DOI: https://doi.org/10.53555/nncse.v1i4.513 Publication URL: http://nnpub.org/index.php/CSE/article/view/513

FILE SYSTEM

Priyanka Sahni[1]*, Nonika Sharma[2]
*[1,2]*Information Technology Dronacharya College of Engineering, Gurgaon*
[1]priyanka.sahni@yahoo.com.au, *[2]Nonikasharma1@gmail.com*

***Corresponding Author*: -**
Email ID- priyanka.sahni@yahoo.com.au

Abstract: -

A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the file's creator and user. In our research paper we're going to focus on File Structure, File Type, File Names, Pathnames.

File Structure - It is a structure, which is according to a required format that operating system can understand. File types also can be used to indicate the internal structure of the file.
File Type – It refers to the ability of the operating system to distinguish different types of files such as text files, source files and the binary files etc. Many Operating system supports many types of files. Operating system like MS-DOS and UNIX have following types of files: Ordinary files
Directory files
Special files
File Operations - A file is an abstract data type. To define a file properly, we need to consider the operations that can be performed on files. The operating system can provide system calls to create, write, read, reposition, delete and truncate files.
File Access Mechanisms - It refers to the manner in which the records of a file may be accessed. There are several ways to access file
1-Sequential access
2-Direct/ Random access
3-Indexed sequential access

INTRODUCTION

All file systems consist of structures necessary for storing and managing data. These structures typically include an operating system boot record, directories, and files.

Functions of a File System:

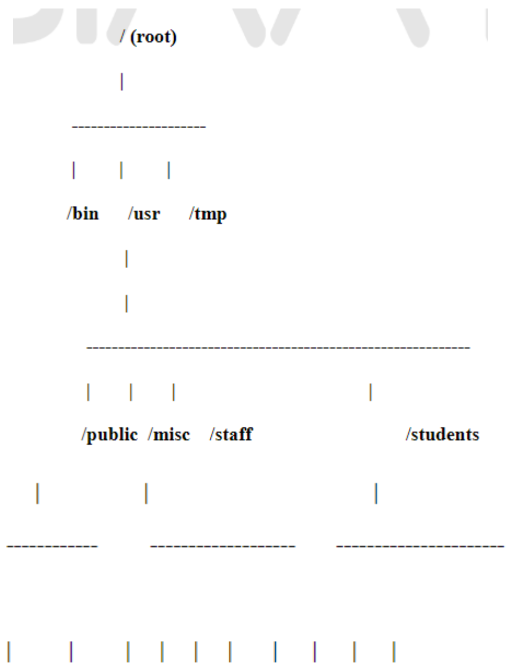
1. Tracking allocated and free space
2. Maintaining directories and file names
3. Tracking where each file is physically stored on the disk

Different file systems are used by different operating systems. Some operating systems can recognize only one file system, while other operating systems can recognize several.

BASICS OF FILE STRUCTURES

File Structure

- All files in the UNIX file system are organized in a multi-leveled hierarchy called a directory tree.
- A family tree is an example of a hierarchical structure that represents how the UNIX file system is organized. The UNIX file system might also be envisioned as an inverted tree or the root system of plant.
- At the very top of the file system is single directory called "root" which is represented by a / (slash). All other files are "descendents" of root.
- The number of levels is largely arbitrary, although most UNIX systems share some organizational similarities. The "standard" UNIX file system is discussed later.
- Example**:**



software/doc/john /mary /bill /carl/tom/dick/mary/lisa

Exercise: Type cd /to go to the root directory of our system. Using the ls and cd commands explore the directory structure.

File Types

The UNIX file system contains several different types of files:

- **Ordinary Files**
  - Used to store your information, such as some text you have written or an image you have drawn. This is the type of file that you usually work with.
  - Always located within/under a directory file
  - Do not contain other files

- **Directories**
  - Branching points in the hierarchical tree
  - Used to organize groups of files
  - May contain ordinary files, special files or other directories
  - Never contain "real" information which you would work with (such as text). Basically, just used for organizing files.
  - All files are descendants of the root directory, (named /) located at the top of the tree.

- **Special Files**
  - Used to represent a real physical device such as a printer, tape drive or terminal, used for Input/Ouput (I/O) operations

- Unix considers any device attached to the system to be a file - including your terminal:
  - By default, a command treats your terminal as the standard input file (stdin) from which to read its input
  - Your terminal is also treated as the standard output file (stdout) to which a command's output is sent
  - Stdin and stdout will be discussed in more detail later
- Two types of I/O: character and block
- Usually only found under directories named /dev

- **Pipes**
  - UNIX allows you to link commands together using a pipe. The pipe acts a temporary file which only exists to hold data from one command until it is read by another
  - For example, to pipe the output from one command into another command
  - **who | wc -l**

This command will tell you how many users are currently logged into the system. The standard output from the who command is a list of all the users currently logged into the system. This output is piped into the wc command as its standard input. Used with the -l option this command counts the numbers of lines in the standard input and displays the result on its standard output - your terminal.

File Names

- UNIX permits file names to use most characters, but avoid spaces, tabs and characters that have a special meaning to the shell, such as:
  & ; ( ) | ? \ ' " ` [ ] { } < > $ - ! /
- Case Sensitivity: uppercase and lowercase are not the same! These are three different files:
  NOVEMBER November november
- Length: can be up to 256 characters
- Extensions: may be used to identify types of files
  libc.a - archive, library file
  program.c - C language source file
  alpha2.f - Fortran source file
  xwd2ps.o - Object/executable code
  mygames.Z - Compressed file
- Hidden Files: have names that begin with a dot (.) For example:
  .cshrc .login .mailrc .mwmrc
- Uniqueness: as children in a family, no two files with the same parent directory can have the same name. Files located in separate directories can have identical names.
- Reserved Filenames:
  / - the root directory (slash)
  . - current directory (period)
  .. - parent directory (double period)
  ~ - your home directory (tilde)

Pathnames

- Specify where a file is located in the hierarchically organized file system
- Must know how to use pathnames to navigate the UNIX file system
- Absolute Pathname: tells how to reach a file begining from the root; always begins with / (slash). For example:

/usr/local/doc/training/sample.f

- Relative Pathname: tells how to reach a file from the directory you are currently in (current or working directory); never begins with / (slash). For example:
  training/sample.f ../bin ~/projects/report.001
- For example, if your current directory is /usr/home/johnson and you wanted to change to the directory /usr/home/quattro, you could use either of these commands:
  cd ../quattro - relative pathname
  cd /usr/home/quattro - absolute pathname