplo, se parte de la dirección lógica 0001001011110000, que corresponde al segmento número 1, desplazamiento 752. Supóngase que este segmento reside en memoria principal, comenzando en la dirección física inicial 0010000000100000. Por tanto, la dirección física es $0010000000100000 + 001011110000 = 0010001100010000$.

Resumiendo, con la segmentación simple, un proceso se divide en un conjunto de segmentos que no tienen que ser del mismo tamaño. Cuando un proceso se trae a memoria, todos sus segmentos se cargan en regiones de memoria disponibles, y se crea la tabla de segmentos.

Resumen

Una de las tareas más importantes y complejas de un sistema operativo es la gestión de memoria. La gestión de memoria implica tratar la memoria principal como un recurso que debe asignarse y compartirse entre varios procesos activos. Para utilizar el procesador y las utilidades de E/S eficientemente, es deseable mantener tantos procesos en memoria principal como sea posible. Además, es deseable también liberar a los programadores de tener en cuenta las restricciones de tamaño en el desarrollo de los programas.

Las herramientas básicas de gestión de memoria son la paginación y la segmentación. Con la paginación, cada proceso se divide en un conjunto de páginas de tamaño fijo y de un tamaño relativamente pequeño. La segmentación permite el uso de piezas de tamaño variable. Es también posible combinar la segmentación y la paginación en un único esquema de gestión de memoria.

Bibliografía

- Tanenbaum, A. S. (2009). Sistemas operativos modernos (3a ed.) (pp. 1-18). México: Pearson Educación.
- Stallings, W. (2005). Sistemas operativos (5a ed.) (pp. 54-67). Madrid: Pearson Educación.

