

programs would have to contain information about which file system formats they support, but with self-describing file systems, these programs can be made portable and can support many different formats just by supporting the ability to interpret the formal description.

Performing “invisible” reads and writes is not the only motivation for reading the disk directly. By bypassing the file system driver, a backup program can improve its performance. Even more performance increases can be realized during incremental backups. Instead of searching the file system, the on-disk inode table can be scanned for candidate files. Scanning a table is considerably faster than searching directories and checking modification times by calling stat (2) on each file.

Another benefit of self-describing file systems is that most file system utility commands can be made independent of the file system type by using the file system’s formal description. Unix commands such as df, fstyp, labelit, and ncheck could be written once instead of developing a different version for each supported file system format.

Self-describing file systems allow applications to be developed that can understand multiple file system formats in an extensible and flexible manner. When a file system format evolves, applications do not need to be modified. Instead, only the formal description of the file system needs to be changed. Additionally, a single application can work with different file system formats, because it can adapt dynamically based on the file system’s formal description.

While the invention has been described with reference to the preferred embodiment thereof it will be appreciated by those of ordinary skill in the art that modifications can be made to the parts that comprise the invention without departing from the spirit and scope thereof.

I claim:

1. A method for use in a file system including at least one server and one disk storage device for access by at least one client, said method comprising the steps of:

attaching said client to said file system; and

reading a formal description of the file system by said client from said disk storage device, wherein the formal description of the file system enables said client to find and interpret at least one data structure that includes file information that enables the client to directly read and write data to and from said disk storage, block allocation for the data being performed by the server, and wherein the formal description of the file system lacks a data structure that includes file information.

2. The method of claim 1, wherein reading a formal description further comprises:

reading enough information to find and interpret the physical block and offset containing an inode of a given file given the inode number of the inode of the given file.

3. The method of claim 2, wherein reading a formal description further comprises:

reading enough information to find and interpret the block list of a given file given an offset into the file and a length.

4. The method of claim 3, wherein attaching said client to a file system comprises:

sending a mount request; and  
receiving a mount response.

5. The method of claim 4, further comprising saving said formal description for future use when an I/O request is made by said client.

6. The method of claim 5, further comprising associating said disk storage device with a Storage Area Network (SAN).

7. The method of claim 5, wherein said client and said server are implemented on the same hardware.

8. The method of claim 1, wherein the formal description of the file system includes an algorithm used to implement the file system.

9. The method of claim 1, wherein the formal description of the file system does not include an algorithm used to implement the file system.

10. A method for reading or writing data from a storage resource, the method comprising:

acquiring, from the storage resource, a description of a file system associated with the storage resource, wherein the description of the file system enables a client to find and interpret at least one data structure having file information, and wherein the description of the file system lacks a data structure that includes file information; and

finding and interpreting at least one data structure that includes file information for reading or writing directly to the storage resource based on the file information, block allocation for the data being performed by a server.

11. The method of claim 10, further comprising reading or writing data blocks associated with the file system.

12. The method of claim 10, further comprising:

on the basis of a file identifier, finding and interpreting a block and an offset, the block and the offset being associated with a file on said file system.

13. The method of claim 12, further comprising:

on the basis of an offset into a file and a length, finding and interpreting a block list associated with the file.

14. The method of claim 10, wherein the description of the file system includes an algorithm used to implement the file system.

15. The method of claim 10, wherein the description of the file system does not include an algorithm used to implement the file system.

16. An apparatus for reading and writing data from a storage resource, the apparatus being configured to:

acquire, from the storage resource, a description of a file system associated with a storage resource, wherein the description of the file system enables a client to find and interpret at least one data structure having file information, and wherein the description of the file system lacks a data structure that includes file information; and to

find and interpret at least one data structure that includes file information for reading or writing directly to the storage resource based on the file information, block allocation for the data being performed by a server.

17. The apparatus of claim 16, further comprising a computer configured to read and write data blocks associated with the file system.

18. The apparatus of claim 16, further comprising a computer configured to read and write files associated with the file system.

19. The apparatus of claim 18, wherein the computer is configured to find and interpret, on the basis of a file identifier, a block and offset associated with a file.

20. The apparatus of claim 18, wherein the computer is configured to find and interpret, on the basis of an offset into the file and a length, a block list associated with a file.