Unidad IV

ALMACENAMIENTO REAL

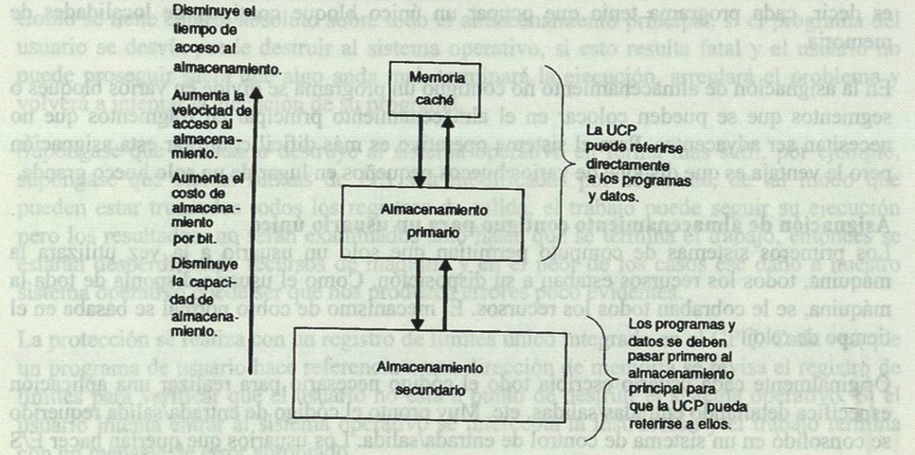
Objetivo de esta unidad.- Durante el desarrollo de esta unidad, se analizarán las técnicas más comunes en las asignaciones y relocalizaciones de programas y datos en memoria principal.

El alumno deberá entender las técnicas para el manejo de memoria real y su aplicación para sistemas operativos de un solo usuario o para sistemas de memoria virtual; deberá comprender los conceptos de páginación, segmentación y memoria asociativa, para entender el funcionamiento de la memoria virtual en un sistema operativo.

Por sencillez consideremos: Memoria = Almacenamiento

Memoria Principal	=	Memoria Real	=	Memoria Primaria

Organización de almacenamiento se entiende que es la forma en que se considera el almacenamiento principal.



Estrategias de Administración de Almacenamiento

1. Estrategias de Obtención
2. Estrategias de Colocación
3. Estrategias de Reemplazo

Estrategias de Obtención.- Determinan cuando debe obtenerse la siguiente parte del programa o los datos que se van a transferir del almacenamiento secundario al principal.

La estrategia convencional es la *obtención por demanda* en la cual la siguiente parte del programa o de los datos se

transfiere al almacenamiento principal cuando un programa en ejecución hace referencia a ella.
Se tiene la creencia de que como generalmente no se puede predecir hacia donde pasara el control de un programa, el trabajo extra para hacer superposiciones y anticipar el futuro superaría por mucho los beneficios esperados aunque se dice que mejorará el rendimiento de los sistemas la *obtención anticipada.*

Estrategias de Colocación.- Tienen que ver con la determinación de la parte del almacenamiento principal donde se colocara un programa entrante (Primer ajuste, Mejor ajuste y Peor ajuste).

Estrategias de Reemplazo.- Están relacionadas con la determinación de qué parte del programa o de los datos se debe desalojar para dejar espacio a los programas entrantes.

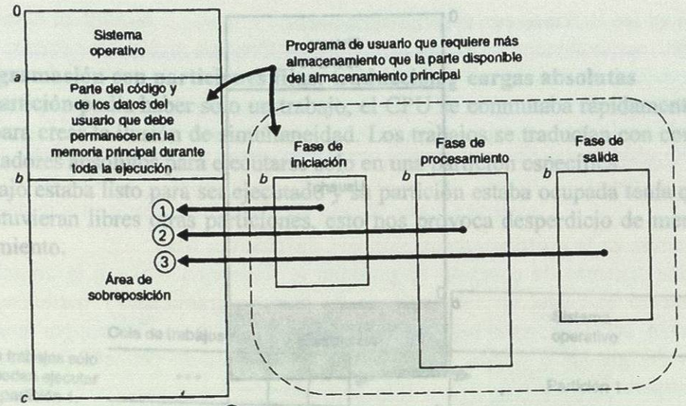
Asignación de almacenamiento contiguo y no contiguo
Los primeros sistemas de computo requerían una asignación de almacenamiento contiguo, es decir, cada programa tenia que ocupar un único bloque contiguo de localidades de memoria.

En la asignación de almacenamiento no contiguo un programa se divide en varios bloques o segmentos que se pueden colocar en el almacenamiento principal en fragmentos que no necesitan ser adyacentes. Para el sistema operativo es más difícil controlar esta asignación pero la ventaja es que dispone de varios huecos pequeños en lugar de un solo hueco grande.

Asignación de almacenamiento contiguo para un usuario único
Los primeros sistemas de computo permitían que solo un usuario a la vez utilizara la máquina, todos los recursos estaban a su disposición. Como el usuario disponía de toda la máquina, se le cobraban todos los recursos. El mecanismo de cobro normal se basaba en el tiempo de reloj.

Originalmente cada usuario escribía todo el código necesario para realizar una aplicación especifica detallando entradas/salidas, etc. Muy pronto el código de entrada/salida requerido se consolidó en un sistema de control de entrada/salida. Los usuarios que querían hacer E/S ya no tenían la necesidad de codificar directamente las instrucciones, sino que llamaban a la rutina de IOCS para que ejecutara el trabajo.

El tamaño de los programas está limitado por la cantidad de memoria principal pero es posible ejecutar programas más grandes utilizando el concepto de superposiciones. Si una sección de un programa no se necesita durante el resto de la ejecución del programa se puede traer del almacenamiento secundario otra sección para ocupar el espacio utilizado por la que ya no se necesite.

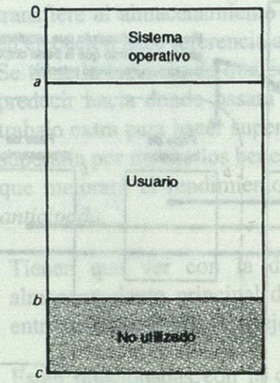


Protección de los sistemas de usuario único
Como se tiene control absoluto sobre todo el almacenamiento principal, si el programa del usuario se desvía puede destruir al sistema operativo, si esto resulta fatal y el usuario no puede proseguir sabrá que algo anda mal, terminará la ejecución, arreglará el problema y volverá a intentar la ejecución de su programa.

Supóngase que el usuario destruye al sistema operativo en forma más sutil, por ejemplo, supóngase que ciertas rutinas de E/S son modificadas por accidente, de tal modo que pueden estar truncados todos los registros de salida, el trabajo puede seguir su ejecución pero los resultados no serán examinados sino hasta que se termina el trabajo, entonces se estarán desperdiciando recursos de máquina y en el peor de los casos ese daño a nuestro sistema operativo puede ser que nos produzca errores poco evidentes.

La protección se realiza con un registro de límites único integrado en el CPU. Cada vez que un programa de usuario hace referencia a una dirección de memoria se revisa el registro de límites para verificar que el usuario no esta a punto de destruir el sistema operativo. Si el usuario intenta entrar al sistema operativo se intercepta la instrucción y el trabajo termina con un mensaje de error apropiado.

El usuario necesita entrar de vez en cuando al sistema operativo para utilizar ciertos servicios, como por ejemplo la E/S, el problema se resuelve dando al usuario una instrucción específica con la cual puede solicitar los servicios del sistema operativo, es decir, una llamada al supervisor.



Procesamiento por lotes de flujo único

Los trabajos en general requerían un tiempo de preparación se cargaba el sistema operativo, se montaban las cintas y los discos, se preparaban impresoras, etc y también requerían de largos tiempos de descarga. El computador estaba ocioso todo ese tiempo. Los diseñadores se dieron cuenta que podían optimizar esos tiempos de transición entre procesos o trabajos pudiendo reducir la cantidad de tiempo que se desperdiciaba en los trabajos, esto condujo al desarrollo de los sistemas de procesamiento por lotes (BATCH).

En el procesamiento por lotes de flujo único, los trabajos se agrupan en lotes cargándolos consecutivamente en cinta o en disco. Un procesador de flujo de trabajos lee las instrucciones en JCL (Job Control Languaje) y facilita la preparación para el siguiente trabajo. Cuando termina un trabajo, el lector de flujo de trabajos lee automáticamente las instrucciones del lenguaje de control para el siguiente trabajo y realiza las acciones de mantenimiento apropiadas para facilitar la transición del siguiente trabajo.

Multiprogramación de particiones fijas

Los diseñadores ven una vez más que se puede aprovechar el CPU notablemente mediante una administración intensiva. Esta vez decidieron administrar sistemas de multiprogramación en los cuales los usuarios compiten al mismo tiempo con los recursos del sistema.

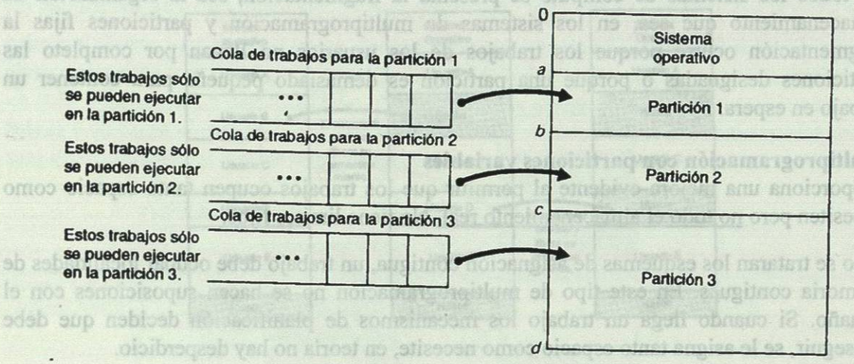
El trabajo que esta esperando la terminación de un proceso de E/S cederá el CPU a otro trabajo que este listo para realizar cálculos si es que hay uno en espera, de este modo pueden efectuarse simultáneamente las operaciones de E/S en los cálculos del CPU.

Para aprovechar al máximo la multiprogramación es necesario que varios trabajos residan al mismo tiempo en el almacenamiento principal del computador, de este modo cuando un trabajo solicita E/S el CPU puede conmutarse a otro y realizar cálculos sin retraso.

Multiprogramación con particiones fijas: traducción y cargas absolutas

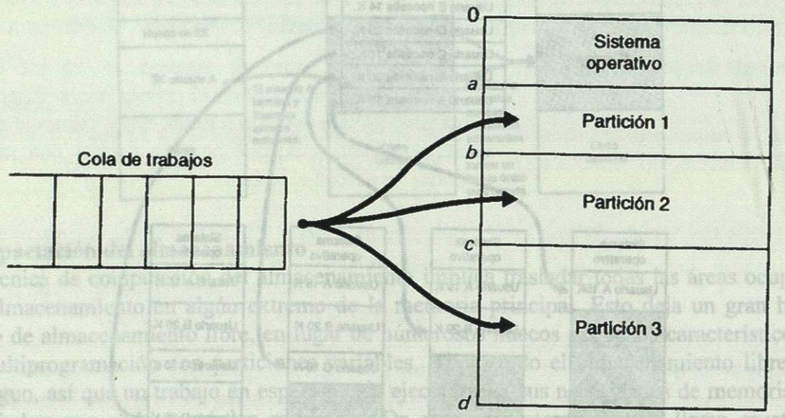
En cada partición podía haber solo un trabajo, el CPU se conmutaba rápidamente entre los usuarios para crear la ilusión de simultaneidad. Los trabajos se traducían con compiladores y ensambladores absolutos para ejecutarse solo en una partición específica.

Si un trabajo estaba listo para ser ejecutado y su partición estaba ocupada tenía que esperar aunque estuvieran libres otras particiones, esto nos provoca desperdicio de memoria o de almacenamiento.



Multiprogramación con particiones fijas: traducción y carga con reubicación

Los compiladores, ensambladores y cargadores con reubicación sirven para producir programas que se puedan ejecutar en cualquier partición disponible que sea lo suficientemente grande para contenerlos.



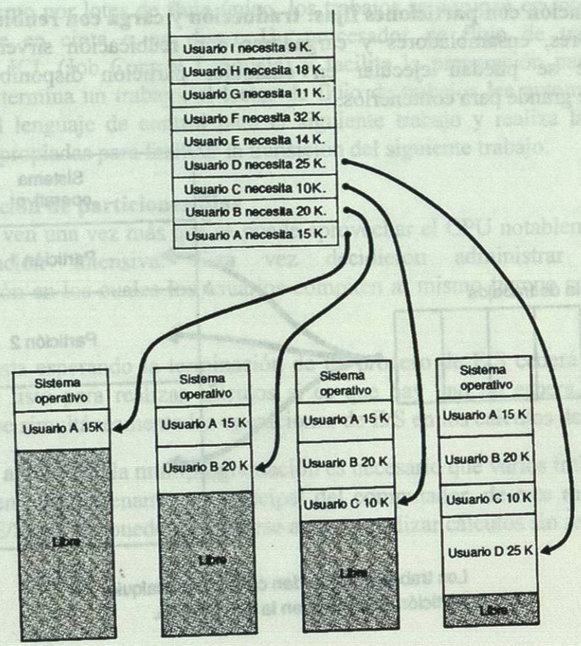
Los trabajos se pueden colocar en cualquier partición disponible en la que quepan.

Protección en los sistemas con multiprogramación
Esto se logra a menudo con varios registros límites. Con dos registros se pueden establecer los límites superior e inferior de la partición de un usuario o bien el límite superior o el inferior y la longitud de la región. El usuario que necesita llamar al sistema operativo utiliza una instrucción de llamada al supervisor para hacerlo, esto permite al usuario cruzar el limite del sistema operativo y solicitar sus servicios sin poner en peligro la seguridad del sistema operativo.

Fragmentación en la multiprogramación con particiones fijas
En todos los sistemas de computo se presenta la fragmentación, sea la organización de almacenamiento que sea, en los sistemas de multiprogramación y particiones fijas la fragmentación ocurre porque los trabajos de los usuarios no llenan por completo las particiones designadas o porque una partición es demasiado pequeña para contener un trabajo en espera.

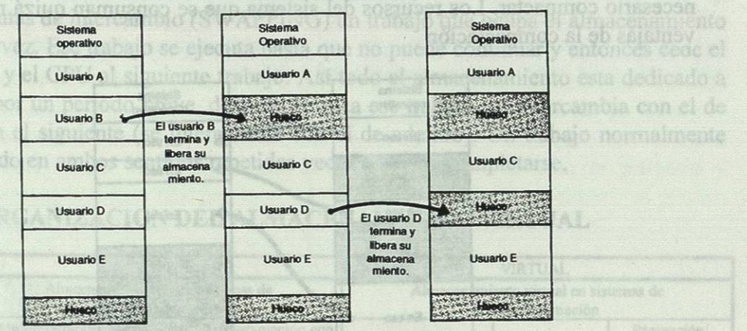
Multiprogramación con particiones variables
Proporciona una mejora evidente al permitir que los trabajos ocupen tanto espacio como necesiten pero no todo el almacenamiento real. No tiene limites fijos.

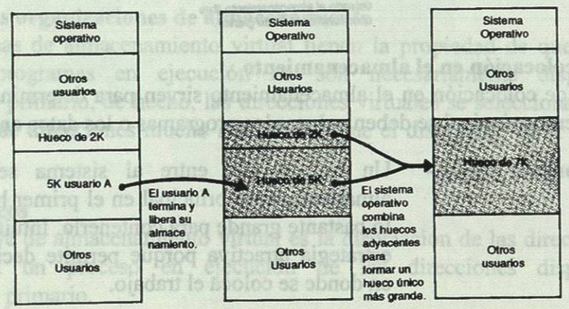
Solo se trataran los esquemas de asignación contigua, un trabajo debe ocupar localidades de memoria contiguas. En este tipo de multiprogramación no se hacen suposiciones con el tamaño. Si cuando llega un trabajo los mecanismos de planificación deciden que debe proseguir, se le asigna tanto espacio como necesite, en teoría no hay desperdicio.



Condensación de huecos
Cuando termina un trabajo en un sistema de multiprogramación con particiones variables se puede comprobar si el almacenamiento liberado colinda con otras áreas libres de almacenamiento, comúnmente denominados huecos.

El proceso de fusionar huecos adyacentes para formar un solo hueco más grande se le denomina condensación. Mediante la condensación de huecos se puede recuperar los bloques contiguos de almacenamiento más grande que sea posible.





Compactación del almacenamiento
La técnica de computación del almacenamiento implica trasladar todas las áreas ocupadas del almacenamiento en algún extremo de la memoria principal. Esto deja un gran hueco único de almacenamiento libre, en lugar de números huecos pequeños característicos de la multiprogramación con particiones variables. Ahora todo el almacenamiento libre esta contiguo, así que un trabajo en espera puede ejecutarse si sus necesidades de memoria son satisfechas por el hueco único resultante. De manera pintoresca se le llama eructo de almacenamiento (Burping Storage) o recolección de basura (Garbage Collection).