

Chapter 2: File System

Operating system

TS. DO TIEN DUNG

dungdt@ptit.edu.vn



Posts and Telecommunications
Institute of Technology
Faculty of Information Technology 1



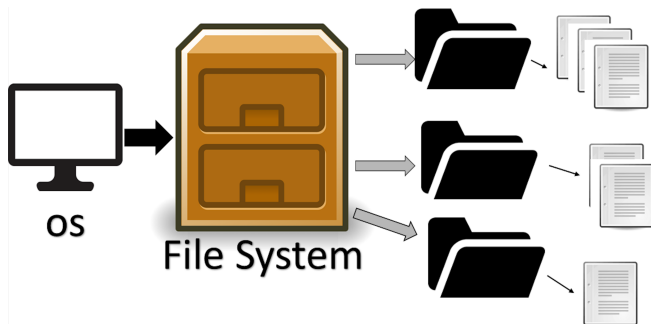
Faculty of Information Technology
Post and Telecommunication Institute of Technology

August 15, 2023

Inside folder Organizer

1. Definitions
2. File access methods
3. File Operations (các thao tác với file)
4. Folder
5. File volume allocation
6. Disks management
7. File systems robustness
8. File systems security
9. FAT files system

- *File is defined as a collection of related information, which is named and stored in external memory.*

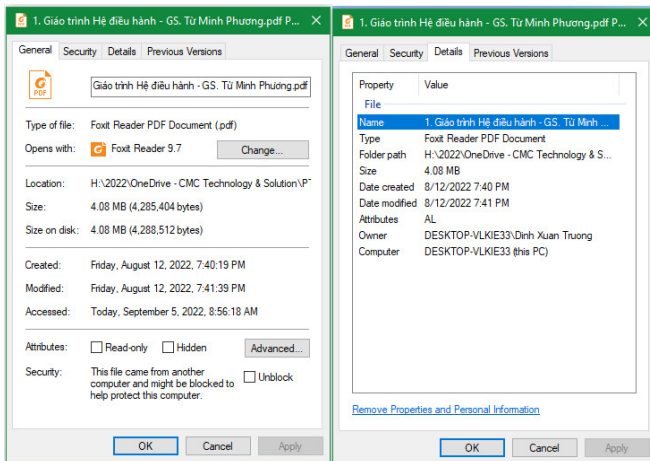


Hình 2: Files system

- *File attribute:* to manages files based on content, OS also defines attributes, properties (thuộc tính, tính chất).

► File attributes:

- File name
- File type
- File volume
- File creator, owner
- File access permission
- Thời gian tạo file, sửa file, truy cập lần cuối File created, edit time, last edited time
- File location



Hình 3: File attributes

► File name:

- Identify files.
- Most used information when users work with files.
- Rules of file names:

Hệ điều hành	Độ dài tối đa	Phân biệt chữ hoa, chữ thường	Cho phép sử dụng dấu cách	Các ký tự cấm
MS-DOS	8 cho tên file 3 cho mở rộng	không	không	Bắt đầu bằng chữ cái hoặc số Không được chứa các ký tự / \ [] : ; = , ^ ? @
Windows NT FAT	255 ký tự cho cả tên file và đường dẫn	không	có	Bắt đầu bằng chữ cái hoặc số Không được chứa các ký tự / \ [] : ; = , ^ ? @
Windows NT NTFS	255	không	có	Không được chứa các ký tự / \ < > * :
Linux (EXT3)	256	Có	có (nếu tên file chứa trong ngoặc kép)	Không được chứa các ký tự ! @ # \$ % ^ & * () [] { } ' " \ ; < > ' :

Hình 4: Rule of file name

- ▶ Information inside files could be different.
- ▶ Some files have unstructured information: text file. Structured files such as: database file, file excel.
File structure is also different and depends on the information of files
- ▶ OS need to know and support file structures?
- ▶ Support file structure at OS level:
 - Advantages:
 - ▶ File operations will be easier for application programmers.
 - ▶ OS can control file operations.
 - Disadvantages:
 - ▶ Size up the system volume/
 - ▶ Reduce the flexibility of system.
- ▶ In reality, OS consider files as collection of unstructured byte.

- ▶ Most of OS do not support and manage structured file.
- ▶ The file structure is managed by the application program and users.
- ▶ In UNIX, DOS, WINDOWS, files are considered as collection of bytes.
- ▶ Application programs will create and manage their own file structures.
For instance: *The graphical program saves the file as an unzipped binary, the data management system program creates a file that includes logs.*

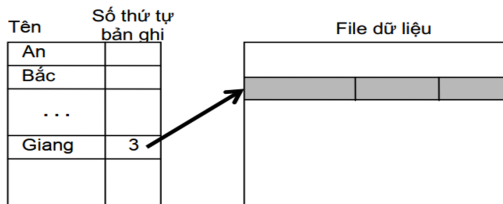
- ▶ To read/write files, OS must specify how to access the file contents. Each OS supports one or more different access ways.
- ▶ **Sequential access:**
 - Information is read/write by byte/loge from the file beginning.
 - Use pointer to allocate the current position in files.
 - After one read/write operation, the pointer position will increase one to next position.
 - This type of access is suitable for files stored on magnetic tape. Information is written or read in the direction of the tape rotation.

► Direct access:

- File is considered as numbered blocks/logs.
- Blocks can be accessed by arbitrary order.
- For instance: it can read 50th block, then 13th and 101th.
- Direct access is based on disk access characteristics, which allows accessing to any block.
- File is allocated at different blocks in disk, which allows non-sequential access.

► Index-based access:

- Allows access to logs in the file, not by the number or position of log, but by the corresponding key to that log.
- File has one private index: includes keys and pointer to logs in files.
- Access: find corresponding key in index list, then pointer identify logs and access directly to it.



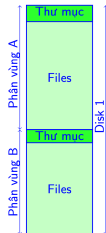
Hình 5: Index-based access

- ▶ OS regulate operations that users and applications can work with files.
- ▶ Most used operations are:
 - **Create file:** create empty file; File is allocated the space with attributes such as created time, name, user.
 - **Delete file:**
 - ▶ Free up the space that file takes in the disk.
 - ▶ Free up the file in folder.
 - ▶ Free up the space can simply mark as free space.
 - **Open file:**
 - ▶ Before read/write file.
 - ▶ Read file attributes in the disks to speed up the next read/write operation.
 - **Close file:**
 - ▶ Delete file information in the memory table to give a space for coming file.

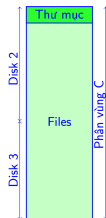
- ▶ Many OS restrict the number of opened files at the same time, so close the accessed file is important.
- **Write to file:**
 - ▶ Data is written at the current position in file.
 - ▶ If the current position is at the end of file, information is added and the file volume increase.
 - ▶ If the current position is not at the end, information will be overwritten.
- **Read file** Information at current position will be read.
- **seek** Identify the current file position.
- **Read file attributes:** Some application need to read the file attributes such as access permission for security purpose.
- **Change file attributes:** Reset some file attributes.
- **Delete file content:** Delete file content, release the space volume that file occupies, but keep file attributes except length.

- **Lock file:** When many processes change file content, it may result in unexpected results, so OS allow locking file. When a process opens file, it can request to lock file and do not allow other processes access, read or write that file.

- ▶ The number of files stored in the disks is huge => need to organize files for easy management and access.
 - ▶ Disk is assigned into partition/volume (logic disk).
- * Disk is assign into:
Partitions, Minidisks, Volumes.
- * Each partition is separated storage area. It can have OSs.

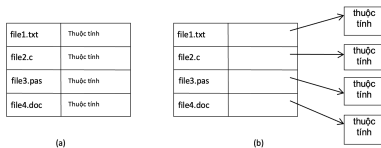


- ▶ Combine some disks into a big logic disk.
 - * Users only care about file and folder logic.
 - * They do not care about the disk partition management.



- ▶ To manage files in disk logic, file information is stored in folder.
- ▶ **Folder** is formed from *entries* and each entry is one file.
- ▶ *Entry* include file information: name, capacity, location, file type or pointer of storage information.
- ▶ Folder as a table, each row is an entry. Finding necessary row based on file name. Folder allow mapping file name to that row.
- ▶ Storage methods:

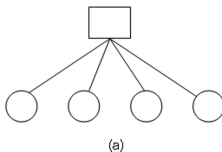
- All file properties are stored in folder. File contain content only => volume of entries, folder is big.
- A part of properties is stored with file content. Folder only store necessary information for finding file location => volume is decreased



Hình 6: File properties storage a) With folder b) With file

- ▶ Open file:
 - OS find in folder/entry mapping to file name.
 - Read properties and location of file in table containing opened files.
 - If entry mapping to another file structure of file properties, this structure will be written in the table.
- ▶ File searching: folder structure allow searching with file name.
- ▶ File create: create new entry and add in folder.
- ▶ File delete: information of file and entry will be deleted from folder.
- ▶ Folder list: list files and information of that entry.
- ▶ Change name: only work with folder, not file content.

- ▶ One level folder:
 - Most simplest
 - Only one folder and all files are stored in this folder.
 - It is difficult to choose file name.
 - It is difficult to search file.

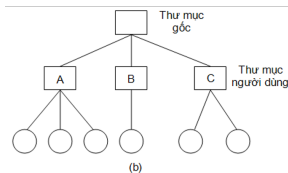


□ = Thư mục ○ = File

Hình 7: One level folder structure

► Two level folder:

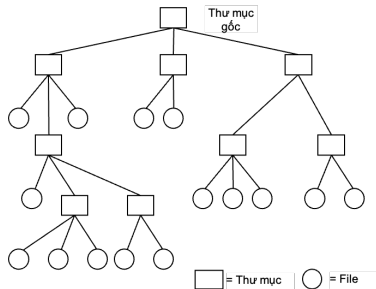
- Assign each user their own folder(UFD: User File Directory) containing their files.
- When user access file, file will be search in their name folder.
- User can name files with same name.
- Isolate users. Files that share with many users => copy to all user folders => waste of resource.



Hình 8: Two level folder structure

► Tree structure folder:

- Folder can contain other smaller folder and files.
- Folder structure is shown as tree structure: branch is folder and leaf is file.

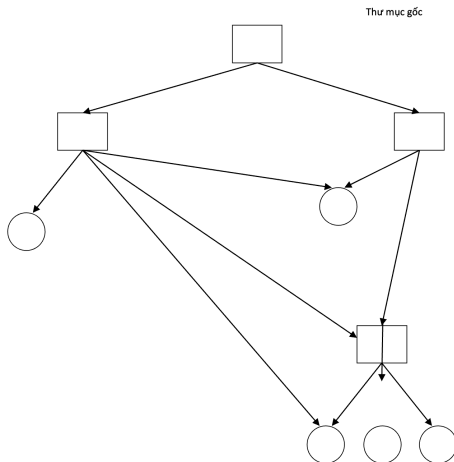


Hình 9: Tree structure folder

► **Tree structure folder:**

- Distinguish file entry and small folder entry: add special bit in entry
 - 1: folder entry
 - 0: file entry
- Each time, user work with current directory
- Organize tree structure folder for each disk:
 - In file systems like FAT, DOS, tree folder is structured for disk. Folder system is forest, each tree in one disk.
 - Linux: whole system contain one tree.

► Acyclic graph:



- Files and folder can be at different folders.

- This is a non-cyclic graph, and a extension of tree structure.
- Tree: leaf and branch may belong to other branches.
- Deployment:
 - ▶ Use Sử dụng liên kết: con trỏ tới thư mục hoặc file khác
 - ▶ Create the copies of shared files and store in different folder => ensure the synthesis and consistency => avoid cross back-up, store one file many times.
- Flexible but complicated

- ▶ Describe file location in folder
- ▶ Absolute directory:
 - Directory from root folder, to intermediate folder, file.
For ex: C: -> bc -> bin -> bc.exe
- ▶ Relative Directory:
 - From the current folder Tính từ thư mục hiện thời
 - Add 2 special entries in folder: "." represents current folder, và ".." for upper folder.

► List:

- Organize folder as the list of entries.
- Search entries by scanning the list
- Add new file in folder:
 - Scan folder to check the existing of file name
 - New entry is added at the end of the list or one cell in the table
- Open, delete file
- Search in the list (danh sách chậm)
- Store folder in the memory

► Binary tree:

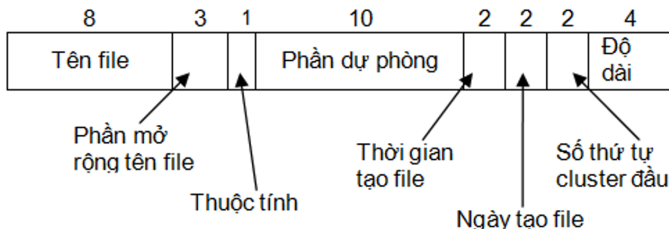
- Increase searching speed thanks to structured data.
- NTFS file system of WinNT

► hash table:

- Use hash function to find location of entry in folder by file name.
- Quick searching time
- Hash function depends on the size of hash table => fixed hash table size.

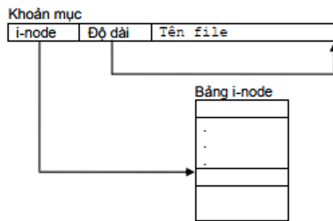
► Folder structure of DOS:

- Each logic disk has its own folder, starting from ROOT folder.
- ROOT folder is placed at the beginning of disk, right after BOOT sector and FAT table.
- ROOT folder contains files and other child folders.
- Child folder can have files and other lower level folders.
- Organize by table: each entry is one row in table and has fixed size of 32 bytes.



► Folder structure of Linux:

- Folder system Ext2 in Linux is simple.
- Entry has file name and I-node address.
- Other information of file properties and location of data block are stored in I-node, not in folder.
- Entry size depends on length of file name.
- The headers of entry has information of entry size.



Chapter 2

- ▶ Definitions
- ▶ File access methods
- ▶ File operations
- ▶ Folder

Next

- ▶ Cấp phát không gian cho file
- ▶ Quản lý không gian trống trên đĩa
- ▶ Độ tin cậy của hệ thống file
- ▶ File system security
- ▶ File system FAT