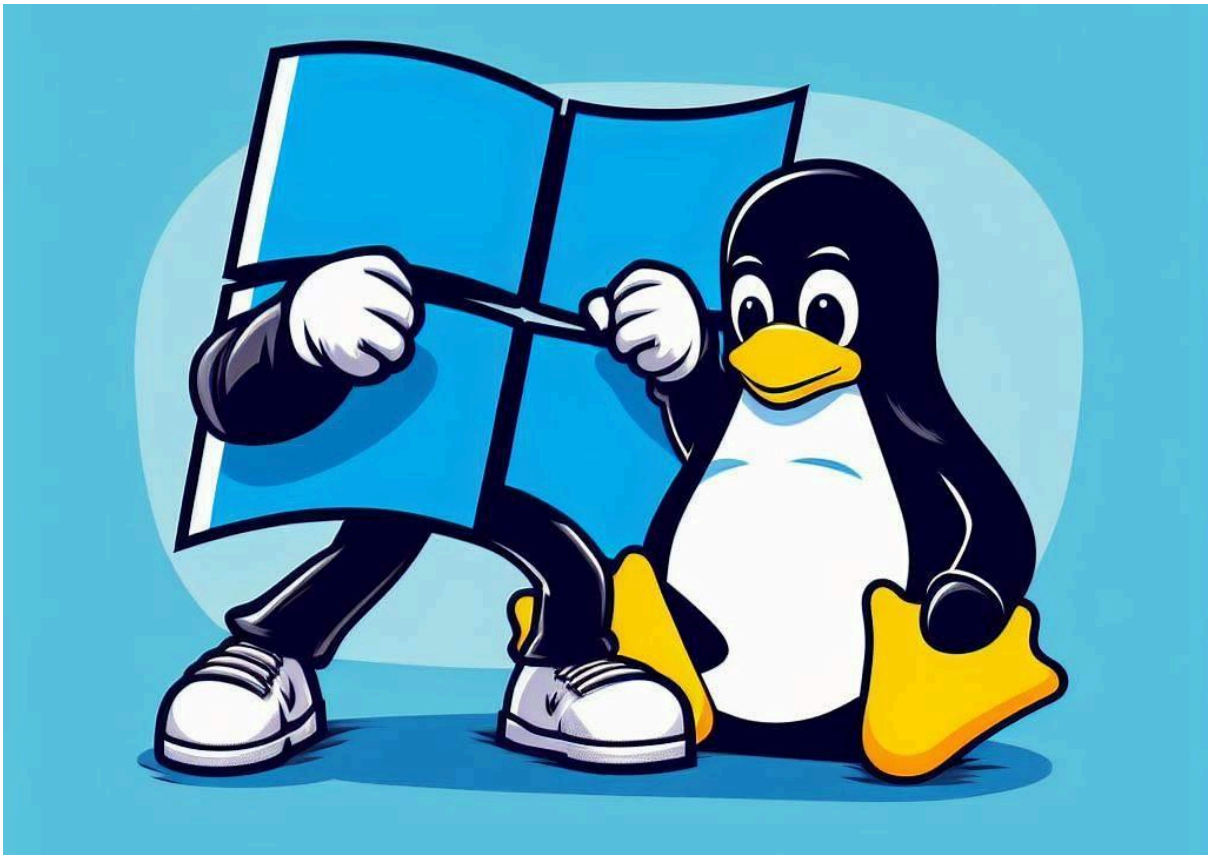
INTRODUCCIÓN A LOS SISTEMAS OPERATIVOS

ADMINISTRACIÓN DE ARCHIVOS

Resumen teoría



Caporal Nicolás

Facultad de Informática

UNLP

2023

ÍNDICE

Clase 1

¿POR QUÉ NECESITAMOS ARCHIVOS?

- Hay una necesidad de almacenar datos
- Tener almacenamiento a largo plazo (NO volatil)
- Permitir a distintos procesos acceder al mismo conjunto de información

CONCEPTO DE ARCHIVO

Es una Entidad abstracta con nombre

Aunque el archivo físicamente este separado en el disco, se debe tener un pensar como un **espacio lógico continuo y direccionable.** Es muy importante el orden de cómo fue armado el archivo

Provee a los programas de datos (archivo de entrada)

Permite a los programas guardar (o modificar) datos (archivo de salida)

El programa mismo es información que debe guardarse en un archivo

ARCHIVOS DESDE EL PUNTO DE VISTA DEL USUARIO

Al usuario le importa:

- Qué operaciones se pueden llevar a cabo
- Cómo nombrar a un archivo
- Asegurar la protección y la integridad
- Compartir archivos entre procesos y/o personas de forma segura
- No tratar con aspectos físicos (el SO debe proveer abstracción de lo físico)

ARCHIVOS DESDE EL PUNTO DE VISTA DEL DISEÑO DEL S.O.

Se espera que:

- Se puedan implementar archivos
- Implementar directorios
- El SO debe manejar y administrar el espacio en disco
- El SO debe manejar el espacio libre (cargar archivos en espacio libre y al borrar un archivo marcar ese espacio como liberado)
- Manejarlo de manera eficiente y realizar mantenimiento

SISTEMA DE MANEJO DE ARCHIVOS

Conjunto de unidades de software (funciones) que proveen los servicios necesarios para la utilización de archivos: (Crear, borrar, buscar, copiar, leer, escribir, etc.)

El sistema de archivos es la parte que se encarga de la administración de los archivos.

- Permite y facilita el acceso a los archivos por parte de los procesos y usuarios

- Permite la abstracción al programador, en cuanto al acceso de bajo nivel (el programador no desarrolla el soft de administración de archivos)

OBJETIVOS DEL S.O. EN CUANTO A ARCHIVOS

- Cumplir con la gestión de datos

- Cumplir con las solicitudes del usuario. (leer, escribir, etc)

- Minimizar la posibilidad de perder o destruir datos → Garantizar la integridad del contenido de los archivos

- Dar soporte de E/S a distintos dispositivos

- Brindar un conjunto de interfaces de E/S para tratamiento de archivos. (tener una forma de interactuar con los archivos)

TIPOS DE ARCHIVOS

Archivos Regulares:

- Texto Plano: Es decir, archivos que puedo abrir y ver (Ej: Source File)

Binarios

- Object File
- Executable File

Directorios

- Un directorio ES UN ARCHIVO que mantienen la estructura en el FileSystem → mantienen información sobre los archivos que están dentro de ese directorio

ATRIBUTOS DE UN ARCHIVO

- Nombre (para el usuario)
- Identificador (internamente para el SO)
- Tipo
- Localización
- Tamaño
- Protección, Seguridad y Monitoreo
    - Owner, Permisos, Password
    - Momento en que el usuario lo modifico, creo, accedio por última vez
    - ACLs (Access Control List)

DIRECTORIOS

Contiene información acerca de archivos y directorios que están dentro de él

El directorio es, en si mismo, un archivo

Contiene entradas, y cada entrada tiene: Nombre de archivo y referencia a I-nodo

Operaciones en directorios (el SO debe proporcionar)**:**

- Buscar un archivo
- Crear un archivo (entrada de directorio)
- Borrar un archivo
- Listar el contenido
- Renombrar archivos
- Etc.

El uso de los directorios ayuda con:

- **La eficiencia:** Localización rápida de archivos
- **Uso del mismo Nombre de archivo** (Diferentes usuarios pueden tener el mismo nombre de archivo)
- **Agrupación:** Agrupación lógica de archivos por propiedades/funciones: (Ejemplo: Programas Java, Juegos, Librerias, etc.)

ESTRUCTURA DE DIRECTORIOS

Los archivos pueden ubicarse siguiendo un path desde el directorio raíz y sus sucesivas referencias (full pathname del archivo o PATH absoluto)

Distintos archivos pueden tener el mismo nombre pero el fullpathname es único

El directorio actual se lo llama "directorio de trabajo (working directory) por eso pwd → print working directory

Dentro del directorio de trabajo, se pueden referenciar los archivos tanto por su PATH absoluto como por su PATH relativo indicando solamente la ruta al archivo desde el directorio de trabajo.

Tanto archivos como directorios se pueden identificar de manera:

- **Absoluta**: El nombre incluye todo el camino del archivo. (Desde '/' en Linux)
- **Relativa**: El nombre se calcula relativamente al directorio en el que se esté (desde el wd)

COMPARTIR ARCHIVOS

En un ambiente multiusuario se necesita que varios usuarios puedan compartir archivos

Debe ser realizado bajo un esquema de protección:

- Derechos de acceso
- Manejo de accesos simultáneos (para garantizar la integridad)

PROTECCIÓN

El propietario/administrador debe ser capaz de controlar:

- Qué se puede hacer: Derechos de acceso
- Quién lo puede hacer

(Con 'chmod' en linux)

DERECHOS DE ACCESO

Los directorios también tienen permisos, los cuales pueden permitir el acceso al mismo para que el usuario pueda usar el archivo siempre y cuando tenga permisos.

Solamente personas autorizadas por el dueño con las operaciones autorizadas deben poder acceder a un archivo.

Posibles operaciones:

- **Execution:** El usuario puede ejecutar
- **Reading:** El usuario puede leer el archivo,
- **Appending:** El usuario puede agregar datos pero no modificar o borrar el contenido del archivo
- **Updating:** El usuario puede modificar, borrar y agregar datos. Incluye la creación de archivos, sobreescribirlo y remover datos (puede borrar datos, pero no el archivo)
- **Changing protection:** El usuario puede modificar los derechos de acceso
- **Deletion:** El usuario puede borrar el archivo
- **Owners (propietarios):**
    - Tiene todos los derechos
    - Pueder dar derechos a otros usuarios. Se determinan clases: Usuario específico Grupos de usuarios Todos (archivos públicos)

Clase 2

CONCEPTOS

Sector: Unidad de almacenamiento utilizada en los Discos Rígidos

Bloque/Cluster: Conjuntos de sectores consecutivos

File System: Define la forma en que los datos son almacenados y cómo son los archivos

FAT (File Allocation Table): Estructura que contiene información sobre en qué lugar están alocados los distintos archivos

PRE-ASIGNACIÓN

Antes se necesitaba saber cuánto espacio va a ocupar el archivo en el momento de su creación porque era obligatorio utilizar sectores contiguos para almacenar los datos de un archivo

Entonces, para no quedarse sin espacio se tendía a definir espacios mucho más grandes que lo necesario. Por lo que obviamente se tenía mucha fragmentación interna en disco

ASIGNACIÓN DINAMICA

El espacio se solicita a medida que se necesita. Por eso es importante la administración del espacio por parte del SO, con un mal manejo se podría asignar un bloque que ya está ocupado por otro archivo

Los bloques de datos pueden quedar de manera no contigua

FORMAS DE ASIGNACION

La asignación de archivos es el método mediante el cual los datos se reparten el espacio de almacenamiento **físico** en el sistema operativo. El kernel asigna espacio de disco a un archivo o directorio en forma de **bloques lógicos**.

ASIGNACIÓN CONTINUA

En la asignación contigua, cada archivo utiliza bloques contiguos en almacenamiento secundario.

Se requiere una pre-asignación (Se debe conocer el tamaño del archivo durante su creación), con lo que los archivos tienden a ser más grandes

La File Allocation Table (FAT) es simple: Sólo una entrada que incluye Bloque de inicio y longitud

El archivo puede ser leído con una única operación

Puede existir fragmentación externa ya que nuestro archivo podría no encontrar la cantidad de bloques contiguos necesarios, eso se soluciona con Compactación, pero es costoso en tiempo, además de requerir un 2do dispositivo

Esta tecnica permite menor latencia por parte del disco.

ASIGNACION ENCADENADA

En la asignación encadenada, cada archivo es una lista enlazada de bloques.

El directorio contiene un apuntador al primer y último bloque.

La asignación se hace con bloques individuales, cada bloque contiene un puntero al siguiente bloque del archivo.

La FAT tiene una Única entrada por archivo: Bloque de inicio y tamaño del archivo

No hay fragmentación externa, se aprovecha todo el espacio

Útil para acceso secuencial (no random)

Los archivos pueden crecer bajo demanda

No se requieren bloques contiguos (podrían estarlo)

Se pueden consolidar los bloques de un mismo archivo para garantizar cercanía de los bloques de un mismo archivo.

ASIGNACION INDEXADA

La FAT contiene un puntero al **bloque** índice.

El **bloque índice** no contiene datos propios del archivo, sino que contiene un índice a los bloques que lo componen. (en orden)

Asignación en base a bloques individuales → No se produce Fragmentación Externa

El acceso "random" a un archivo es **eficiente.** Tengo acceso directo a cualquier parte del archivo

La File Allocation Table tiene una Única entrada con la dirección del bloque de índices (index node / i-node).

VARIANTE: Asignación por secciones

A cada entrada del bloque índice se agrega el campo longitud

El índice apunta al primer bloque de un conjunto almacenado de manera contigua.

VARIANTE: Niveles de indirección

Existen bloques directos de datos

Otros bloques son considerados como bloque índices (apuntan a varios bloques de datos)

Puede haber varios niveles de indirección (es decir, algunos bloques directos, otros con indirección simple, otros indirección doble y así)

GESTION DE ESPACIO LIBRE

El file system debe llevar el control sobre cuáles de los bloques de disco están disponibles. Alternativas Tablas de bits Bloques libres encadenados Indexación

TABLA DE BITS

Se lleva una tabla que representa cada sector (cluster) con un bit que representa si está ocupado (1) o libre (0)

Ventaja: Fácil y rápido encontrar un bloque o grupo de bloques libres.

Desventaja: Tiene que estar en memoria principal para usarla y el tamaño del vector en memoria puede ser considerablemente grande, ya que es una entrada en la tabla (un bit) por cada bloque. Luego de usarla en memoria, hay que escribirla en Disco (lento)

Se calcula: tamaño disco bytes / tamaño bloque en sistema archivo

Ejemplo: Disco 16 Gb con bloques de 512 bytes 32 Mb

BLOQUES ENCADENADOS

Se encadenan los bloques libres.

Se tiene un puntero al primer bloque libre. Cada bloque libre tiene un puntero al siguiente bloque libre

Desventajas:

- Ineficiente para la búsqueda de bloques libres → Hay que realizar varias operaciones de E/S para obtener un grupo libre.
- Si pierdo un enlace, se rompe la cadena.
- Es difícil encontrar bloques libres consecutivos

INDEXACIÓN (o agrupamiento)

Es una variante de “bloques libres encadenados”

El primer bloque libre contiene las direcciones de muchos (N) bloques libres.

Las N-1 primeras direcciones (todas menos la última) son bloques libres.

La N-ésima dirección (la última) referencia otro bloque indice con N direcciones de bloques libres.

Ventaja: Se encuentra rápido un conjunto de bloques libres

Desventaja: Es costoso mantener esta estructura. Si ocupo un conjunto de bloques, hay que hacer un desplazamiento en los sucesivos bloques indices para ocupar el primero y que se sigan cumpliendo propiedades dichas previamente

Clase 3

TIPOS DE ARCHIVOS EN UNIX

- **Archivo común**
- **Directorio:** Es un archivo que mantiene la relación entre el nombre (string) y el I-nodo
- **Archivos especiales** (dispositivos /dev/sda)
- **Named pipes** (comunicación entre procesos)
- **Links** (comparten el i-nodo, solo dentro del mismo filesystem)
- **Links simbólicos** (tiene i-nodo propio, para filesystems diferentes)

¿CÓMO ES UN SISTEMA DE ARCHIVOS EN UNIX?

Cada disco físico puede ser dividido en uno o mas volúmenes. Cada volumen o partición contiene un sistema de archivos.

Cada sistema de archivos contiene:

- **Boot Block:** Código para bootear el S.O.
- **Superblock:** Atributos sobre el File System → Bloques/Clusters libres
- **I-NODE Table:** Tabla que contiene todos los INODOS
- **I-NODO:** Estructura de control que contiene la información clave de un archivo (tiene todo menos el nombre, que está en el directorio)
- **Data Blocks:** Bloques de datos de los archivos

Se usa Asignación Indexada con varios niveles de indirección