

11 IMPLEMENTACIÓN DEL SISTEMA DE ARCHIVOS

- 1 ESTRUCTURA DEL SISTEMA DE ARCHIVOS
- 2 MÉTODOS DE ASIGNACIÓN
- 3 ADMINISTRACIÓN DEL ESPACIO LIBRE
- 4 IMPLEMENTACIÓN DE DIRECTORIOS
- 5 EFICIENCIA Y DESEMPEÑO
- 6 RECUPERACIÓN

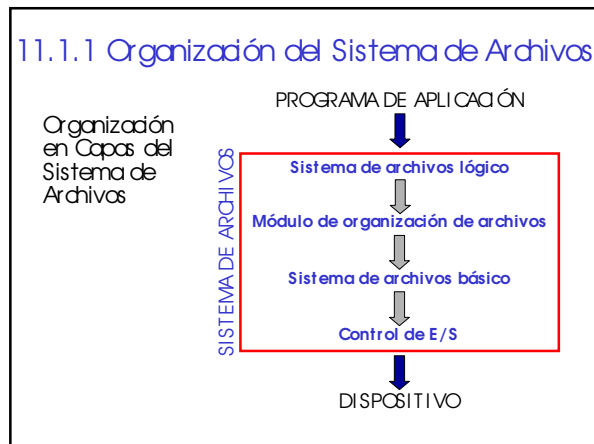
11.1 ESTRUCTURA DEL SISTEMA DE ARCHIVOS

- 1 Organización del Sistema de Archivos
- 2 Carga del Sistema de Archivos

11.1 ESTRUCTURA DEL SISTEMA DE ARCHIVOS

- Estructura de un archivo
 - Unidad de almacenamiento lógica
 - Colección de información relacionada
- El file system reside en el almacenamiento secundario.
- Está organizado en capas (layers).
- El **Bloque de Control de Archivo** (file control block o FCB) es una estructura de almacenamiento que contiene la información acerca de un archivo.

11.1.1 Organización del Sistema de Archivos



11.1.1 Organización del Sistema de Archivos

Organización en Capas del Sistema de Archivos:

- **Dispositivo:** Qué posición del dispositivo debe actuar y qué acciones debe emprender
- **Sistema de archivos básico:** Emitir órdenes genéricas al dispositivo para leer y escribir bloques físicos en disco
- **Módulo de organización de archivos:** Traduce las direcciones de bloques lógicos a bloques físicos.
- **Sistema de archivos lógico:** Emplea la estructura de directorios para proporcionar info. Al módulo de org.

11.1.1 Organización del Sistema de Archivos

Ejemplo de Tabla de Archivos

Índice	Nombre de Archivo	Permisos	Fechas de Acceso	Puntero a bloque de Disco
0	PRUEBA.C	rw rw rw	...	-->
1	CORREO.TXT	rw	...	-->
2				-->
.				
.				
.				
n				

11.1.2 Carga del Sistema de Archivos

- Es necesaria para que éste quede disponible para los procesos del sistema.
- Generalmente, el sistema es cargado en cada dispositivo.
- Permite al Sistema Operativo recorrer la estructura de directorios.

11.2 MÉTODOS DE ASIGNACIÓN

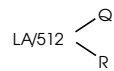
- 1 Asignación Contigua
- 2 Asignación Enlazada
- 3 Asignación Indexada
- 4 Desempeño

11.2.1 Asignación Contigua

- Cada archivo ocupa un conjunto de bloques contiguos en el disco.
- Es simple, solo se requieren la posición de comienzo (nº de bloque) y la longitud (número de bloques).
- Permite acceso directo (random).
- No aprovecha bien el espacio (problema de asignación dinámica).
- Los archivos no pueden crecer. De lo contrario, hay que cambiarlos de posición $\Rightarrow$ fragmentación externa.

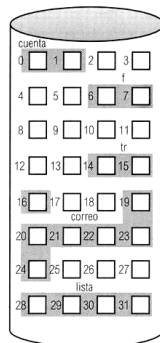
11.2.1 Asignación Contigua

- Mapeo lógico → físico.
- Bloque a ser accedido: Q + dirección de comienzo
- Desplazamiento dentro del bloque = R



11.2.1 Asignación Contigua

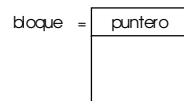
Ejemplo de
Asignación
Contigua



directorio		
archivo	inicio	longitud
cuenta	0	2
tr	14	3
correo	19	6
lista	28	4
f	6	2

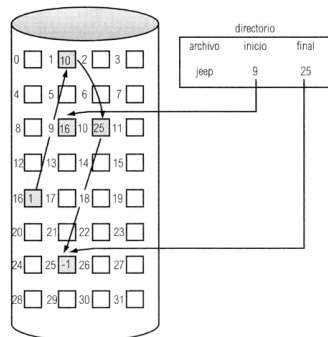
11.2.2 Asignación Enlazada

- Cada archivo es una lista enlazada de bloques de disco; los bloques pueden estar esparcidos en el disco.



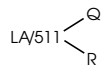
11.2.2 Asignación Enlazada

Se asigna a medida que se necesita y luego se enlaza; ej, el archivo **jeep** comienza en el bloque 9 y termina en el 25.



11.2.2 Asignación Enlazada

- Simple, solo se necesita la dirección de comienzo
- La administración del espacio es eficiente
- No permite acceso directo
- Mapeo :
 - El bloque a ser accedido es el Qésimo en la cadena
 - Desplazamiento dentro del bloque = $R + 1$



11.2.2 Asignación Enlazada

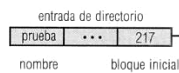
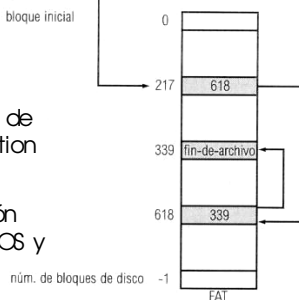


Tabla de Asignación de Archivos (File Allocation Table o **FAT**)

Método de asignación empleado por MS-DOS y OS/2.



11.2.3 Asignación Indexada

- Junta todos los punteros en un bloque de índices

- Visión lógica:

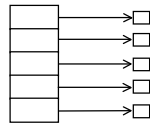
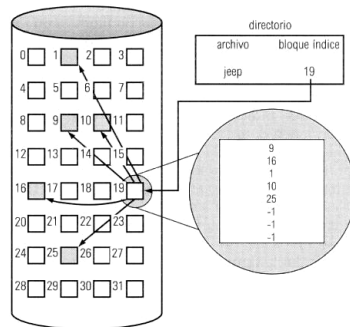


Tabla de índices

11.2.3 Asignación Indexada

Ejemplo de
Asignación
Indexada



11.2.3 Asignación Indexada

MAPEO EN LA ASIGNACIÓN INDEXADA

- Necesita una tabla de índices
- Permite acceso directo
- Permite acceso dinámico sin fragmentación externa pero genera overhead de bloque de índices.

11.2.3 Asignación Indexada

- Para mapear un archivo de tamaño máximo de 256 K palabras y tamaño de bloque de 512 palabras, sólo se necesita 1 bloque para la tabla de índices

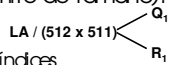


- Q = desplazamiento dentro de la tabla de índices
- R = desplazamiento dentro del bloque

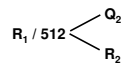
11.2.3 Asignación Indexada

Para mapear un archivo de tamaño ilimitado con tamaño de bloque de 512 palabras.

- 1) **Esquema enlazado:** Enlazar los bloques de la tabla de índices (sin límite de tamaño).



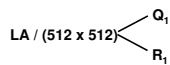
- Q_1 : bloque de la tabla de índices
- R_1 se usa de la siguiente manera

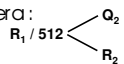


- Q_2 : desplazamiento dentro del bloque de la tabla de índices.
- R_2 : desplazamiento dentro del bloque de archivo

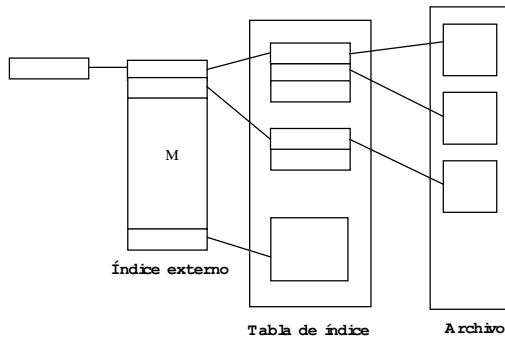
11.2.3 Asignación Indexada

- 2) **Índice de 2 niveles** (tamaño máximo del archivo 512^3)

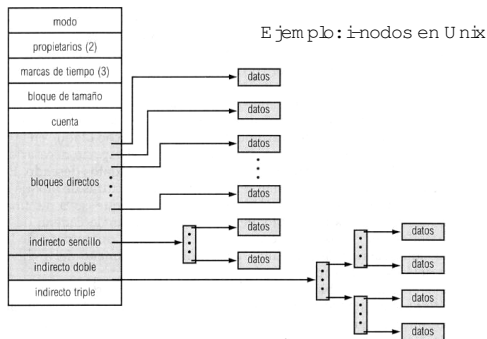


- Q_1 : desplazamiento dentro del índice externo
 - R_1 se usa de la siguiente manera:
- 
- A diagram showing a horizontal line labeled $R_1 / 512$ on the left. A diagonal line extends from this point upwards and to the right, ending at a point labeled Q_2 . Another diagonal line extends from the same point on the horizontal line downwards and to the right, ending at a point labeled R_2 .
- Q_2 : desplazamiento dentro del bloque de la tabla de índices
 - R_2 : desplazamiento dentro del bloque de archivo

11.2.3 Asignación Indexada



11.2.3 Asignación Indexada



11.2.4 Desempeño

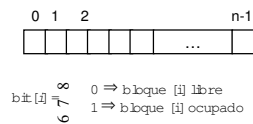
- **Desempeño:**
 - en velocidad: contigua
 - en aprovechamiento del espacio: enlazada e indexada
- **Modo de Acceso:**
 - Contigua: Secuencial y Directo
 - Enlazada: Secuencial
 - Indexada: Secuencial y Directo
- **Complejidad:**
 - Contigua: Baja
 - Enlazada: Media
 - Indexada: Alta

11.3 ADMINISTRAC DEL ESPACIO LIBRE

- 1 Vector de Bits
- 2 Lista Enlazada
- 3 Agrupamiento (Clustering)
- 4 Contabilización

11.3.1 Vector de Bits

- Vector de bits (n bloques)



- Cálculo del número de bloque:
(nº de bits por palabra) *
(nº de palabras en 0) +
desplazamiento del primer bit en 1

11.3.1 Vector de Bits

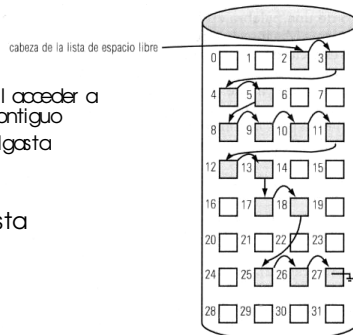
- El mapa de bits requiere espacio extra, ejemplo:
Tamaño de bloque = 2^{12} byte
Tamaño del disco = 2^{30} bytes (1 Gbyte)
 $n = 2^{30}/2^{12} = 2^{18}$ bits (o 32 K byte)
- Es fácil acceder archivos contiguos

11.3.2 Lista Enlazada

- Lista libre

- No es fácil acceder a espacio contiguo
- No se malgasta espacio

Ejemplo de Lista Enlazada



11.3.2 Lista Enlazada

- Es necesario proteger :

- Puntero a la lista libre
- Mapa de bits
 - Debe ser mantenido en disco
 - La copia en memoria y disco pueden diferir
 - No se puede permitir que para el bloque (i) el bit (i)= 1 en memoria y bit (i)= 0 en disco.
- Solución:
 - Poner bit (i)= 1 en disco
 - Asignar bloque (i)
 - Poner bit (i)= 1 en memoria

11.3.3 Agrupamiento (Clustering)

- Modificación de Lista Enlazada.
- Se agrupan direcciones de bloques libres en el primer bloque libre.
- De esas, los n-1 primeras están efectivamente libres.
- El n-ésimo contiene la dirección del siguiente bloque, que a su vez contiene las direcciones de bloques libres y así sucesivamente.

11.3.4 Contabilización

- En vez de mantener las direcciones individuales, para cada grupo de bloques contiguos se mantiene la dirección del primero y luego un contador de bloques consecutivos.
- La lista es más corta, pero requiere más espacio por nodo.

11.4 IMPLEMENTACIÓN DE DIRECTORIOS

- 1 Lista Lined
- 2 Tabla de Dispersión (Hashing)

11.4.1 Lista Lined

- Lista de nombres de archivo, con punteros hacia los bloques de datos.
 - Simple de programar
 - Lento de ejecutar

11.4.2 Tabla de Dispersión (Hashing)

- Lista lineal con estructura de datos dispersos (hash).
 - Disminuye el tiempo de búsqueda en el directorio
 - Colisiones: dos nombres de archivo apuntan a la misma posición
 - Áreas muertas: posiciones que nunca son accedidas
 - Tamaño fijo

11.5 EFICIENCIA Y DESEMPEÑO

- 1 Eficiencia
- 2 Desempeño

11.5.1 Eficiencia

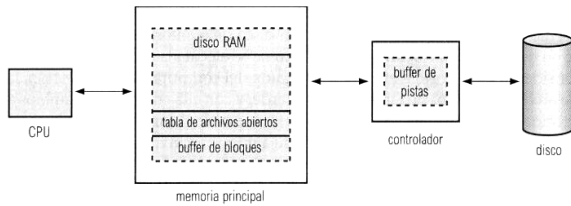
- La Eficiencia depende de:
 - Algoritmos de asignación de disco y manejo de directorio
 - Tipo de dato almacenado en la entrada al directorio de cada archivo (punteros)

11.5.2 Desempeño

- Cache de disco: secciones separadas de la memoria principal para bloques frecuentemente usados
- Técnicas free-behind (liberación atrás: elimina un bloque del buffer tan pronto como se solicita el siguiente bloque, los anteriores son poco usados) y read-ahead (lectura adelantada: se leen y colocan en cache el bloque solicitado y los subsiguientes) para optimizar accesos secuenciales

11.5.2 Desempeño

Mejoramiento del rendimiento de un PC, dedicando secciones de memoria como disco virtual o RAM disk (bajo control del usuario), para hacer cache de disco (bajo control del S.O)



11.6 RECUPERACIÓN

- 1 Verificación de Consistencia
- 2 Respaldo y Restauración

11.6.1 Verificación de Consistencia

- Compara datos en la estructura del directorio con los bloques de datos en disco y trata de reparar las inconsistencias.

11.6.2 Respaldo y Restauración

- Usar programas del sistema para respaldar (back up) los datos del disco en otro dispositivo de almacenamiento (diskette, cinta).
- Recuperar archivos o discos perdidos restaurando los datos desde el respaldo.
