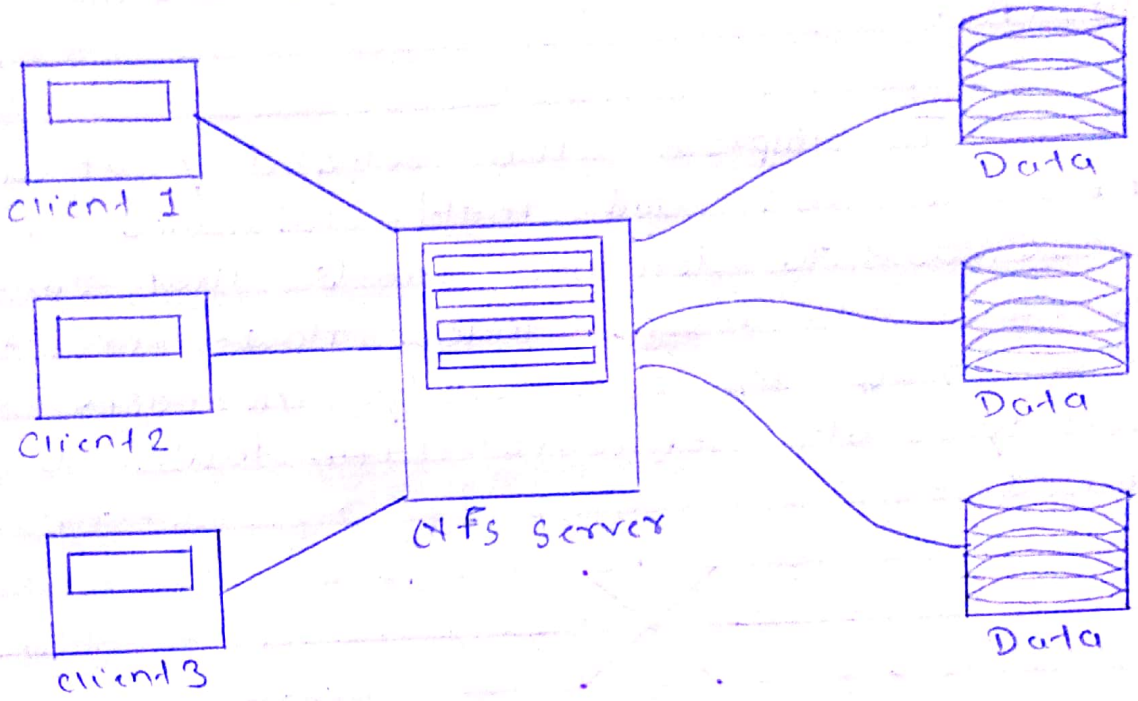

# CASE STUDY NO : 09

| Date | / /20 |
|---|---|
| Page No. | 44 |

Case Study On :- Study file system with design and implementation.

File system design :-

A distributed file system typically has two reasonably distinct components : the true file services and the directory service. The former is concerned with the operations on individual files, such as reading, writing and appending, where as the ladder is concerned with creating and managing ~~directions~~ directories adding and deleting files from directories and so on. In this section we will discuss the true file service interface. In the next one we will discuss the directory service interface.

The file service Interface -

For any file service, whether for a single processor or a distributed system, the most fundamental issue is : What is file ? In many systems, such as UNIX and MS-DOS, a file is an uninterpreted sequence of bytes. The meaning and structure of the information in the files is entirely up to the application programs the operating system is not interested.

A files can have attributes, which are pieces of information about the file but which are not part of the file itself. Typical attributes are the owner, size, creation date

and access permission. The file service usually provides primitives to read and write some of the attributes for ex. it may be possible to change the access permissions but not the size. In a few advanced systems, it may be possible to create and manipulate user-defined attributes in addition to the standard ones.

The Directory Server Interface -

The other part of the file service is the directory service, which provides operations for creating and deleting directories, naming and renaming files and moving them from one directory to another. The nature of the directory service does not depend on whether individual files are transferred in their entirely or accessed remotely.

The directory service defines some alphabet and syntax for composing file names. File names can typically be from 1 to some maximum number of letters, numbers and certain special characters. Some system divides file names into two parts usually separated by a period, such as prog.c for a c program or man.txt for a text file. The second part of the file name, called the file extension, identifies the file type. other systems use an explicit attribute for this purpose, instead of tacking an extension onto the name.

| Date | / /20 |
| --- | --- |
| Page No. | 46 |

## Distributed File system Implementation -

In the preceding section, we have described various aspects of distributed file systems from the user's perspective, that is how they appear to the user. In this section we will see how these system are implemented. We will start out by presenting some experimental information about the file usage. Then we will go on to look at system structure the implementation of caching, replication in distributed systems, the implementation of caching, replication in distributed systems and concurrency control. We will conclude with a short discussion of some lessons that have been learned from experience.

We can implement file system by using two types data structures.

1] On-disk structures -

Generally they contain information about total number of disk blocks, free disk blocks, location of them and etc. Below given are different on-disk structures:

<1> Boot Control Block -

It is usually the first block of volume and it contains information needed to boot an operating system. In UNIX it is called boot block and in NTFS it is called as partition boot sector.

(2) Volume Control Block -

It has information about a particular partition ex- free block count, block size and block pointers etc. In UNIX is called super block and in NTFS it is stored in master file table.

(3) Directory Structure -

They store file names and associated inode numbers. In UNIX, includes file names and associated file names and in NTFS, it is stored in master file table.

(4) Per-file FCB -

It contains details about files and it has a unique identifier number to allow association with directory entry. In NTFS it is stored in master file table.

* In-memory structure -

They are maintained in main memory and these are helpful for file system management for caching. Several in memory structures given below.

1) Mount Table - It contains information about each mounted volume.

2) Directory - structure cache - The cache holds the directory information of recently accessed directories.

3) System wide open - file table - It contains the copy of FCB of each open file.

4) Per - process open - file table - It contains information opened by that particular process and it maps with appropriate system wide open - file.

★ Advantages -

1) DFS allows multiple user to access or store the data.
2) It allows the data to be share remotely.
3) It improved the availability of file, access time, and network efficiency.
4) Improved the capacity to change the size of the data and also improves the ability to exchange the data.
5) Distributed file system provides transparency of data even if server or disk fails.

★ Disadvantages -

1) In distributed file system nodes and connections needs to be secured therefore we can say that security is at stake.
2) There is a possibility of lose of messages and data in the network while movement from one

node to another.

3) Database connection in case of distributed file system is complicated.

Conclusion -

We have described the operation, performance and convenience of a ~~file system~~ transparent adaptive mechanism for file system discovery and replacement. Such a mechanism is generally useful, but offers particularly important support for mobile computers with may experience difference in response time as a result of their movement.