Project 3 Technical Document v2.0

CSE521 Introduction to Operating System

Wei Zhu

12/15/2010

Introduction

This document is written for the project 3 of CSE521 Introduction of Operating System.

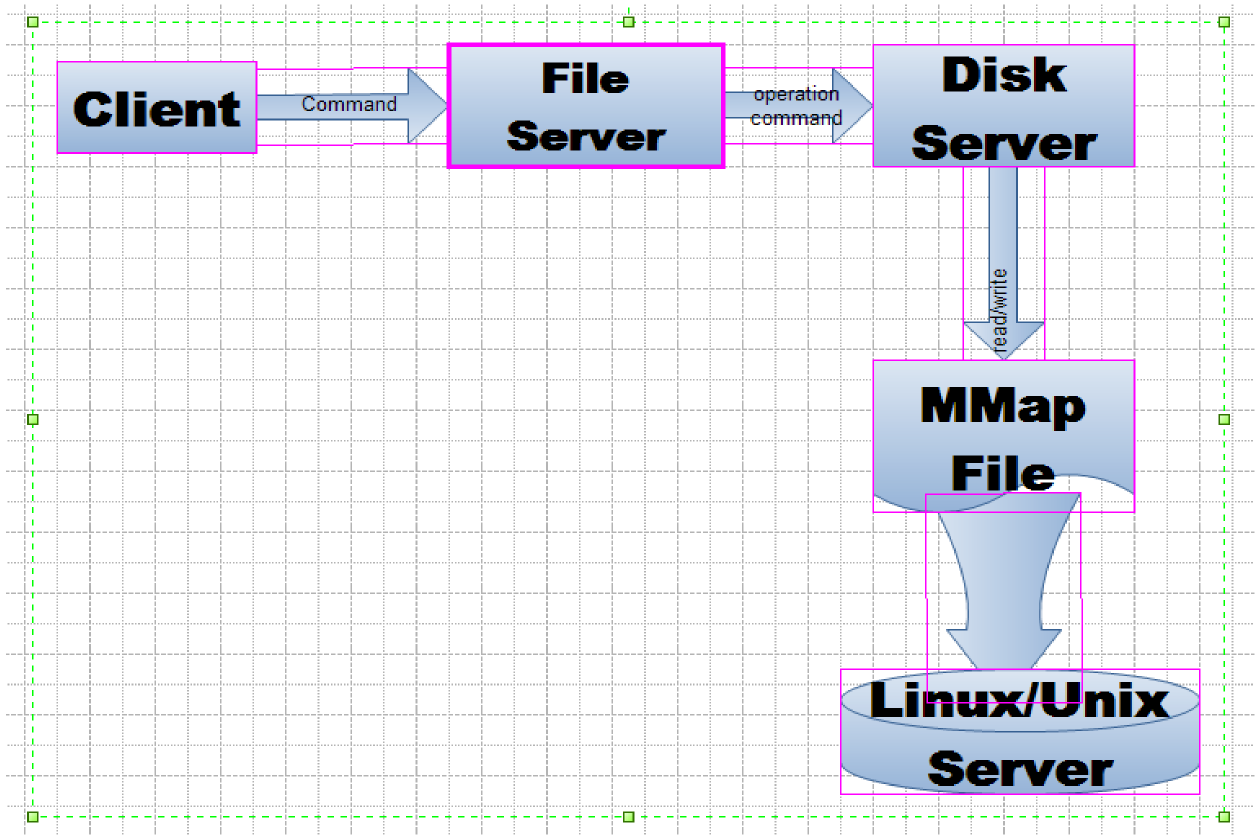
The Project's objective is to implements a storage system for us to get a good understanding of the concepts of socket programming, disk system and file system.

This project including the following parts:

1. Disk storage system
2. File system
3. User manual command
4. Disk/File Client

Overview of the System:

This is the overview structure of the storage system

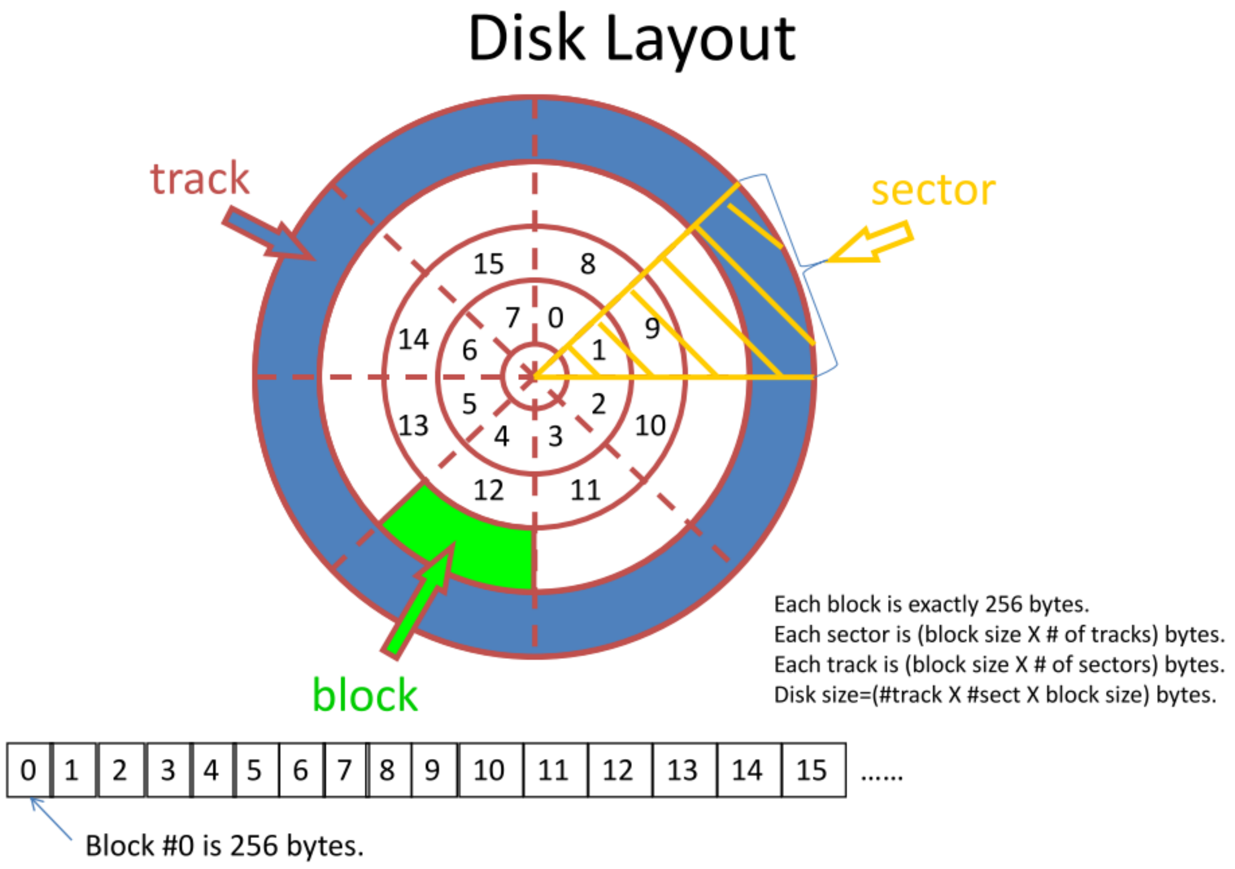


Sub-System Architecture

Disk Storage System

Basic Disk storage server is a Unix-domain socket server to simulate a physical disk. This system simulates the actual data storing to a real disk.

All the operation needs to specific the position in the disk system. In the real storage system, the read/write was depending on the tracker and sector, in the monitoring system, this was implemented in a block array.


Disk Layout
track
sector
block
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Each block is exactly 256 bytes.
Each sector is (block size X # of tracks) bytes.
Each track is (block size X # of sectors) bytes.
Disk size=(#track X #sect X block size) bytes.
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ......
Block #0 is 256 bytes.

(Reference from Scott's PPT)

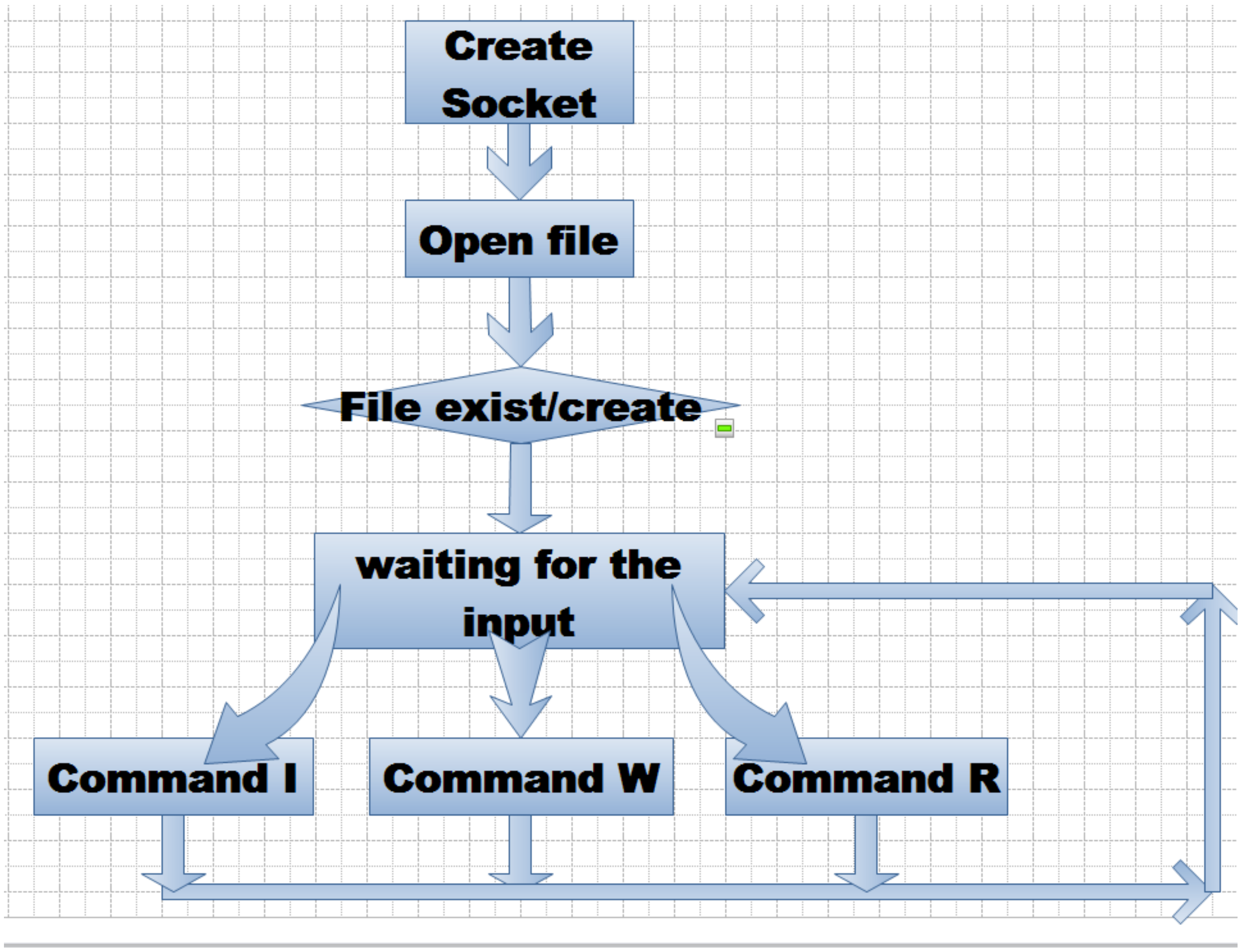
For this purpose, the disk storage system was designed to support following response:

I: Information request. Return the integers representing the disk geometry (the number of cylinders and the number of sectors per cylinder).

R c s: read the request for the content of cylinder c sector s.

W c s l data: write request for cylinder c sector s. l is the number of bytes being provided (maximum of 256).

The data format used in the disk storage server is very strict. And also two clients were provided. One is random command client, which produce the random command and send to server. Another is the client for accepting the command from the standard in.



File System Server

This is a flat file system that keeps track of files in a single directory providing the operations such as: initialize the file system, create a file, read the data from a file, write a file with given data, append data to file and remove a file, etc…

The server can understand the following commands and give the following responses.

Format Command: "F"

- Initialize any tables/data structures/values required to operate the file server (like the File Allocation Table (FAT) and the File Table).
- Save these tables/data structures/values to the disk server.
- Return no response to the file client

Create file command: "C <filename>"

- Create a file named <filename> on the File Server
- Update any data structures used to reflect new file
- Save these data structures to the disk server
- Return response: (a single char of '0', '1' or '2')
    - '0' for success
    - '1' for filename exists already
    - '2' for any other error (such as no space left)

Delete file command: "D <filename>"

- Deletes a file named <filename> on the File Server
- Update any data structures reflecting the deletion of the file
- Save these data structures to the disk server
- Return response: (a single char of '0', '1' or '2')
    - '0' for success on deletion
    - '1' for filename does not exist
    - '2' for any other error (such as if the disk server has been shut down…)

Directory listing command: "L <param>"

- Returns a list of all files on the File Server. The response to return should be a single packet (i.e. one write command) of the following:
  - If <param>==the character '0', return a string of filenames, each line separated by a <newline>, ending with a NULL
  - If <param>== the character '1',return a string of filenames and sizes, each line separated by a <newline>, ending with a NULL
- If there are 0 files on the File Server, return a single NULL character

Write file command: "W <filename> <lengthofdata> <data>"

- Writes (or overwrites) data to the file <filename>
  - <filename> must exist and have been previously created with "C" command
  - <lengthofdata> must be >= 1 and will be also <= 1000
  - <data> can be from 1 to 1000 characters of values 0 to 255.
- Update data structures and save data structures to disk server.
- Return response: (a single char of '0', '1' or '2')
  - 0 for success
  - 1 for filename does not exist
  - 2 for any other error

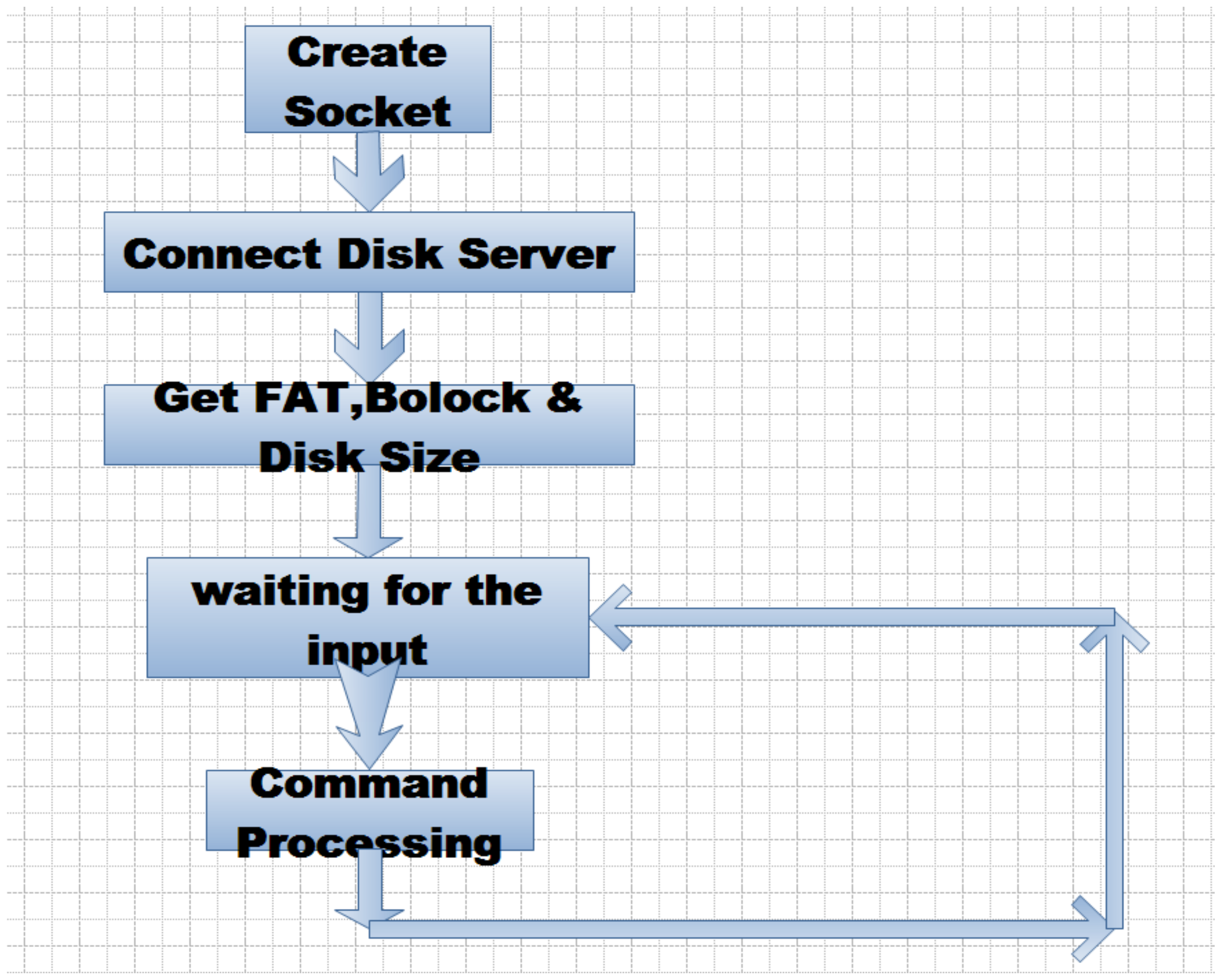
Read file command: "R <filename>"

- Reads data from the file <filename>,<filename> must exist and have been previously created with "C" command
- Return response is: "<return code><length><space><data>"
  - <return code> is '0' for success/'1' for filename does not exist/'2' for any other error
  - <length> is a text representation of a number value. For example, length of 10 would be "10".
  - <space> is just a space
  - <data> is a number of characters.

Append file command: “A <filename> <lengthofdata> <data>”

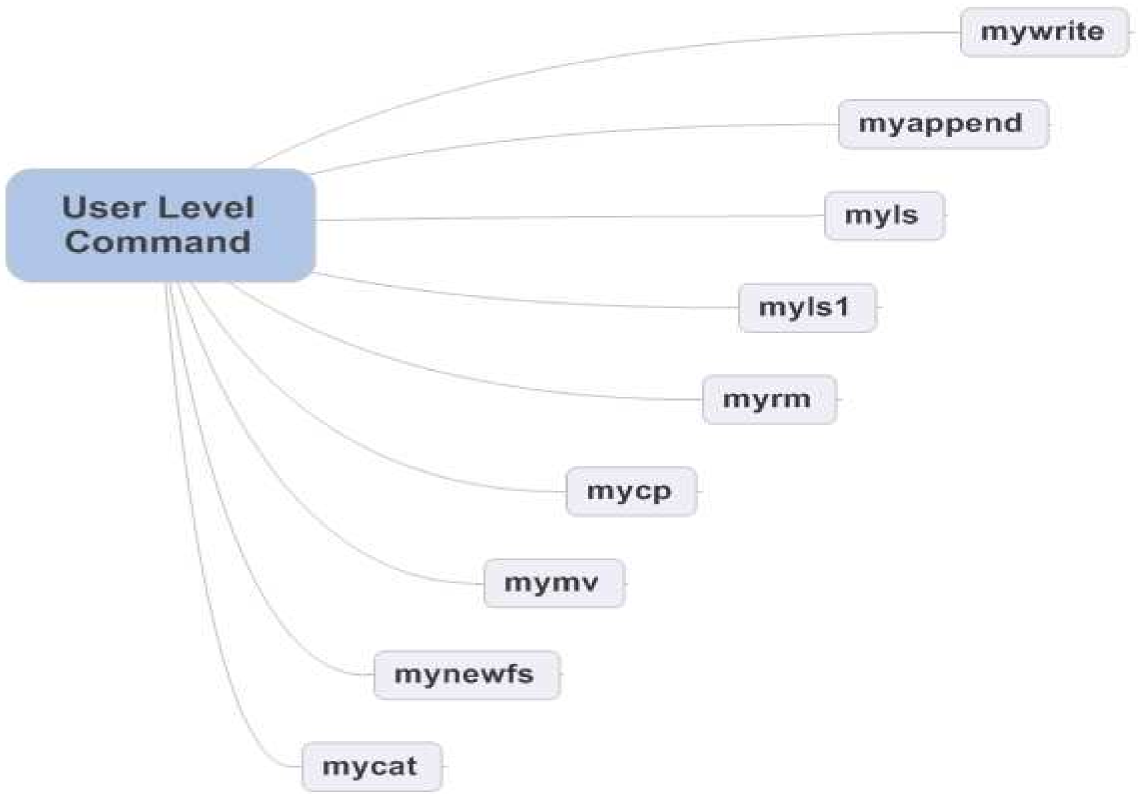
- Appends data to the end of the file <filename>, <filename> must exist and have been previously created with “C” command, <lengthofdata> must be >= 1 and will be also <= 1000, <data> can be from 1 to 1000 characters of values 0 to 255.
- Update data structures and save data structures to disk server.
- Return response: (a single char of ‘0’, ‘1’ or ‘2’)
    - 0 for success
    - 1 for filename does not exist
    - 2 for any other error

The following is the figure of the file system structure



Users level command

To make the file system useful, a few basic command-line user level commands was implemented. Each of them is very short program, they are each simply a client which talks to the file system Unix-domain socket.



Testing Result

The Disk-Storage Server

```
. /stdindclient MYDS

initial the parameter

created socket

loop entered

I

command line is I

send line is I

read count is 4

received response: 5 10

loop entered

W 1 1 3 abc

command line is W 1 1 3 abc

send line is W 1 1 3 abc

read count is 1

received response: 1

loop entered

R 1 1

command line is R 1 1

send line is R 1 1

read count is 257

received response: 1abc

loop entered
```

The File System Server

```
>C file1

command line is C file1

send line is C file1

read count is 1

received response: 0

>W file1 abc

command line is W file1 abc

send line is W file1 abc

read count is 21

received response: Unsupported command

>W file1 4 abc

command line is W file1 4 abc

send line is W file1 4 abc

read count is 1

received response: 0

>A file1 4 efg

command line is A file1 4 efg

send line is A file1 4 efg

read count is 1

received response: 0
```

```
>R file1

command line is R file1

send line is R file1

read count is 6

received response: 04 abc

>W file1 4 abcd

command line is W file1 4 abcd

send line is W file1 4 abcd

read count is 1

received response: 0

>R file1

command line is R file1

send line is R file1

read count is 10

received response: 04 abcdefg

>W file1 4 abcd

command line is W file1 4 abcd

send line is W file1 4 abcd

read count is 1

received response: 0

> W file1 8 abcdefgh

command line is W file1 8 abcdefgh

send line is W file1 8 abcdefgh

read count is 1

received response: 0
```

The User Level Command Testing Script

```
. /mynewfs

echo
012345678910111213141516171819202122232425262728293031323334353637383940414243444546
4748495051525354555657585960616263646566676869707172737475767778798081828384858687
888990919293949596979899012345678910111213141516171819202122232425262728293031323333
4353637383940414243444546474849505152535455565758596061626364656667686970717273747
5767778798081828384858687888990919293949596979899|./mywrite f1

echo
012345678910111213141516171819202122232425262728293031323334353637383940414243444546
4748495051525354555657585960616263646566676869707172737475767778798081828384858687
888990919293949596979899012345678910111213141516171819202122232425262728293031323333
4353637383940414243444546474849505152535455565758596061626364656667686970717273747
5767778798081828384858687888990919293949596979899|./myappend f1

. /myls

. /mycp f1 f2

. /mycp f2 f3

. /mycp f3 f4

. /myrm f2

. /myrm f3

. /mymv f4 f5

. /mymv f5 f6

echo TA|. /myappend f6

. /myls1

. /mycat f6

echo 123|. /mywrite f9

echo 345|. /myappend f9

echo 678|. /myappend f9

echo 999|. /myappend f9
```