Operating Systems Project 4

Emre Derman

21703508

Department of Computer Engineering
Bilkent University

Introduction

In this project, I implemented a simple file system which consists of a single directory and files.

The System Structures

File System architecture consists of blocks and entries. First block considered as superblock which holds the system informations and the structure of the blocks are follows

1-number of blocks (1 x 4)bytes

2-inode no (1 x 4)bytes

the rest is idle.

Superblock followed by bitmap blocks which are shows the availability status of the blocks. The bitmap approach has been implemented. The corresponding block information has been reached by using shift operation and the structure of the blocks are follows

1- unsigned char array has been used with a size of 4096.

(4096 x 1) bytes

bitmap blocks followed by 4 directory blocks which contains 32 directory entry array and the structure of the blocks are follows:

1- filename [110] (1x110) bytes

2- inode no (1x4) bytes

3- size (1x4) bytes

Total 128 directory consists in my file system which are matched with inodes. Since the directory entry blocks followed by the file control block nodes which are holds the data of the file and the structure of the blocks are follows:

1-inodeNo (1 x 4)bytes

2-indexNodeNo (1 x 4)bytes

3-location_pointer(holds the read / written part of the file) (1 x 4)

4-mode(open mode) (1 x 4)bytes

5-usedStatus (1 x 4)bytes

Lastly I used index blocks for each inode. The index block holds the corresponding data block block numbers inside the corresponding file.

The size of the index_block is 4096 and the structure of the blocks are follows:

1-int block_numbers[1024] (4 x 1024)bytes

Data entries ı used to hold the data and the structure of the entry is:

1- data (1x4) bytes

2- location(1x4) bytes

Lastly, I used data blocks which are holds the data entry array inside and the structure of the blocks are:

1- data [4096 / sizeof(data)] 512 bytes

Conclusion

In this project I made experiments for appending 1000 bytes per step for different file sizes. However, due to the version conflict that happened on github lost the last implementations I made on the project code. However; I had a chance to observe that most of the times writing files takes the most time rather than reading due to my implementation. Also as the size of the file increases the time that spend on the operations increases.However the append operation takes more time than reading and reading takes more time to file creation.