Directory

Directories

Files in an OS are generally named using a hierarchical naming system based on directories. Almost all OS use a hierarchical naming system. In such a system, a path name consists of component names separated with a separator character.

- A directory is a name space and the associated name maps each name into either a file or another directory.
- Directories contain bookkeeping information about files.
- Directories are usually implemented as files and in OS point of view there is no distinction between files and directories.

Directory operations

- Create: Produces an ``empty'' directory. Normally the directory created actually contains . and .., so is not really empty
- Delete: Requires the directory to be empty (i.e., to just contain . and ..). Commands are normally written that will first empty the directory (except for . and ..) and then delete it. These commands make use of file and directory delete system calls.
- Opendir: Same as for files (creates a ``handle'')
- Closedir: Same as for files
- Readdir: In the old days (of unix) one could read directories as files so there was no special readdir (or opendir/closedir). It was believed that the uniform treatment would make programming (or at least system understanding) easier as there was less to learn.
- Rename: As with files
- Link: Add a second name for a file;
- Unlink: Remove a directory entry. But if there are many links and just one is unlinked, the file remains.

- The structure of the directories and the relationship among them are the main areas where file systems tend to differ, and it is also the area that has the most significant effect on the user interface provided by the file system.
- The most common directory structures used by multi-user systems are:
  - **Single-level directory**
  - **Two-level directory**
  - **Tree-structured directory**
  - **Acyclic directory**

Single-Level Directory

- In a single-level directory system, all the files are placed in one directory. This is very common on single-user OS's.
- Limitations
  - Unique name is a problem

    More number of files

    More than one user

Two-Level Directory

- In the two-level directory system, the system maintains a master block that has one entry for each user. This master block contains the addresses of the directory of the users.

Limitations

- This structure effectively isolates one user from another. This is an advantage when the users are completely independent, but a disadvantage when the users want to cooperate on some task and access files of other users.

Tree-Structured Directory

- In the tree-structured directory, the directory themselves are files. This leads to the possibility of having sub-directories that can contain files and sub-subdirectories.

Limitations

- An interesting policy decision in a tree-structured directory structure is how to handle the deletion of a directory.
- If a directory is empty, its entry in its containing directory can simply be deleted.
- However, suppose the directory to be deleted id not empty, but contains several files, or possibly sub-directories.
- Some systems will not delete a directory unless it is empty.

Acyclic-Graph Directory

- The acyclic directory is an extension of the tree-structured directory structure. In the tree-structured directory, files and directories starting from some fixed directory are owned by one particular user.
- In the acyclic structure, this prohibition is taken out and thus a directory or file under directory can be owned by several users.

Path Name

- An absolute path name is a name that gives the path from the root directory to the file that is named.
- When hierarchical file system is used, the absolute path may be long if many levels of subdirectories are present.
- Hierarchical naming system use compound names with several parts. Each part is looked up in a flat name space usually called a directory. If this look up leads to another directory, then the next part of the compound name is looked up in that directory.
- To reduce the length of the path name, working/current directory concept is used. This allows the use of relative path names.
- A relative path name is a path name that starts at the working directory rather than the root directory.

Aliases

- Hierarchical directory systems provide a means for classifying and grouping files.
- All the files in a single directory tend to be related in some way, but sometimes a file is related to several different groups and it is an inconvenient restriction to have no place it in just one group.
- This problem can be handled with the concept of a file alias. It is a file name that refers to a file that also has another name. There may be two or more absolute path names.

File Access Mechanisms

File access mechanism refers to the manner in which the records of a file may be accessed. There are several ways to access files –

Sequential access

Direct/Random access

Indexed sequential access

Sequential access

A sequential access is that in which the records are accessed in some sequence, i.e., the information in the file is processed in order, one record after the other. This access method is the most primitive one. Example: Compilers usually access files in this fashion.

Direct/Random access

Random access file organization provides, accessing the records directly.

Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing.

The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.

Indexed sequential access

This mechanism is built up on base of sequential access.

An index is created for each file which contains pointers to various blocks.

Index is searched sequentially and its pointer is used to access the file directly.