SISTEMAS OPERATIVOS

UNIDAD 8: ADMINISTRACIÓN DE LA INFORMACIÓN

Licenciatura en Ciencias de la Computación
Licenciatura en Sistemas de Información
Tecnicatura Universitaria en Programación Web

Departamento de Informática

Facultad de Ciencias Exactas, Físicas y Naturales

Universidad Nacional de San Juan

Esquema de contenidos

1. FUNCIONES DEL ADMINISTRADOR DE LA INFORMACIÓN 2
2. ARCHIVOS 4
2.1. Concepto de archivo 4
2.2. Atributos de un archivo 4
2.3. Estructura de un archivo 4
3. DIRECTORIOS DE ARCHIVOS 5
3.1. Contenido del directorio de archivos 5
3.2. Ubicación del directorio de archivos 7
3.3. Estructura del directorio de archivos 7
3.4. Nombrado de archivos 8
4. EL SISTEMA DE ARCHIVOS 10
4.1. Modelo General de un Sistema de Archivos 10
4.2. Sistema de Archivo Simbólico 11
4.3. Sistema de Archivo Básico 12
4.4. Verificación de Control de Acceso 13
4.5. Sistema de Archivo Lógico 13
4.6. Sistema de Archivo Físico 14
4.7. Módulo de Estrategia de Asignación 15
4.8. Módulo de Estrategia de Periférico 15
4.9. Resumen de los módulos 15
5. IMPLEMENTACIÓN DE ARCHIVOS (Método de acceso) 16
5.1. Asignación contigua 16
5.2. Asignación enlazada 18
5.3. Asignación indexada 19
6. GESTIÓN DEL ESPACIO LIBRE 20
6.1. Mapa o vector de bits 20
6.2. Agrupamiento de la lista enlazada 21
7. BIBLIOGRAFÍA 22

1. FUNCIONES DEL ADMINISTRADOR DE LA INFORMACIÓN

El módulo de Administración de la Información o Gestión de Archivos o Administrador de Archivos es aquel conjunto de software de sistema que proporciona servicios a los usuarios y aplicaciones para el uso de archivos. Típicamente, la única forma en la que un usuario o aplicación puede acceder a los archivos es a través del sistema de gestión de archivos. Esto elimina la necesidad de que el usuario o programador desarrolle software de propósito especial para cada aplicación. Además,

proporciona al sistema una forma consistente y bien definida de controlar su recurso más importante.
Posee las siguientes funciones básicas:

1. **Almacenar** toda la información de los archivos del sistema a través de varias tablas, siendo la mayor de ellas el Directorio de Archivos.
   Estas tablas contienen para cada archivo: nombre, ubicación, longitud (cantidad de registros del archivo), longitud del registro lógico, longitud del registro físico, formato de registros (fijo, variable, etc.), organización (secuencial, indexada, aleatoria, etc.), fecha de creación, fecha de expiración, derechos de acceso, extensiones, etc.

2. Decidir qué **política** es utilizada para determinar dónde y cómo la información es almacenada.
   Algunos factores que influyen en esta política son: utilización efectiva del almacenamiento secundario, acceso eficiente, flexibilidad a usuarios y, por último, protección de derechos de acceso sobre información pedida.

3. **Asignación y desasignación** del recurso de la información (archivos)
   Una vez que la decisión de permitir a un proceso tener acceso a cierta información es efectuada, los módulos de ubicación de la información deben encontrar dicha información, hacer accesible dicha información al proceso y por último establecer los derechos de acceso apropiados. De igual manera, una vez que la información no se necesita más, las entradas en tablas asociadas a esta información pueden ser eliminadas. Si el usuario ha actualizado la información, la copia original de la información puede ser actualizada para posible uso de otros procesos.

Comúnmente, al conjunto de módulos del Administrador de Información se lo referencia como el *Sistema de Archivos* (*File System*). Es una meta al concebir un Sistema de Archivos que el usuario sea liberado de problemas tales como la ubicación de la información que está referenciando, así como el formato real de dicha información en el sistema y el problema de acceso al periférico de almacenamiento. Al quedar liberado el usuario-programador de tales escollos, puede concentrarse totalmente en su problema específico (estructura lógica y operaciones necesarias para procesar dicha estructura de datos).

2. ARCHIVOS

2.1. Concepto de archivo

Las computadoras pueden almacenar información en varios soportes de almacenamiento, como discos magnéticos, discos ópticos y memorias flash. Para que el sistema informático sea cómodo de utilizar, el SO proporciona una vista lógica uniforme para el almacenamiento de la información. El SO realiza una abstracción de las propiedades físicas de los dispositivos de almacenamiento, con el fin de definir una unidad lógica de almacenamiento, el archivo. Los archivos son mapeados por el SO sobre dispositivos físicos. Estos dispositivos son generalmente, no volátiles, de modo que los contenidos persisten aunque se produzcan fallos de alimentación o reinicios de sistema.
Se definirá archivo a una colección de información relacionada entre sí almacenada en la memoria secundaria de una computadora.

2.2. Atributos de un archivo

Los archivos tienen un nombre (una cadena de caracteres), por comodidad para los usuarios humanos, y para referirse a él se utiliza ese nombre. Algunos sistemas diferencian entre mayúsculas y minúsculas dentro de los nombres.
Los atributos de un archivo varían de un sistema a otro, pero típicamente son los siguientes: nombre, identificador, tipo, ubicación, tamaño, protección, fecha, hora, identificación del usuario, etc.
La información acerca de los archivos se almacena en la estructura de directorios, que también reside en el almacenamiento secundario. Típicamente, una entrada de directorio está compuesta del nombre de un archivo y de su identificador unívoco. El identificador, a su vez, permite localizar los demás atributos del archivo.

2.3. Estructura de un archivo

Cuatro términos aparecen normalmente cuando se habla sobre archivos:

- Campo
- Registro
- Archivo
- Base de datos

Un *campo* es el elemento básico de los datos. Un campo individual contiene un único valor, tal como el apellido de un empleado, una fecha o la edad. Se caracteriza por su

longitud y el tipo de datos (por ejemplo, ASCII, cadena de caracteres, decimal). Dependiendo del diseño del archivo, el campo puede tener una longitud fija o variable. En este último caso, el campo está formado normalmente por dos o tres subcampos: el valor real almacenado, el nombre del campo, y en algunos casos, la longitud del campo. En otros casos de campos de longitud variable, la longitud del campo se indica mediante el uso de símbolos de demarcación especiales entre campos.
Un *registro* es una colección de campos relacionados que pueden tratarse como una unidad por alguna aplicación. Por ejemplo, un registro de empleado podría contener campos tales como nombre, número de seguridad social, clasificación de trabajo, fecha de contratación, etc. De nuevo, dependiendo del diseño, los registros pueden ser de longitud fija o variable.
Un *archivo o fichero* es una colección de registros similares. El archivo se trata como una entidad única por parte de los usuarios y las aplicaciones. Los archivos se pueden referenciar por nombre. Dichos archivos se pueden crear y borrar. Las restricciones de control de acceso normalmente se aplican a nivel del archivo. Es decir, en un sistema compartido, el acceso a los archivos completos es permitido o denegado a los usuarios y los programas.
Una *base de datos* es una colección de datos relacionados. Los aspectos esenciales de una base de datos son que la relación que exista entre los elementos de datos sea explícita y que la base de datos se diseñe para su uso por parte de varias aplicaciones diferentes. Una base de datos podría contener toda la información relacionada con una organización o proyecto, tal como información de negocio o de estudio científico. La base de datos está formada por uno o más tipos de archivos. Normalmente, hay un sistema de gestión de base de datos separado del SO, aunque hace uso de algunos programas de gestión de archivos.

3. DIRECTORIOS DE ARCHIVOS

3.1. Contenido del directorio de archivos

Asociado con cualquier sistema de gestión de archivos y colección de archivos, se encuentra el concepto de directorio. El directorio contiene información sobre los archivos, incluyendo atributos, ubicación y propiedad. Gran parte de esta información, especialmente la que concierne a almacenamiento, la gestiona el SO. El directorio es a su vez un archivo, accesible por varias rutinas de gestión de archivos. Aunque parte de la información de los directorios está disponible para los usuarios y las aplicaciones, esto se proporciona generalmente de forma indirecta por las rutinas del sistema.

En la tabla se muestra la información normalmente utilizada en el directorio por cada archivo del sistema. Cada entrada de la tabla corresponde a un archivo, el cual es reconocible por su nombre. Prácticamente todos los sistemas tratan con diferentes tipos de archivos y distintas organizaciones de archivos (secuencial, secuencial indexado, indexado o aleatorio) y esta información también se proporciona. Una importante categoría de información sobre cada archivo trata sobre el almacenamiento, incluyendo su ubicación y tamaño. En sistemas compartidos, es también importante proporcionar información que se utilice para controlar el acceso de los distintos usuarios a los archivos. Típicamente, un usuario es el propietario del archivo y puede conceder ciertos privilegios de acceso a otros usuarios. Finalmente, se utiliza información de uso para gestionar la utilización actual del archivo y registrar la historia de su uso.

Atributo	**Información básica**
Nombre de archivo	Nombre escogido por el creador (usuario o programa). Debe ser único dentro de un directorio específico
Tipo de archivo	Por ejemplo texto, binario, módulo de carga, etc
Organización de archivo	Para sistemas que soportan diferentes organizaciones
	Información de direccionamiento
Volumen	Indica el dispositivo en el cual se almacena el archivo
Dirección inicial	Dirección física inicial en almacenamiento secundario (por ejemplo, cilindro, pista y número de bloque en disco)
Tamaño utilizado	Tamaño actual del archivo en bytes, palabras o bloques
	Información de control de acceso
Propietario	Usuario que tiene él control del archivo El propietario puede conceder/denegar acceso a otros usuarios y cambiar estos privilegios
Información de acceso	Una versión sencilla de este elemento incluye el nombre de usuario y clave para cada usuario autorizado
Acciones permitidas	Controla la lectura, escritura, ejecución y la transmisión a través de la red
	Información de uso
Fecha de creación	Fecha en la que el archivo se coloca por vez primera en el directorio

Identidad del creador	Normalmente aunque no necesariamente el propietario actual
Fecha de último acceso de lectura	Fecha de la última vez que se leyó un registro
Identidad de último lector	Usuario que hizo la última lectura
Fecha de último acceso de modificación	Fecha de la última actualización, inserción o borrado
Identidad de último modificador	Usuario que hizo la última modificación
Fecha de la última copia de seguridad	Fecha de la última vez que el archivo fue copiado en otro medio de almacenamiento
Uso actual	Información sobre la actividad actual sobre el archivo, tal como proceso o procesos que tienen el archivo abierto, si está bloqueado por un proceso y si el archivo ha sido actualizado en memoria principal pero no en disco

Tabla de elementos de información de un directorio

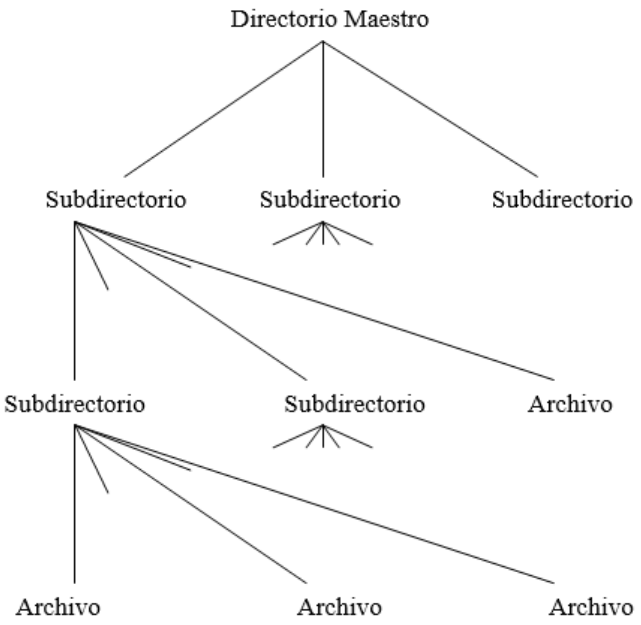
3.2. Ubicación del directorio de archivos

La forma en la que la información de la tabla del apartado anterior se almacena difiere ampliamente entre los sistemas.
Si la totalidad del directorio de archivos fuera mantenido todo el tiempo en memoria principal, se necesitaría una gran cantidad de memoria solo para éste propósito, por lo que surge la idea de mantener el directorio de archivos dentro de los propios medios de almacenamiento (discos, memorias electrónicas).
De esta manera se almacena en memoria principal sólo una parte del directorio con las entradas que pertenecen a archivos que fueron referenciados anteriormente.

3.3. Estructura del directorio de archivos

En la evolución de la computación, hubo también una evolución en la forma de estructurar la información, pasando del directorio único para todo el sistema, por otro esquema de dos niveles donde hay un directorio por cada usuario y un directorio maestro, hasta llegar a la estructura actual.
Esta técnica consiste en utilizar una estructura jerárquica en forma de árbol (ver imagen), lo que la hace más potente y flexible, y es casi universalmente adoptada. Consiste en un directorio maestro, que tiene bajo dicho directorio varios directorios de usuario. Cada uno de estos directorios de usuario, a su vez, podría tener subdirectorios

y archivos como entradas. Esto se cumple para todos los niveles: es decir, en cada nivel, un directorio podría estar formado por subdirectorios y/o archivos.



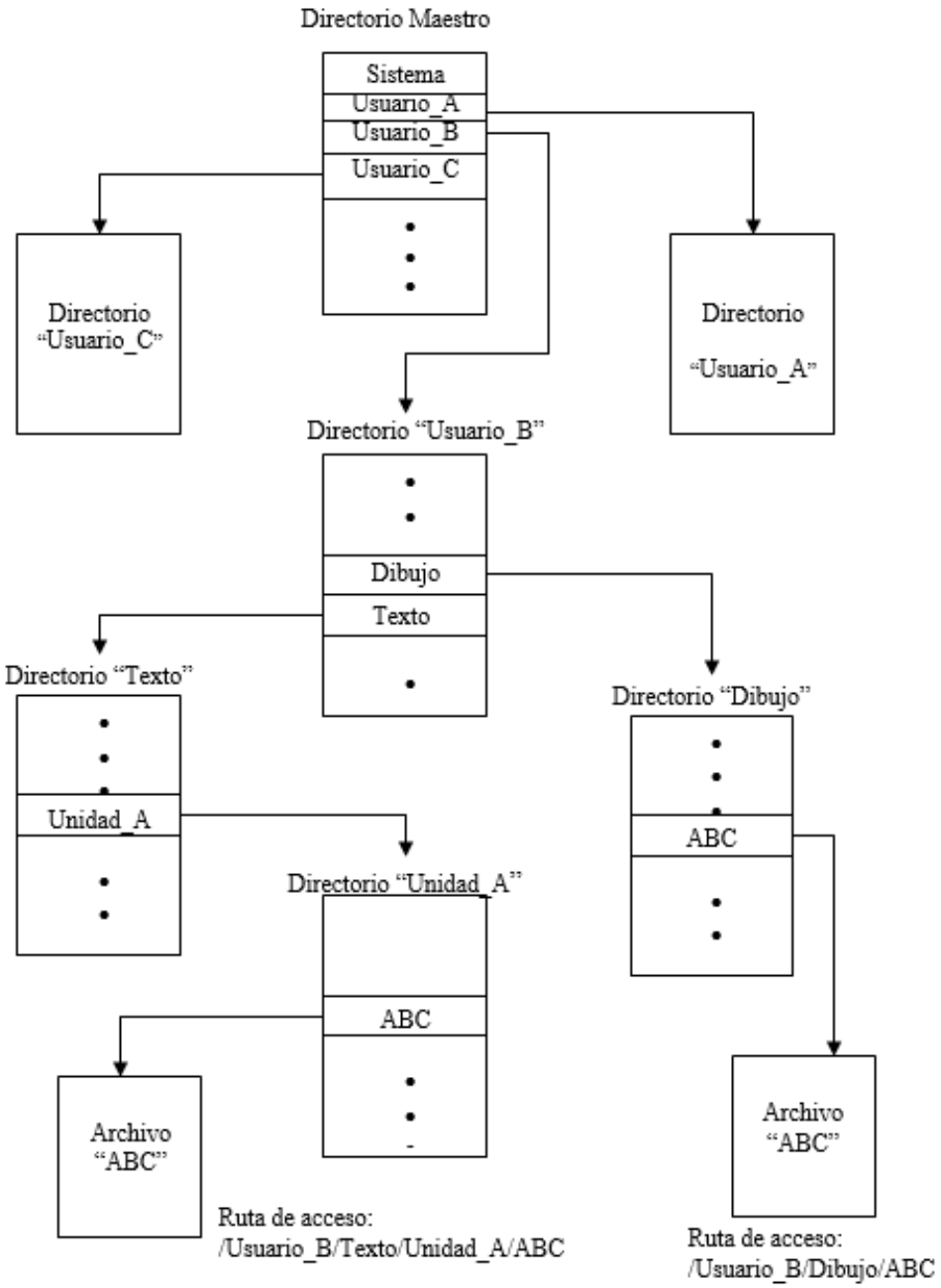
3.4. Nombrado de archivos

Como se dijo, los usuarios necesitan poder referenciar un archivo mediante un nombre simbólico. Claramente, cada archivo en el sistema debe tener un nombre único a fin de que las referencias al mismo no sean ambiguas. Por otro lado, es inaceptable obligar a que los usuarios proporcionen nombres únicos, especialmente en un sistema compartido.

El uso de un directorio estructurado en forma de árbol minimiza la dificultad de asignar nombres únicos. Cualquier archivo del sistema se puede localizar siguiendo un camino (path) desde el directorio raíz o maestro y bajando por las ramas hasta alcanzar el archivo. El conjunto de nombres de directorios, finalizando en el nombre del archivo, constituye un nombre de camino para el archivo.

Por ejemplo, el archivo de la esquina inferior izquierda de la figura tiene el camino /Usuario_B/texto/Unidad A/ABC. La barra se utiliza para delimitar (separar) nombres en la secuencia. El nombre del directorio maestro es implícito, porque leídos los nombres comienzan en dicho directorio. Obsérvese que es perfectamente aceptable tener varios archivos con el mismo nombre, siempre que ambos archivos tengan nombres de camino únicos, lo que equivale a decir que el mismo nombre de archivo se puede utilizar en diferentes directorios. En el ejemplo, hay otro archivo en el sistema con el nombre ABC, pero cuyo nombre completo es /Usuario_B/Dibujo/ABC.

Aunque los nombres de camino facilitan la selección de los nombres de archivo, sería complicado para el usuario tener que escribir el camino completo cada vez que se hace una referencia a un archivo. Normalmente, un usuario interactivo o un proceso está asociado con un directorio actual, que se suele denominar *directorio de trabajo*. Los archivos se pueden referenciar de manera relativa al directorio de trabajo.

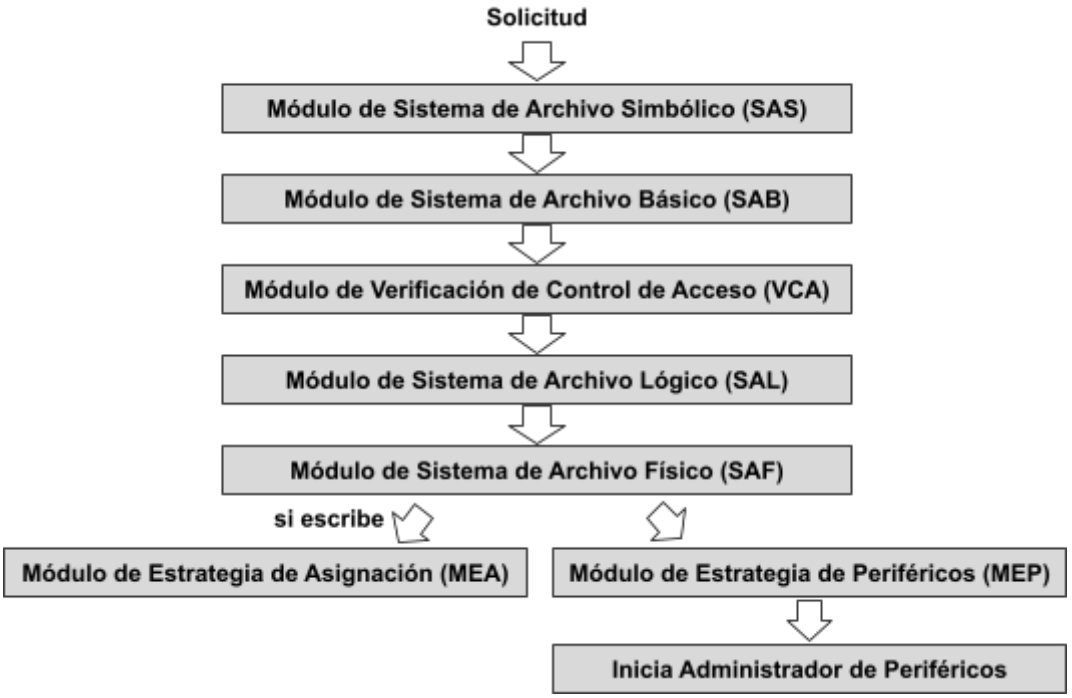


4. EL SISTEMA DE ARCHIVOS

4.1. Modelo General de un Sistema de Archivos

Tanto los HDD como SSD son los dispositivos preferidos como el principal tipo de almacenamiento secundario para mantener sistemas de archivos.
Un sistema de archivos (File System) debe proporcionar un acceso eficiente y cómodo al disco, permitiendo que los datos (archivos) puedan almacenarse, localizarse y extraerse fácilmente. Un sistema de archivos acarrea dos problemas de diseño bastante diferentes:

1. El primer problema es definir qué **aspecto** debe tener el sistema de archivos para el usuario. Esta tarea implica definir un archivo y sus atributos, las operaciones permitidas sobre los archivos y la estructura de directorio utilizada para organizar los archivos.
2. El segundo problema es crear **algoritmos y estructuras de datos** que permitan mapear el sistema lógico de archivos sobre los dispositivos físicos de almacenamiento secundario.



Modelo General de un Sistema de Archivos

El propio sistema de archivos está compuesto, generalmente, de muchos niveles diferentes. La estructura que se muestra en la imagen es un ejemplo de diseño en niveles; cada nivel del diseño utiliza las funciones de los niveles inferiores para crear nuevas funciones que serán, a su vez, utilizadas por los niveles superiores a ese.

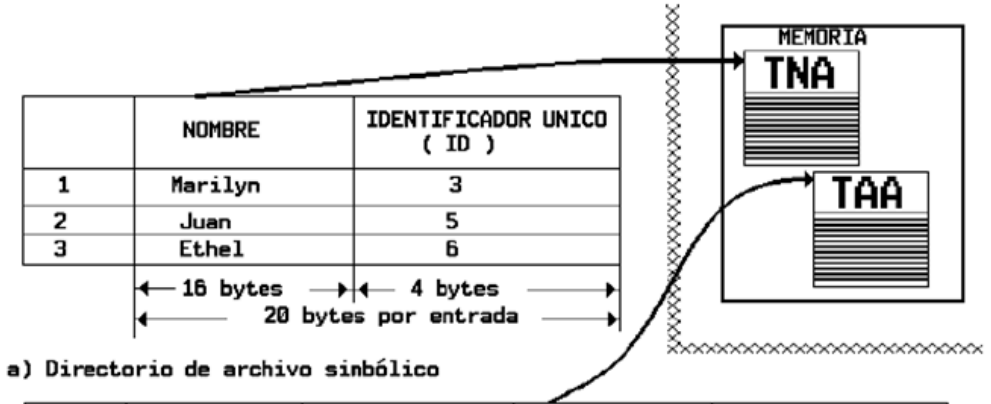
También es necesario destacar que aunque los detalles específicos dados a continuación pueden variar significativamente de un sistema a otro, la estructura básica es común a la mayoría de los sistemas actuales. Es decir, dependiendo del sistema específico que se trate, algunos de estos módulos se fusionan, se desglosan en aún más módulos, o incluso algunos desaparecen. Aún así la estructura subyacente sigue siendo la misma.

4.2. Sistema de Archivo Simbólico

El primer módulo que es llamado cuando llega un pedido al Sistema de Archivos es el módulo denominado Sistema de Archivo Simbólico (SAS).
Una típica llamada sería por ejemplo:

CALL SAS (READ,“JUAN”,4,1200)

En la instrucción se pide leer el registro lógico número 4 del archivo “JUAN”, para colocar su contenido en la dirección 1200 de memoria principal.



	NOMBRE	IDENTIFICADOR UNICO (ID)
1	Marilyn	3
2	Juan	5
3	Ethel	6

16 bytes — 4 bytes
20 bytes por entrada

a) Directorio de archivo simbólico

ID	LONGITUD DE REGISTRO LOGICO	CANTIDAD DE REGISTROS LOGICOS	DIRECCION AL PRIMER BLOQUE FISICO	DIRECTORIO Y CONTROL DE ACCESO
1	---	---	0	Dir. Básico
2	20	3	1	Dir. Simbólico
3	80	10	2	Todos Leen
4	1000	3	3	Libre
5	500	7	6	Todos Leen
6	100	30	12	MARTA Lee JUAN Lee/Graba
7	1000	2	10	Libre
8	1000	1	15	Libre

b) Directorio de archivo básico

El SAS usa el nombre del archivo para localizar la única entrada que posee el archivo en el Directorio de Archivos.

Es necesario aclarar, antes de seguir adelante, que el Directorio de Archivos se divide en dos partes. Un Directorio de Archivo Simbólico (DAS) y un Directorio de Archivo Básico (DAB) como se muestra en la figura.
El Sistema de Archivo Simbólico debe buscar en el Directorio de Archivo Simbólico la entrada perteneciente al archivo requerido y de esta forma encontrar su identificador (ID) dentro del sistema, para pasárselo al módulo denominado Sistema de Archivo Básico (SAB).
En el ejemplo CALL SAS (READ,“JUAN”,4,1200) devolvería 5 donde 5 es el ID del archivo “JUAN”.
Normalmente como el DAS es mantenido en el periférico de almacenamiento, lo que se hace es copiar en memoria las entradas del DAS que corresponden a archivos en uso (llamados archivos abiertos o activos), de tal forma que estas entradas son usadas para evitar operaciones de E/S sobre la misma zona en el periférico. Esta tabla que se mantiene en memoria principal con las entradas del DAS correspondientes a archivos abiertos recibe el nombre de Tabla de Nombres Activos (TNA).

4.3. Sistema de Archivo Básico

El segundo módulo que es llamado en la secuencia se denomina Sistema de Archivo Básico (SAB). La llamada sería de esta forma:

CALL SAB (READ,5,4,1200)

Donde todos los parámetros son iguales a la llamada del módulo SAS, a excepción del segundo parámetro que constituye el identificador que le pasó el SAS.
El SAB se vale del identificador del archivo (ID) para localizar la entrada correspondiente a este en el Directorio de Archivo Básico.
Dicha entrada es mantenida en memoria principal con el objeto de ahorrar posteriores accesos de E/S buscando la misma entrada. La tabla que alberga todas las entradas del Directorio de Archivo Básico de los archivos abiertos recibe el nombre de Tabla de Archivos Activos (TAA). Esta tabla se genera y mantiene en memoria principal a diferencia del directorio básico (DAB) y del directorio simbólico (DAS). Su entrada consta de la información que puede visualizarse:

Para la próxima etapa (Verificación de Control de Acceso (VCA)) se utiliza la entrada del archivo correspondiente en la TAA, la cual contiene la información del archivo ID 5.
En resumen el módulo SAB se encarga de:

- El Directorio de Archivo Básico.
- La Tabla de Archivos Activos.
- La comunicación con el módulo VCA.

4.4. Verificación de Control de Acceso

La invocación al módulo de Verificación de Control de Acceso es de la siguiente forma:

CALL VCA (READ,5,4,1200)

Este módulo actúa como un punto de control entre el Sistema de Archivo Básico y el módulo de Sistema de Archivo Lógico, de tal forma que verifica si la operación que se quiere realizar sobre el archivo en cuestión (READ, WRITE, etc.) está permitida en la entrada correspondiente al archivo en la TAA. Esto significa que a un proceso se le permitirá el acceso a un archivo dependiendo de los permisos que se le hayan otorgado.
En caso que no sea permitido realizar dicha operación sobre el archivo se genera una condición de error, por lo cual el pedido al Sistema de Archivos es cancelado. Si efectivamente se puede realizar esa operación entonces el control pasa (se invoca) al módulo de SAL.

4.5. Sistema de Archivo Lógico

Una llamada al módulo de Sistema de Archivo Lógico (SAL) posee la misma sintaxis que la llamada al módulo anterior. Luego, para el ejemplo, quedaría:

CALL SAL (READ,5,4,1200)

El Sistema de Archivo Lógico convierte el pedido de un registro lógico en el pedido de una secuencia de bytes lógicos, la cual se entrega al Sistema de Archivo Físico (SAF).
Esto es así, puesto que para el SAF un archivo no es más que una secuencia de bytes sin ningún tipo de formato.

En esta capa de software se maneja el método de acceso (se verá más adelante en este capítulo).
Para el ejemplo, se supone un formato de registro lógico de longitud fija, luego la conversión necesaria se la puede obtener de la información de la entrada en la TAA.
Dirección del byte lógico (dbl) = (número de registro - 1) * longitud del registro lógico = 3 * 500 = 1500 long. de la secuencia de bytes lógicos (lsb) = long. del registro lógico

4.6. Sistema de Archivo Físico

Teniendo ya calculado el comienzo de la secuencia de bytes y su longitud, el SAL invoca al módulo de Sistema de Archivo Físico (SAF) de la siguiente forma:

CALL SAF (READ,5,1500,500,1200) 1500 es dbl y 500 el lsb

Donde el tercer y cuarto parámetro corresponden al dbl y el lsb respectivamente y el resto de parámetros son exactamente los mismos que los de la llamada el módulo anterior.
El SAF tiene por función determinar en qué bloque físico del dispositivo de almacenamiento se encuentra la secuencia de bytes lógicos pasados por el SAL, para lo cual se vale de la entrada en la TAA más el dbl y el lsb.
El bloque es entonces leído y colocado en un buffer preasignado en memoria principal y, luego, es extraído de este buffer la secuencia de bytes pedidos y colocados en el área de buffer del usuario (en el caso del ejemplo, en la dirección 1200).
Para realizar estos cálculos el SAF debe saber las funciones de mapeo y longitud del bloque físico que utiliza cada periférico de almacenamiento. Si estas fueran las mismas para todo tipo de periférico podrían estar tranquilamente insertas en las mismas rutinas del SAF, pero tal cosa no sucede pues hay variaciones de un periférico a otro.
Esto se resuelve manteniendo tal información en el mismo volumen de almacenamiento, de tal forma que cuando se inicializa el sistema toda esta información pueda ser leída en una entrada de la TAA.
Si se hace un pedido de escritura y el bloque sobre el cual se quiere escribir no está asignado previamente entonces el SAF invoca al Módulo de Estrategia de Asignación (MEA) para que este le proporcione un bloque libre en memoria secundaria sobre el cual pueda efectuar la escritura.

4.7. Módulo de Estrategia de Asignación

Este módulo se encarga de llevar un registro del espacio libre disponible en el almacenamiento secundario (este tema se verá más adelante), tal información aparecerá reflejada en la TAA.

4.8. Módulo de Estrategia de Periférico

Este módulo convierte el número de bloque físico en el formato adecuado para el periférico en cuestión (por ejemplo bloque 7 = pista 3, sector 23). Además de esto, inicializa los comandos adecuados de E/S para el tipo de periférico sobre el cual se va a realizar la operación.
Cabe destacar que todos los módulos anteriormente citados son totalmente independientes de cualquier tipo de periférico con excepción del último nombrado. De aquí en más, el control pasa a manos del Administrador de Periféricos.

4.9. Resumen de los módulos

A modo de resumen se reseña brevemente el accionar de los módulos:

1. Sistema de archivo simbólico: transforma el nombre del archivo en el identificador único del Directorio de archivos. Utiliza la Tabla de nombres activos (TNA) y el directorio de archivo simbólico (DAS).
2. Sistema de archivos básico: utiliza el directorio de archivo básico (DAB) y copia la información del archivo en la Tabla de archivos activos (TAA).
3. Verificación de control de acceso: verifica los permisos de acceso al archivo.
4. Sistema de archivo lógico: transforma el pedido lógico en una secuencia de bytes lógicos.
5. Sistema de archivo físico: calcula la dirección física.
6. Módulo de estrategia de asignación: consigue espacio disponible en el periférico (en el caso de grabación).
7. Módulo de estrategia de periférico: transforma la dirección física según las características específicas del periférico requerido.

5. IMPLEMENTACIÓN DE ARCHIVOS (Método de acceso)

El acceso directo inherente a la naturaleza de los discos proporciona una gran flexibilidad a la hora de implementar los archivos. En casi todos los casos, habrá múltiples archivos almacenados en el mismo disco. El principal problema es cómo asignar el espacio a esos archivos de modo que el espacio de disco se utilice eficazmente y que se pueda acceder a los archivos de forma rápida. Hay tres métodos principales de asignación del espacio de disco que se utilizan habitualmente: asignación contigua, enlazada e indexada. Cada método tiene sus ventajas y desventajas. Usualmente, cada SO utiliza un único método para implementar todos los archivos.

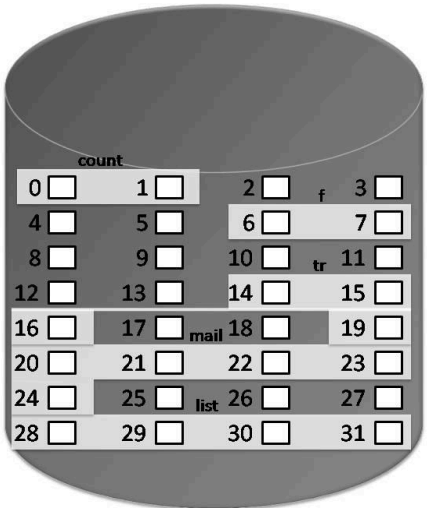
5.1. Asignación contigua

La asignación contigua requiere que cada archivo ocupe un conjunto de bloques contiguos en el disco. Las direcciones de disco definen una ordenación lineal del disco. Con esta ordenación, suponiendo que sólo haya una tarea accediendo al disco, acceder al bloque b + 1 después del bloque b normalmente no requiere ningún movimiento de cabezal. Cuando hace falta un movimiento de cabezal (para pasar del último sector de un cilindro al primer sector del siguiente cilindro) el cabezal sólo necesitará moverse de una pista a la siguiente, por tanto, el número de reposicionamientos del cabezal del disco requeridos para acceder a los archivos que están asignados de modo contiguo es mínimo, al igual que lo es el tiempo de búsqueda cada vez que hace falta reposicionar los cabezales. El sistema operativo para mainframes VM/CMS de IBM utiliza una asignación contigua, debido precisamente a las altas prestaciones que proporciona.

La asignación contigua de un archivo está definida por la asignación de disco del primer bloque y por la longitud del archivo (en unidades de bloques). Si el archivo tiene n bloques de longitud y comienza en la ubicación b, entonces ocupará los bloques b, b + 1, b + 2, ..., b + n-1. La entrada de directorio de cada archivo indicará la dirección del bloque de inicio y la longitud del área asignada al archivo.

Acceder a un archivo que haya sido asignado de manera contigua resulta sencillo. Para el acceso secuencial, el sistema de archivos recuerda la dirección de disco del último bloque al que se haya hecho referencia y leerá el siguiente bloque cuando sea necesario.

Sin embargo, la asignación contigua también tiene sus problemas. Una de las dificultades estriba en encontrar espacio para un nuevo archivo.



archivo	Inicio	longitud
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

El problema de la asignación contigua puede verse como un caso concreto del problema general de asignación dinámica del almacenamiento que se ha analizado en la unidad de Administración de Memoria y que se refiere a cómo satisfacer una solicitud de tamaño n a partir de una lista de huecos libres. Las estrategias más comunes para seleccionar un hueco libre a partir del conjunto de huecos disponibles son las de primer ajuste y mejor ajuste. Las simulaciones muestran que tanto la técnica de primer ajuste como la de mejor ajuste son más eficientes que la de peor ajuste tanto en términos de tiempo como en términos de utilización del almacenamiento. Las técnicas de primer ajuste y de mejor ajuste son muy similares en lo que se refiere a la utilización del espacio de almacenamiento, pero la técnica de primer ajuste es generalmente más rápida.

Todos estos algoritmos sufren el problema de la fragmentación externa. A medida que se asignan y borran archivos, el espacio libre del disco se descompone en pequeños fragmentos. Siempre que el espacio libre se descompone en fragmentos distintos aparece el problema de la fragmentación externa y este problema llegará a ser significativo cuando el fragmento contiguo más grande sea insuficiente para satisfacer una solicitud; el espacio de almacenamiento estará fragmentado en una serie de huecos, ninguno de los cuales será lo suficientemente grande como para almacenar los datos.

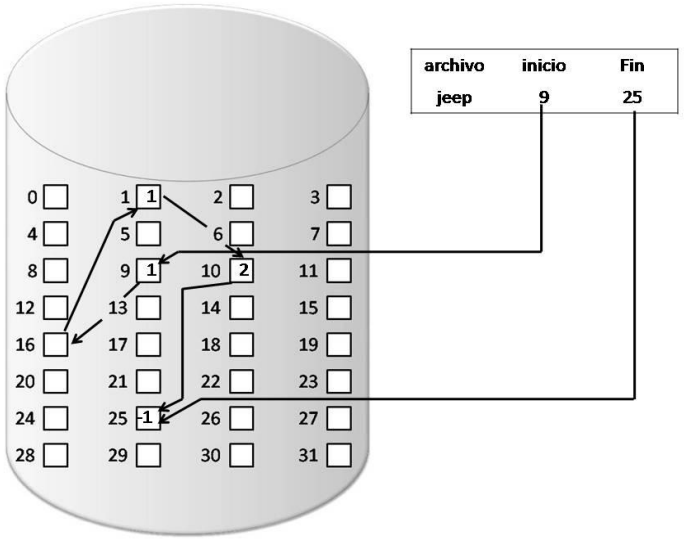
Por lo tanto, inevitablemente será necesario realizar tareas de compactación periódicamente.

Otro problema de los mecanismos de asignación contigua es determinar, de antemano, cuánto espacio se necesita para un archivo. En el momento de crear el archivo, el SO debe determinar la cantidad total de espacio que va a necesitar ese archivo, lo que es bastante difícil de estimar. Si se asigna poco espacio a un archivo, se puede encontrar con que el archivo no puede ampliarse, debiendo desplazarlo a otro lugar.

5.2. Asignación enlazada

El mecanismo de asignación enlazada resuelve todos los problemas de la asignación contigua. Con la asignación enlazada, cada archivo es una lista enlazada de bloques de disco, pudiendo estar dichos bloques dispersos por todo el disco. El directorio contiene un puntero al primer y al último bloque de cada archivo. Por ejemplo, un archivo de cinco bloques podría comenzar en el bloque 9 y continuar en el bloque 16, luego en el 1, después en el bloque 10 y, finalmente, en el bloque 25 (ver figura). Cada bloque contiene un puntero al bloque siguiente. Dichos punteros no están a disposición del usuario. Con este mecanismo, si cada bloque tiene 512 bytes de tamaño y una dirección de disco (el puntero) requiere 4 bytes, el usuario verá bloques de 508 bytes.

Para crear un nuevo archivo, simplemente se crea una nueva entrada en el directorio. Con el mecanismo de asignación enlazada, cada entrada de directorio tiene un puntero al primer bloque de disco del archivo. Este puntero se inicializa con el valor nulo (el valor de puntero que indica el fin de la lista) para representar un archivo vacío. El campo de tamaño también se configura con el valor 0. Una escritura en el archivo hará que el sistema de gestión de espacio libre localice un bloque libre y la información se escribirá en este nuevo bloque y será enlazada al final del archivo. Para leer un archivo, simplemente se leen los bloques siguiendo los punteros de un bloque a otro. No hay ninguna fragmentación externa con asignación enlazada y puede utilizarse cualquiera de los bloques de la lista de bloques libres para satisfacer una solicitud. No es necesario declarar el tamaño del archivo en el momento de crearlo y el archivo puede continuar creciendo mientras existan bloques libres. En consecuencia, nunca es necesario compactar el espacio de disco.



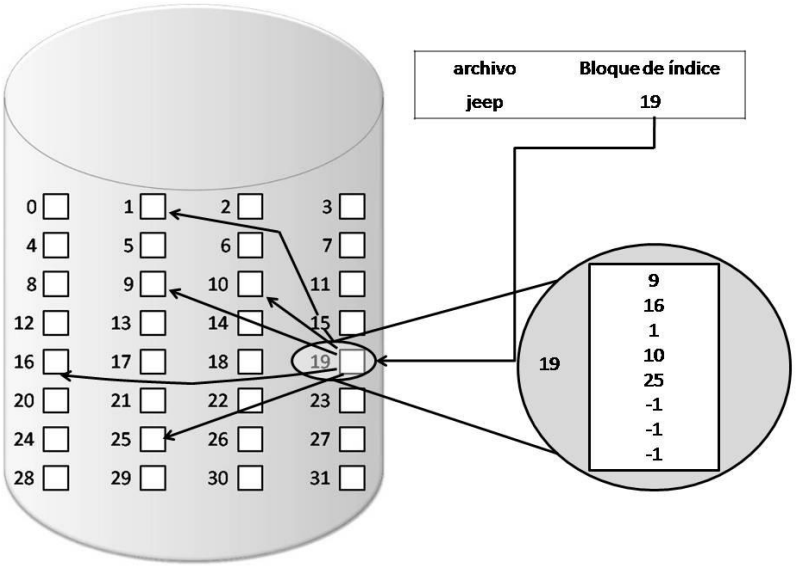
Sin embargo, el mecanismo de asignación enlazada también tiene sus desventajas. El principal problema es que sólo se puede utilizar de manera efectiva para los archivos

de acceso secuencial. Para encontrar el bloque i-ésimo de un archivo, se comienza al principio de dicho archivo y sigue los punteros hasta llegar al bloque i-ésimo. Cada acceso a un puntero requiere una lectura de disco y algunos de ellos requerirán un reposicionamiento de cabezales. En consecuencia, resulta muy poco eficiente tratar de soportar una funcionalidad de acceso directo para los archivos de asignación enlazada.
Otra desventaja es el espacio requerido para los punteros. Si un puntero ocupa 4 bytes de un bloque de 512 bytes, el 0,758 por ciento del espacio del disco se estará utilizando para los punteros, en lugar de para almacenar información útil. Cada archivo requerirá un poco más de espacio de lo que requeriría con otros mecanismos.
Otro problema de la asignación enlazada es la fiabilidad. Recuerde que los archivos están enlazados mediante punteros que están dispersos por todo el disco; considere ahora lo que sucedería si uno de esos punteros se pierde o resulta dañado. Un error en el software del SO o un fallo del hardware del disco podrían hacer que se interprete incorrectamente un puntero, este error, a su vez, podría hacer que un enlace apunte a la lista de bloques libres o a otro archivo.
Este mecanismo de asignación enlazada, aunque con variantes, es la que se basa en el uso de una tabla de asignación de archivos (FAT, file allocation table) utilizada en los SO MS-DOS y OS/2.

5.3. Asignación indexada

El método de la asignación enlazada resuelve los problemas de fragmentación externa y de declaración del tamaño que presentaba el método de la asignación contigua. Sin embargo, no puede soportar el acceso directo, ya que los punteros a los bloques están dispersos junto con los propios bloques por todo el disco y deben extraerse secuencialmente. El mecanismo de asignación indexada resuelve este problema agrupando todos los punteros en una única ubicación: el *bloque de índice*.
Cada archivo tiene su propio bloque de índice, que es una matriz de direcciones de bloques de disco. La entrada i-ésima del bloque de índice apunta al bloque i-ésimo del archivo. El directorio contiene la dirección del bloque de índice. Para localizar y leer el bloque i-ésimo, se utiliza el puntero contenido en la entrada i-ésima del bloque de índice.
Cuando se crea el archivo, se asigna el valor nulo a todos los punteros del bloque de índice. Cuando se escribe por primera vez en el bloque i-ésimo, se solicita un bloque al gestor de espacio libre y su dirección se almacena en la entrada i-ésima del bloque de índice.
El mecanismo de asignación indexada soporta el acceso directo, sin sufrir el problema de la fragmentación externa, ya que puede utilizarse cualquier bloque del disco para satisfacer una solicitud de más espacio. Sin embargo, el mecanismo de asignación indexada sí sufre del problema del desperdicio de espacio. El espacio adicional

requerido para almacenar los punteros del bloque de índice es aún mayor que el que se requiere en el caso de la asignación enlazada. Considere un caso común en el que se tiene un archivo de sólo uno o dos bloques. Con el mecanismo de asignación enlazada, sólo se pierde el espacio de un puntero por cada bloque, mientras que con el mecanismo de asignación indexada, es preciso asignar un bloque de índice completo, incluso si sólo uno o dos punteros de este bloque van a tener un valor distinto del valor nulo.



6. GESTIÓN DEL ESPACIO LIBRE

Puesto que el espacio del disco es limitado, es necesario reutilizar el espacio de los archivos borrados para los nuevos archivos. Para controlar el espacio libre del disco, el sistema debe llevar el control del espacio libre. Se usan dos métodos principalmente.

6.1. Mapa o vector de bits

La lista de espacio libre se implementa como un mapa de bits o vector de bits. Cada bloque está representado por 1 bit. Si el bloque está libre, el bit será igual a 1; si el bloque está asignado, el bit será 0.

Por ejemplo, considere un disco en el que los bloques 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 17, 18, 25, 26 y 27 están libres y el resto de los bloques están asignados. El mapa de bits de espacio libre sería:

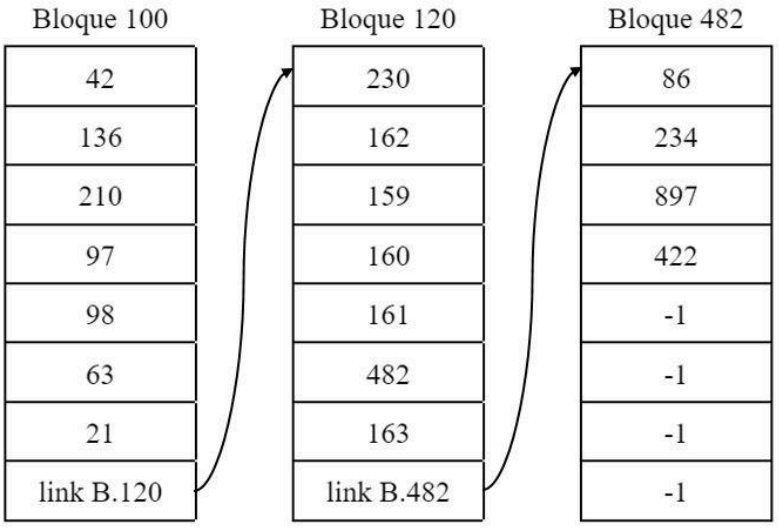
0011110011111100011000000011100000 ...

La principal ventaja de este enfoque es su relativa simplicidad y la eficiencia que permite a la hora de localizar el primer bloque libre o n bloques libres consecutivos en el disco. De hecho, muchas computadoras suministran instrucciones de manipulación de bits que pueden utilizarse de manera efectiva para dicho propósito. Por ejemplo, la familia Intel a partir del 80386 y la familia Motorola a partir del 68020 tienen instrucciones que devuelven el desplazamiento dentro de una palabra del primer bit con el valor 1.

Desafortunadamente, los vectores de bit son ineficientes a menos que se mantenga el vector completo en la memoria principal. Mantener ese vector en la memoria principal es posible para los discos de pequeño tamaño, pero no necesariamente para los discos de tamaño más grande. Un disco de 1,3 GB con bloques de 512 bytes necesitaría un mapa de bits de más de 332 KB para controlar todos los bloques libres, aunque si se agrupan todos los bloques en clusters de cuatro se reduce este número a unos 83 KB por disco. Un disco de 40 GB con bloques de 1 KB requeriría más de 5 MB para almacenar el mapa de bits.

6.2. Agrupamiento de la lista enlazada

Consiste en almacenar las direcciones de n bloques libres (tantos números de bloques de disco como quepan) en el primer bloque libre. Los primeros n-1 de estos bloques estarán realmente libres. El último bloque contendrá la dirección a otro bloque que contendrá otros n bloques libres, etc.



En el siguiente ejemplo se muestran 3 bloques (100, 120 y 482) que contienen los bloques libres del disco, siendo el bloque 100 el bloque inicial para llevar a cabo este conteo.

7. BIBLIOGRAFÍA

Fundamentos de Sistemas Operativos, A. Silberschatz
Sistemas Operativos. Aspectos internos y principios de funcionamiento, W. Stallings
Sistemas operativos modernos. A. Tanenbaum
Sistemas operativos. Un enfoque basado en conceptos. D. Dhamdhere