

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

5.1 ARCHIVOS

El S.O. da una visión uniforme para todos los sistemas de almacenamiento, definiendo una unidad lógica de almacenamiento denominada **archivo**.

Se considera como archivo a un conjunto de información relacionada definida por su creador. Corresponden a los programas y los datos de cualquier tipo. En general un archivo es una serie de bits, bytes o registros.

Los archivos son nombrados y referenciados por su nombre.

Además, tienen otras propiedades o **atributos** como su tipo, la fecha y la hora de creación, identificador del creador, longitud, etc.

5.1.1 TIPOS Y ESTRUCTURAS DE ARCHIVOS

Los archivos pueden contener cualquier tipo de información y dependiendo de su uso tendrán una determinada estructura.

Casos posibles :

- a) El S.O. puede tener conocimiento de estas distintas estructuras lógicas y considerar diferentes tipos de archivos.
Desventajas :
 - 1) Mayor tamaño del S.O.
 - 2) Gran rigidez ya que sólo se pueden considerar los tipos de archivos definidos por el sistema.
- b) En S.O.'s como Unix no se definen los tipos de archivos, con lo que se tiene una gran flexibilidad, pero con un soporte mínimo, de forma que los programas de aplicación tienen que incluir el código que interprete la estructura adecuada de los archivos.
El S.O. sólo empaqueta y desempaqueta los bytes cuando los escribe y los lee del disco de acuerdo a los bloques físicos del mismo.
- c) Considerar un archivo como una secuencia de registros lógicos (en desuso), cada uno con su propia estructura interna. La visión de Unix y MS-DOS es una particularización de ésta, reduciéndose el registro lógico a un byte.
- d) Estructuras de árbol (restringido a grandes computadores corporativos) . Los registros tienen una clave y la operación básica no es obtener el siguiente registro, sino obtener el registro con la clave especificada.

El acceso original a los archivos (heredado de las cintas), era secuencial. Una operación de lectura leía el registro actual y automáticamente avanzaba al siguiente registro del archivo. La operación de escritura añadía el registro al final nuevo final del archivo.

Con la aparición de los discos se pudo acceder directamente a cualquier registro del archivo, dando lugar al acceso arbitrario. Las operaciones de lectura y escritura deben de incluir el número de registro como parámetro.

En la actualidad todos los archivos se consideran de acceso arbitrario.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

5.1.2 OPERACIONES CON LOS ARCHIVOS

CREATE (Crear)

```
Llamada: CREATE (nombre_archivo, atributos)

/* buscar nombre del archivo en el directorio */

/* si se localiza, enviar mensaje de duplicado, o crear nueva versión y sobre grabar */

/* localizar una entrada libre del directorio */

/* si no hay ninguna asignarla */

/* asignar espacio para el archivo */

/* registrar bloques asignados en el directorio */

/* registrar los atributos del archivo en el directorio */
```

Realiza las siguientes acciones :

- 1) Buscar si ya existe ese archivo.
- 2) Asignarle una entrada en el directorio.
- 3) Asignarle espacio en disco.
- 4) Registrar sus atributos en el directorio.

En algunos sistemas la propia función de crear el archivo lleva implícita la apertura del mismo, haciendo una llamada a OPEN.

OPEN (Abrir)

```
Llamada: ID_conexión = OPEN (nombre_archivo, modo_acceso)

/* buscar nombre del archivo en el directorio */

/* si no se localiza, enviar mensaje de error y volver */

/* verificar permiso de acceso al archivo, si no hay permiso, indicar error */

/* crear un bloque de control de archivo copiando a la memoria principal la lista de atributos y direcciones */

/* crear identificador ID_conexión */

/* inicializar el índice del archivo */

/* devolver el identificador de conexión ID_conexión */
```

Establece un enlace entre el programa y el archivo, trasladando los atributos y la lista de direcciones desde el disco a la memoria principal.

SEEK (Buscar)

```
Llamada: SEEK (ID_conexión, posición_lógica)

/* verificar identificador de conexión ID_conexión */

/* calcular posición adecuada */

/* actualizar el marcador de archivo */
```

Se cambia la posición del apuntador para señalar al byte o registro cuya dirección lógica se suministra en la llamada.

READ (Leer)

```
Llamada: estatus = READ (ID_conexión, num_bytes, in_buffer)

/* verificar ID_conexión */

/* verificar archivo abierto para lectura: autorización de acceso */

/* sincronizar con otros usuarios activos, si hace falta: compartimiento */

/* calcular número y direcciones de los sectores a leer: correlación */

/* emitir orden de lectura al controlador del dispositivo */

/* verificar resultado de lectura del dispositivo */

/* copiar num_bytes de datos desde el almacenamiento intermedio al in_buffer */

/* actualizar marcador del archivo */

/* devolver estatus */
```

La lectura se hace en la posición actual y en la llamada se tiene que especificar la cantidad de datos necesarios y proporcionar un búfer para su transmisión. El resultado de la operación se indica mediante el valor que se devuelve en status.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

WRITE (Escribir)

```
Llamada: estatus = WRITE (ID_conexión, num_bytes, out_buffer)

/* verificar ID_conexión */

/* verificar archivo abierto para escritura: autorización de acceso */

/* sincronizar con otros usuarios activos, si hace falta: compartimiento */

/* si se amplía el archivo, asignar los bloques requeridos: asignación de espacio */

/* actualizar directorio si se añaden nuevos bloques */

/* calcular número y direcciones de los sectores a escribir: correlación */

/* copiar num_bytes de datos desde el out_buffer al almacenamiento intermedio */

/* emitir orden de escritura al controlador del dispositivo */

/* verificar resultado de escritura del dispositivo */

/* actualizar marcador del archivo */

/* devolver estatus */
```

Escribe en un archivo nuevo o añade a uno existente.

En este caso, el sistema de archivos llama al módulo de asignación de espacio para proporcionar el número de bloques libres requerido.

Los bloques del directorio y los índices de los archivos serán modificados en consecuencia.

Algunos sistemas tienen una forma restringida de write, con la que solo se puede añadir al final del archivo (APPEND)

CLOSE (Cerrar)

```
Llamada: CLOSE (ID_conexión)

/* verificar ID_conexión */

/* si no se localiza, enviar mensaje de error y volver */

/* verificar permiso de acceso al archivo, si no hay permiso, indicar error */

/* si la copia de la entrada del directorio guardada en el bloque de control del archivo ha sido actualizada, entonces reescribirla en el disco */

/* liberar el bloque de control de archivo */

/* borrar identificador */
```

Se libera la tabla con la lista de atributos y direcciones que se mantenía en memoria, actualizándola en disco si se ha modificado.

En muchos sistemas, la terminación obligada o voluntaria de procesos incluye el cierre de todos los archivos abiertos, incluyendo en las rutinas EXIT o ABORT una llamada a CLOSE.

DELETE (Borrar)

```
Llamada: DELETE (nombre_archivo)

/* buscar nombre del archivo en el directorio */

/* si no se localiza, enviar mensaje de archivo no encontrado */

/* verificar permisos, si no hay permiso de acceso, indicar error */

/* verificar si se está utilizando, si está abierto enviar mensaje de archivo abierto */

/* liberar la entrada en el directorio */

/* liberar el espacio asignado al archivo */
```

Elimina un archivo y libera el espacio que ocupa, borrando del directorio la entrada al archivo y liberando el espacio asignado al mismo.

Hay S.O. que como medida de seguridad, mantienen una copia de los archivos borrados hasta que reciben la orden expresa de liberar ese espacio.

RENAME (Renombrar)

Cambia el nombre del archivo en la tabla del directorio correspondiente. Se puede suplir copiando el archivo original con otro nombre y borrando en original.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

COPYFILE (Copiar)

```
Llamada: COPYFILE (nombre_archivo_orig, nombre_archivo_destino)

/* abrir el archivo origen nombre_archivo_orig */

/* si no se localiza, enviar mensaje de archivo no existente */

/* crear archivo destino nombre_archivo_destino, con permisos de escritura */

/* si no se puede crear, ya existe, enviar mensaje de archivo ya existente */

/* abrir el archivo destino */

/* Realizar la copia. Leer bloques del archivo origen nombre_archivo_orig y copiarlos en el
archivo destino nombre_archivo_destino */

/* cerrar el archivo origen nombre_archivo_orig */

/* cerrar el archivo destino nombre_archivo_destino */
```

Lleva implícita la creación de un nuevo archivo, aunque en algunas versiones se permite copiar en un archivo ya existente grabando sobre la información ya existente.

LECTURA Y MODIFICACIÓN DE ATRIBUTOS

Muchos sistemas permiten la lectura, y en algunos casos la modificación de algunos atributos.

5.2 DIRECTORIOS DE ARCHIVOS

DEFINICIÓN

Son tablas simbólicas de archivos. Una entrada típica de directorio puede contener :

- Nombre, tipo y número de versión del archivo.
- Puntero de acceso al archivo, dirección de comienzo en el disco.
- Lista de atributos.

Nombre	Tipo	Versión	Puntero	Tamaño	Permisos	Fechas
libro	txt	3	30125	50	-rw-r--r--	12/12/94 10/01/95
ordena	obj	1	25432	120	-rw-rw-rw-	05/11/94 22/01/95
ordena	exe	1	10125	30	-rwxrwx-rw-	01/01/95 01/10/95
lista	txt	2	65390	40	-rw-r--r--	10/10/94 05/12/94
libmat	lib	4	00912	90	-rw-rw-rw-	01/03/94 05/08/94

En muchos sistemas, la tabla del directorio está dividida en dos. En una se mantienen los nombres de los archivos con un número de identificación, el cual da acceso a otra donde se tiene el puntero de acceso al archivo y la lista de atributos. Esto agiliza la gestión de los enlaces, la generación de alias y homónimos.

Nombre	ID
libro	1
ordena	2
ordena	3
lista	4
libmat	5

ID	Tipo	Versión	Puntero	Tamaño	Permisos	Fechas
1	txt	3	30125	50	-rw-r--r--	12/12/94 10/01/95
2	obj	1	25432	120	-rw-rw-rw-	05/11/94 22/01/95
3	exe	1	10125	30	-rwxrwx-rw-	01/01/95 01/10/95
4	txt	2	65390	40	-rw-r--r--	10/10/94 05/12/94
5	lib	4	00912	90	-rw-rw-rw-	01/03/94 05/08/94

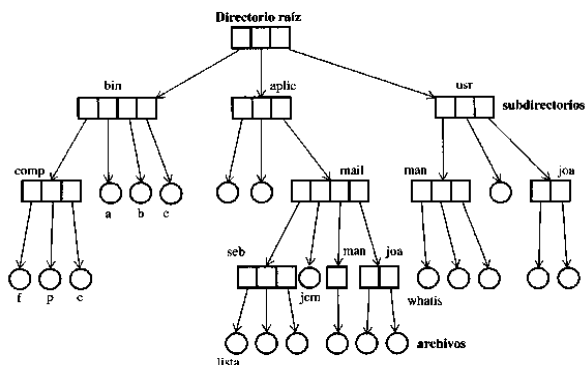
TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

El número de directorios varía de un sistema a otro.

Casos :

- Directorio de nivel único.** Un único directorio almacena todos los archivos del sistema. Sistema usado en los primeros microcomputadores.
Inconvenientes: Especial cuidado en no duplicar el nombre de archivos, sobre todo en entornos multiusuario. Problemas para asignar protección a los archivos.
- Directorio por usuario.** Se elimina el problema a la hora de crear nombres de archivos entre distintos usuarios, pero el problema persiste cuando un usuario tiene muchos archivos.
- Árbol de directorios.** Las entradas del directorio correspondiente tienen un atributo más que indica si corresponde a un archivo o a un subdirectorio. Jerarquización.



Normalmente, cada usuario tiene su “directorio inicial” indicado en un archivo de “cuentas”.

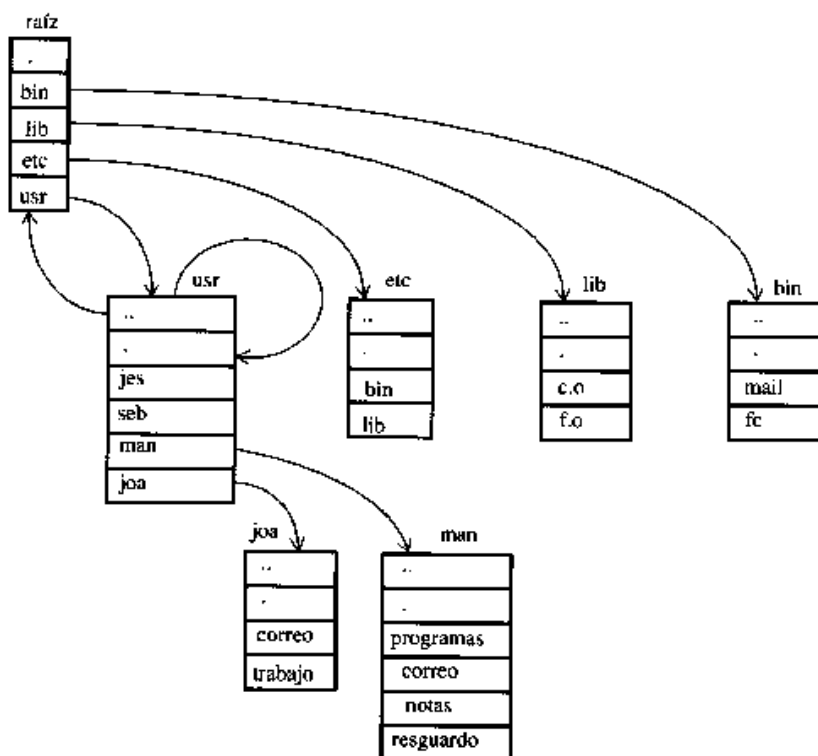
Cuando se hace referencia a un archivo o directorio, se busca en el directorio actual. Si no se encuentra el usuario debe cambiar de directorio o indicar el camino completo.

Los nombres de los caminos pueden ser completos o relativos.

Normalmente los S.O. tienen dos entradas especiales para cada directorio:

- “.” Para el propio directorio. Con un puntero a si mismo.
- “..” Para el directorio padre.

Estas pueden ser usadas en los caminos relativos.



Estructura típica de directorios en un sistema Unix.

Si el directorio actual es /usr/man, copiar el archivo notas de este directorio en el directorio /usr/joa se puede hacer de las siguientes formas:

- `cp notas ../joa/notas`
Se indica el archivo origen y destino con caminos relativos al directorio actual.
- `cp notas /usr/joa/notas`
Se usa un camino relativo para el directorio actual y absoluto para el destino.
- `cp /usr/man/notas /usr/joa/notas`
Se usan camino absolutos para los dos.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

¿DÓNDE Y COMO UBICAR LOS DIRECTORIOS ?

Como los directorios suelen ser muy grandes, se guardan en disco. Esto facilita montar y desmontar dispositivos.

Es habitual considerar los directorios como archivos que tienen una lista de todos los archivos. Para localizar el directorio raíz al arrancar el sistema, éste se sitúa en una dirección conocida por el volumen desde el que se arranca el sistema.

5.2.1 OPERACIONES CON DIRECTORIOS

MAKEDIR (Crear directorio)

En el directorio actual, se crea una entrada para un nuevo subdirectorio vacío, salvo las entradas “.” y “..”.

REMOVEDIR (Borrar directorio)

Elimina un directorio vacío. Si no lo está, se pueden hacer dos cosas:

- 1) No permitir borrarlo.
- 2) Suponer que se quiere borrar todos los archivos y subdirectorios que contiene.

OPENDIR (Abrir directorio) y CLOSEDIR (Cerrar directorio)

Al abrirlo se copian sus tablas en memoria.

Al cerrarlo se actualizan en el disco.

READDIR (Leer directorio)

Devuelve en un formato estándar la entrada actual del directorio. Por defecto el actual.

RENAMEDIR (Cambiar de nombre)

LINK (Enlazar)

Forma de hacer que un archivo o subdirectorio aparezca en varios directorios.

En la llamada se especifica el nombre del archivo y el camino de acceso, creándose un enlace entre este camino y el archivo ya existente.

La estructura de directorio es en este caso un grafo dirigido acíclico.

UNLINK (Desenlazar)

Eliminación del enlace y borrado de la correspondiente entrada en el directorio.

TEMA 5

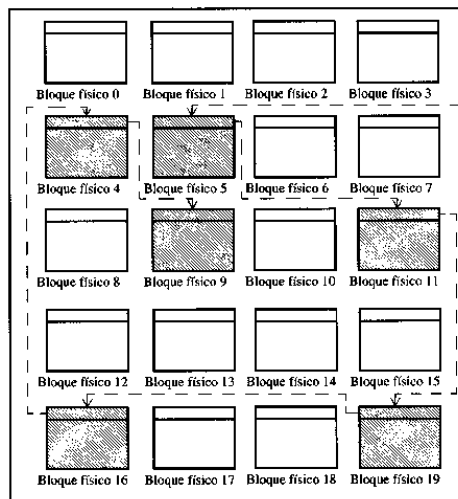
GESTIÓN DEL SISTEMA DE ARCHIVOS

5.3 REALIZACIÓN DEL SISTEMA DE ARCHIVOS

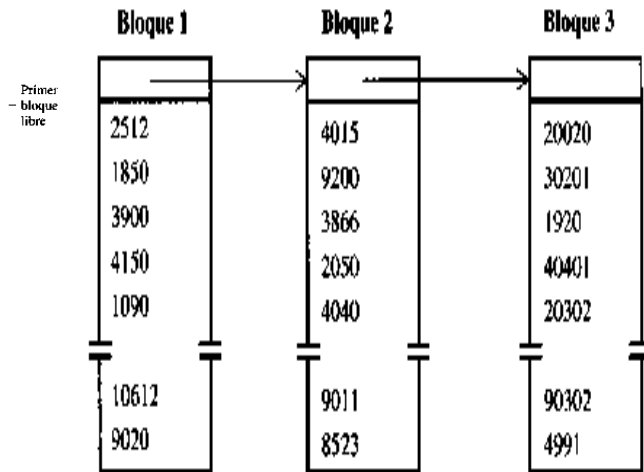
5.3.1 GESTIÓN DEL ESPACIO LIBRE

Para mantener una lista del espacio libre en disco se usan dos métodos :

- Mapa de bits** de los bloques libres. Un disco con n bloques necesitará un mapa de bits con n bits. Los bloques libres se representan con un 1 y los ocupados con un 0, o viceversa. Por ejemplo, en un disco de 300Mb con bloques de 1Kb necesitará un mapa de bits de 300Kb, que se almacena en 38 bloques de 1Kb.
- Lista enlazada** de bloques libres, manteniendo un puntero al primer bloque libre. No es muy eficiente, puesto que para leer la lista hay que leer todos los bloques. Como mejora, se puede utilizar una lista de bloques en la que cada bloque contiene números de bloques libres.



Lista enlazada de bloques libres



Mejora al método de la lista enlazada

El mapa de bits ocupa menos espacio, ya que sólo usa un 1 bit por cada bloque, en vez de los 19 que necesita el método de la lista enlazada. Si existe suficiente memoria para mantener el mapa de bits, este método es preferible. Aunque cuando el disco está casi lleno podría ser mas rentable el método de las listas enlazadas.

La elección del tamaño de los bloques requiere realizar un estudio previo de cuan grandes en promedio van a ser los archivos en ese sistema. Lo habitual es 512bytes, 1Kb., 2Kb.

5.3.2 MÉTODO DE ASIGNACIÓN CONTIGUA

Requiere que cada archivo ocupe un conjunto de direcciones contiguas en el disco. Las entradas en los directorios indicarán el bloque de comienzo y la longitud.

La dificultad está en asignarle el espacio correcto cuando se crea el archivo. Las soluciones son del tipo primero en ajustarse o mejor en ajustarse.

Ventaja :

- Buen rendimiento cuando se quiere leer un archivo completo.

Inconvenientes :

- Debe conocerse el tamaño máximo del archivo en el momento de su creación.
- Si al crecer el espacio asignado no es suficiente, se tiene que reubicar.
- Produce fragmentación externa. Puede solucionarse con compactación pero es una solución muy costosa.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

5.3.3 MÉTODO DE ASIGNACIÓN MEDIANTE LISTAS ENLAZADAS

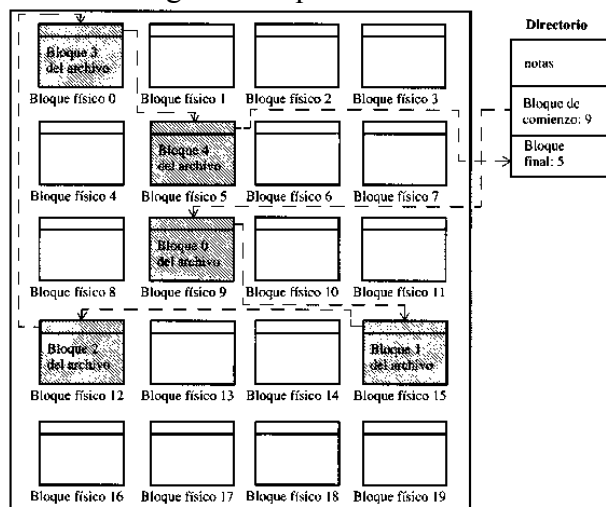
Unos pocos bytes del comienzo de los bloques se usan como puntero al siguiente bloque. Las entradas en el directorio sólo tienen que guardar un puntero al primer bloque del disco asignado al archivo. Escribir en el archivo supone coger uno de los bloques libres y añadirlo al final de la lista. Para facilitar esta operación, también suele haber un puntero al último bloque del archivo. Para leer un archivo sólo hay que seguir los punteros bloque a bloque.

Ventajas:

- a) No causa fragmentación externa.
- b) No se tiene que declarar el tamaño del archivo cuando se crea.
- c) Los archivos pueden crecer sin ningún problema mientras haya bloques libres.

Inconvenientes:

- a) El acceso aleatorio a un archivo es muy lento, puesto que hay que seguir los punteros hasta llegar al bloque deseado.



5.3.4. MÉTODO DE ASIGNACIÓN MEDIANTE INDEXACIÓN

Se crea una tabla de índices en la que se colocan los índices a los bloques de los archivos.

El directorio contiene la dirección del bloque donde están los índices a los bloques de datos (concepto análogo a las tablas de páginas de memoria).

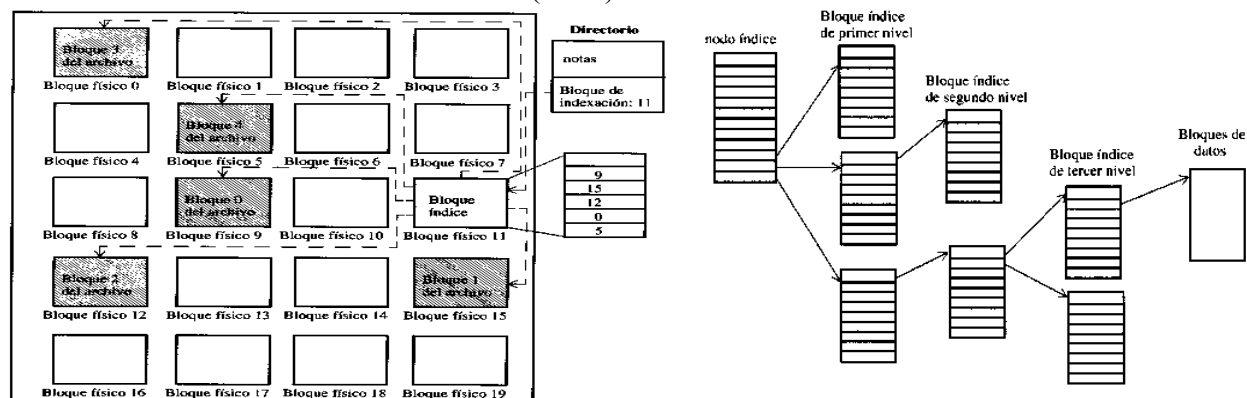
Ventajas :

- a) Todo el bloque está disponible para los datos.
- b) Se soporta con la misma eficacia el acceso aleatorio que el secuencial.
- c) No sufre fragmentación externa.

Inconvenientes :

- a) Pérdida de espacio en el bloque de índices para índices pequeños.

Para archivos grandes se necesitaría asignar más bloques a la tabla de índices, pudiéndose utilizar sistemas de indexación de varios niveles (Unix).



TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

5.3.5 ALGUNOS ASPECTOS DE DISEÑO

a) Problemas de diseño :

- 1) Definir como verá el usuario el sistema de archivos. La estructura de directorios, los archivos y sus atributos.
- 2) Crear los algoritmos y las estructuras de datos que relacionan el sistema de archivos lógicos con el dispositivo físico de almacenamiento.

b) Niveles :

El sistema de archivos se puede considerar compuesto por varios niveles o capas. Cada nivel en el diseño usa las posibilidades de los niveles inferiores para crear nuevas características que usarán los niveles superiores.

- (0) **Dispositivos físicos de control** de las transferencias de información entre la memoria y el disco.
- (1) **Sistema básico de archivos**, el cual usa las características del nivel anterior para leer y escribir bloques en el disco. La indicación de los bloques que tiene que leer y escribir la hace un módulo de organización de archivos que conoce el tipo de ubicación usado por los archivos y la situación de los mismos.
- (2) **Sistema de archivos lógicos**, que conoce la estructura de directorios. Éstos se pueden tratar como archivos de forma que el módulo de organización puede leer y escribir los directorios cuando se producen peticiones del sistema de archivos lógicos.
- (3) **Programas de aplicaciones**.

c) Control de acceso :

Deberá verificar los permisos y los bloqueos de lectura y escritura.

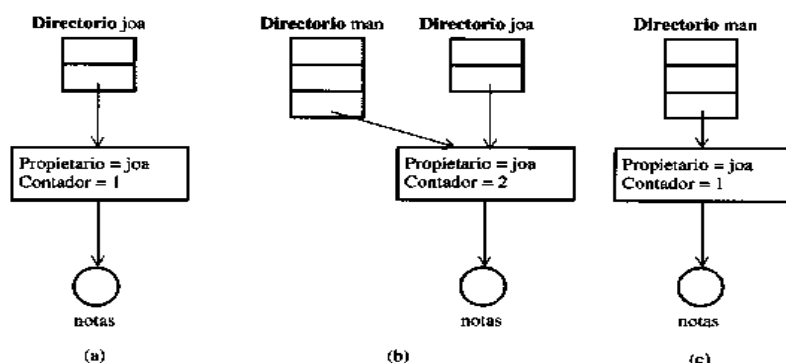
5.3.6 COMPARTICIÓN DE ARCHIVOS

Se puede realizar de distintas formas (operación LINK).

En UNIX existe una pequeña estructura de datos, el **i-nodo**, asociada al propio archivo, de forma que los directorios apuntarían al i-nodo correspondiente.

La creación de un enlace aumenta el contador de enlaces del i-nodo, de forma que el sistema conoce el número de entradas de directorio que apuntan al archivo.

Cuando se intenta eliminar el archivo original, si se eliminase el archivo y su i-nodo si más, el directorio tendría una entrada que apuntaría a un i-nodo no válido. La solución es dejar el i-nodo con su contador = 1.



TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

Enlaces simbólicos: Consiste en crear un nuevo archivo que contiene solamente la ruta de acceso al archivo al que se quiere enlazar. Sólo el propietario del archivo tiene un apuntador al i-nodo.

Ventajas :

- a) Se pueden utilizar para enlazar archivos a otras máquinas.

Inconvenientes :

- a) Alto coste. Hay que leer el archivo que contiene la ruta de acceso, analizarla y seguirla hasta alcanzar el i-nodo. Se necesita incluir un i-nodo para cada enlace simbólico, así como un bloque más en disco para almacenar la ruta de acceso, aunque ésta se podría almacenar en el propio i-nodo.
- b) Al restaurar un backup podrían crearse dos copias del mismo archivo.

5.4 CACHÉS DE DISCO

DEFINICIÓN

Consiste en mantener un depósito de copias de bloques del disco en la memoria principal, de forma que permita eliminar accesos repetitivos al disco.

Cuando se tiene que cargar un bloque en una caché totalmente ocupada se pueden utilizar los algoritmos FIFO, LRU, segunda oportunidad, etc.

Escritura diferida. La escritura se hace en la caché y posteriormente en el disco. Se pueden presentar situaciones de inconsistencia o corrupción de archivos entre ambas escrituras se produce un fallo del sistema. En UNIX se hace un vaciado de la caché a intervalos periódicos mediante una llamada al sistema (sync), que obliga a todos los bloques modificados a actualizarse en el disco.

Caché de escritura directa. Los bloques modificados se escriben inmediatamente en el disco (MS-DOS). Disminuye el rendimiento.

Ventajas del uso de cachés.

- a) Mejora del tiempo efectivo de acceso al disco y del tiempo de respuesta de las aplicaciones.
- b) Eliminación de algunos accesos a disco. Mientras los bloques están en la caché pueden ser leídos y escritos varias veces antes de actualizarse en el disco. Además se puede conseguir que los archivos temporales no se escriban en el disco.
- c) En la utilización de discos compartidos en red, supone una gran reducción de la carga en el servidor de archivos y en la red.

5.5 SEGURIDAD Y PROTECCIÓN

El término **seguridad** se suele referir al problema general de proteger la información tanto de daños físicos como de accesos inadecuados o no permitidos.

El término **mecanismo de protección** se aplica a los procedimientos específicos utilizados por el S.O. para asegurar la información del computador.

5.5.1 INTEGRIDAD DEL SISTEMA DE ARCHIVOS

Tipos de problemas :

- a) Bloques de disco en mal estado. Se puede conocer cuáles son y crear una lista.

Soluciones:

- 1) Elegir una pista de reserva para sustituir los bloques defectuosos.
 - 2) Eliminarlos de la lista de bloques libres.
- b) Fallo del sistema durante una operación de escritura. Puede dar lugar a inconsistencias en el sistema de archivos. Si además es un bloque con i-nodo, directorios o la lista de bloques libres, el problema puede ser crítico.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

Problemas usuales de inconsistencia :

Problema	Solución
Un bloque aparece en la lista de bloques usados y bloques libres	Eliminarlo de la lista de bloques libres.
Un bloque no aparece en ninguna lista.	Añadirlo a la lista de bloques libres.
Un bloque está repetido en la lista de bloques libres.	Reconstruir la lista de bloques libres.
Un bloque está asignado a dos o mas archivos.	Se asigna un bloque libre a un archivo y se copia el contenido del bloque que estaba asignado a los dos. Seguramente la información de al menos un archivo no será consistente.

Suele haber utilidades del S.O. que detectan las inconsistencias y, si no son muy graves, las corrigen (fsck en Unix).

Cuando el deterioro es irreparable, es necesario recurrir a las copias de seguridad.

Las copias de seguridad pueden hacerse en cintas magnéticas o en discos. Deben hacerse copias completas periódicamente, entre las que pueden intercalarse algunas copias incrementales.

5.5.2 ATAQUES A LA INTEGRIDAD Y SEGURIDAD DEL SISTEMA DE ARCHIVOS

Un sistema seguro debe mantener la integridad (los datos deben ser correctos), la disponibilidad y la privacidad de la información. Esto supone la protección frente a modificaciones no autorizadas y a la modificación no detectada de datos, así como la resistencia a la penetración:

Formas de penetración en un sistema informático :

- Utilización por parte de un intruso de la cuenta de un usuario legítimo.
- Ejecución de programas “caballos de Troya”, que ocultan parte de su funcionalidad, normalmente destinada a obtener datos o derechos de acceso del usuario.
- Propagación de gusanos y virus informáticos. El virus es parte del código de un programa, mientras que el gusano es un programa en sí mismo.
- Inspección del sistema de archivos. Acceso al archivo de contraseñas.

5.5.3 PRINCIPIOS DE DISEÑO DE SISTEMAS SEGUROS

- El diseño del sistema debe ser público. Los algoritmos pueden ser conocidos pero las claves deben ser secretas.
- El estado predefinido es el de no acceso. Los derechos de acceso deben adquirirse sólo con permiso explícito.
- Verificar la autorización actual.
- Mínimos privilegios. Cada proceso debe utilizar el mínimo grupo de privilegios para completar su tarea.
- Mecanismos simples e integrados. Mantener el diseño tan sencillo como se posible facilita la verificación y corrección de las implementaciones. El mecanismo debe estar integrado hasta las capas más bajas del sistema.
- Psicológicamente aceptable. El mecanismo debe ser fácil de usar, de forma que sea aplicado correctamente y no se rechazado por los usuarios.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

5.5.4 IDENTIFICACIÓN DE USUARIOS

Contraseñas :

Los usuarios eligen contraseñas fáciles de recordar.

Medidas:

- a) Cifrar la contraseña junto con un número aleatorio de bits.
- b) Exigir a los usuarios que cambien periódicamente la contraseña.
- c) Limitar el número de intentos fallidos de acceso bloqueando la cuenta o cortando la línea.

Identificación mediante artefactos :

Bandas magnéticas o tarjetas electrónicas. El usuario debe introducir la tarjeta y además suministrar una contraseña.

Las tarjetas inteligentes mantienen la contraseña del usuario secreta para el sistema, almacenándola en la propia tarjeta.

Identificación física :

Se utilizan características propias del usuario :

- a) Fisiológicas. Huellas dactilares o vocales, características faciales o geometría de la mano.
- b) De comportamiento. Análisis de firmas o patrones de voz.

La efectividad de las técnicas de identificación se hace en función de las falsas aceptaciones y de falsos rechazos.

5.6 MECANISMOS DE PROTECCIÓN Y CONTROL DE ACCESO

La **política de protección** dice qué se hará, mientras que los **mecanismos de protección** dicen cómo se hará.

5.6.1 DOMINIOS DE PROTECCIÓN

Un sistema de cálculo se puede idealizar como un conjunto de procesos y objetos. Por objetos se entiende tanto las distintas unidades del computador como las diferentes informaciones que almacenan.

Principio de la necesidad de saber: Un proceso sólo debe poder acceder a aquellos recursos para los cuales está autorizado y que necesita en ese momento para completar su tarea.

Dominio de protección: Conjunto de derechos de acceso, cada uno de los cuales está formado por un par de la forma:

<nombre del objeto, conjunto de sus derechos>

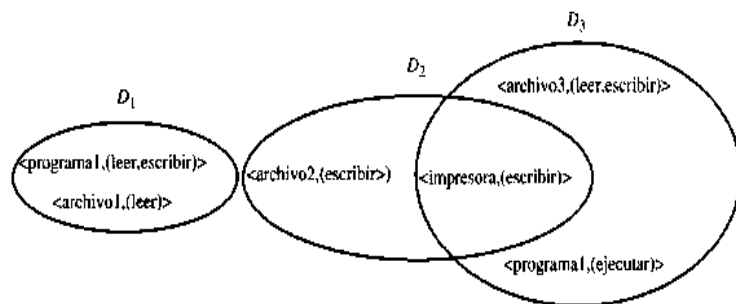
Cada proceso trabaja dentro de un dominio, el cual especifica los recursos a los cuales puede tener acceso. Cada dominio define un conjunto de objetos y las operaciones que se les pueden aplicar. La capacidad para ejecutar una operación sobre un objeto es un derecho de acceso.

En cada momento se ejecuta un proceso en algún dominio de protección.

En UNIX, el dominio de un proceso está definido por el identificador de usuario (uid) y del grupo (gid). Para un uid y un gid dados, se puede elaborar una lista completa de todos los objetos a los que se puede tener accesos y el tipo de acceso.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS



Para poder leer y escribir el objeto programa1 es necesario que un proceso se este ejecutando en el dominio D1, pero para poderlo ejecutar es necesario que esté en el dominio D3. Los dominios D2 y D3 comparten el objeto impresora con derecho de sólo escritura. En cada momento se ejecuta algún proceso algún dominio de protección.

5.6.2 MATRIZ DE ACCESO

Las relaciones entre dominios y objetos se pueden representar mediante una matriz de acceso. Las filas representan dominios y las columnas representan objetos.

Cada celda de la matriz contiene un conjunto de derechos de acceso.

Una forma sencilla de realizarla es mediante una tabla global consistente en tripletas (dominio, objeto, derechos)

Inconvenientes :

- 1) La tabla puede ser muy grande y no se puede conservar en memoria, por lo que requiere operaciones adicionales de E/S.
- 2) No se pueden aprovechar las agrupaciones de objetos o dominio. Por ejemplo, si todos pueden leer un objeto, éste deberá tener una entrada en cada dominio.

Soluciones :

- 1) Almacenar la matriz por columnas (lista de accesos) o por filas (lista de capacidades).

Dominios	Objetos				
	programa1	archivo1	archivo2	archivo3	Impresora
D ₁	leer/escribir	leer			
D ₂			escribir		escribir
D ₃	ejecutar			leer/escribir	escribir

5.6.3 LISTA DE ACCESOS

A cada objeto se le asocia una lista ordenada con todos los dominios que pueden tener acceso al objeto y la forma de dicho acceso.

Inconveniente :

El retardo que se provoca con la búsqueda para verificar la autoridad de un sujeto para acceder al objeto. Las listas pueden ser muy largas.

Solución :

Dividir a los usuarios en grupos y almacenar sólo los derechos de los grupos.

En UNIX, las listas están reducidas a tres entradas por archivo: propietario, grupo y otros.

5.6.4 LISTA DE CAPACIDADES

Almacenar la matriz de acceso por filas. A cada dominio o sujeto, se le asocia una lista de objetos a los cuales puede tener acceso, junto con una indicación de las operaciones permitidas sobre cada uno.

La lista de capacidades es un objeto protegido, mantenido por el S.O. y al que el usuario sólo puede acceder de forma indirecta.

Nunca se permite que una capacidad se mueva al espacio de direcciones accesibles por un proceso de usuario.

Manteniendo las capacidades seguras, los objetos a los que protegen también están seguros frente a un acceso no autorizado.

TEMA 5

GESTIÓN DEL SISTEMA DE ARCHIVOS

Formas de distinguir las capacidades :

- a) Asociar a cada objeto una etiqueta que indica si es una capacidad o un dato accesible. Los programas de aplicaciones no deben tener acceso directo a estas etiquetas. Esta restricción de acceso se hace mediante hardware o firmware.
- b) Dividir el espacio de direcciones asociado a un programa en dos partes:
 - 1) Una accesible al programa con sus datos e instrucciones.
 - 2) Otra que contiene la lista de capacidades y a la que sólo puede acceder el S.O.

Para ello es conveniente disponer de un espacio segmentado de memoria.

Inconveniente :

Para revocar el acceso a un objeto, el sistema tiene que determinar todas las capacidades existentes para el mismo y eliminarlas.

Solución:

Hacer que las capacidades apunten a un objeto indirecto en vez de al real. Para invalidar la capacidad, basta con romper el enlace entre el objeto indirecto y el real.