Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Distributed File Systems

by

SUPREET SINGH VIRDI
(17312704)
15TH DEC, 2017

INTRODUCTION

The task defined is to build the distributed file system. There are many functionality in the distributed file systems. I have decided to implement the below mentioned functionalities.

- Basic Functionality (i.e. to open/read/write the file within Distributed file system).
- A Directory Services or name server, which stores the file path and allow the users to access the file.
- Locking Server is used to avoid the ambiguity in case multiple user modify same file at same time.
- Cashing is used on client side, to speed up the file system if client want to access the file he will access it again.

This is the base of the distributed file system, consist of the server, which provides the access to the file, and perform the specific functionality and there will be a client side, which provide the interface to the file system. The code is available on the Github repository (https://github.com/supreet29/Distributed-File-System).

IMPLEMENTATION

DFS is implemented in the python 3 and this require web.py file to run the REST services.

Start the name server and then run the file server then client will connect to the server.

Python namserver.py
Python Fileserver.py
Python Client.py

Then in the file server it will prompt in the file server.
`http://0.0.0.0:8080/`

After connecting to the client when client perform the task and then status of the API request can be seen.

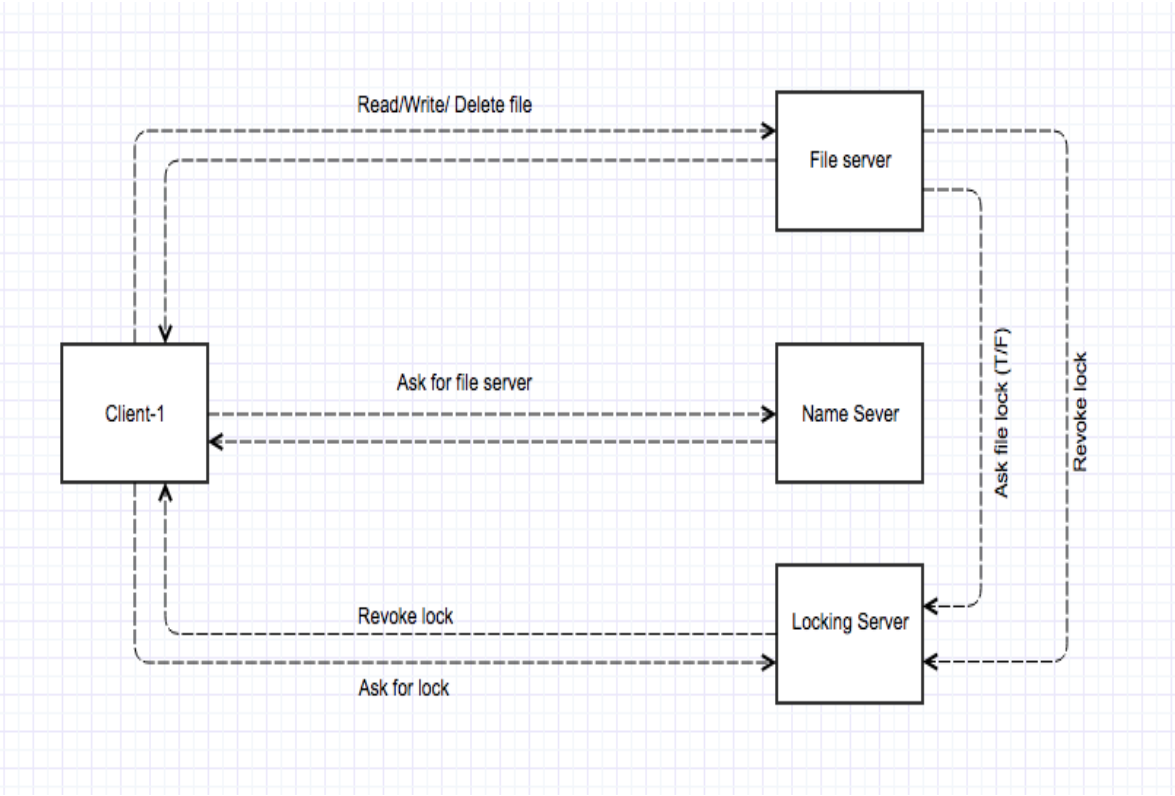
```
127.0.0.1:59491 - - [15/Dec/2017 09:07:59] "HTTP/1.1 GET
```

```
/filepath/ex.rtf" - 200 OK
inside get
127.0.0.1:59990   -   -   [15/Dec/2017   10:40:00]   "HTTP/1.1   GET
/filepath/ex" - 200 OK
```

OVERALL ARCHITECTURE

The server and client will be configured with the JSON file and all the details will be stored in dictionaries.

```
{
   "nameserver": "localhost:8000",
   "lockserver": "localhost:8001",
   "srv": "localhost:8002",
   "directories": ["/DFS_Test"],
   "fsroot": "/"
}
```

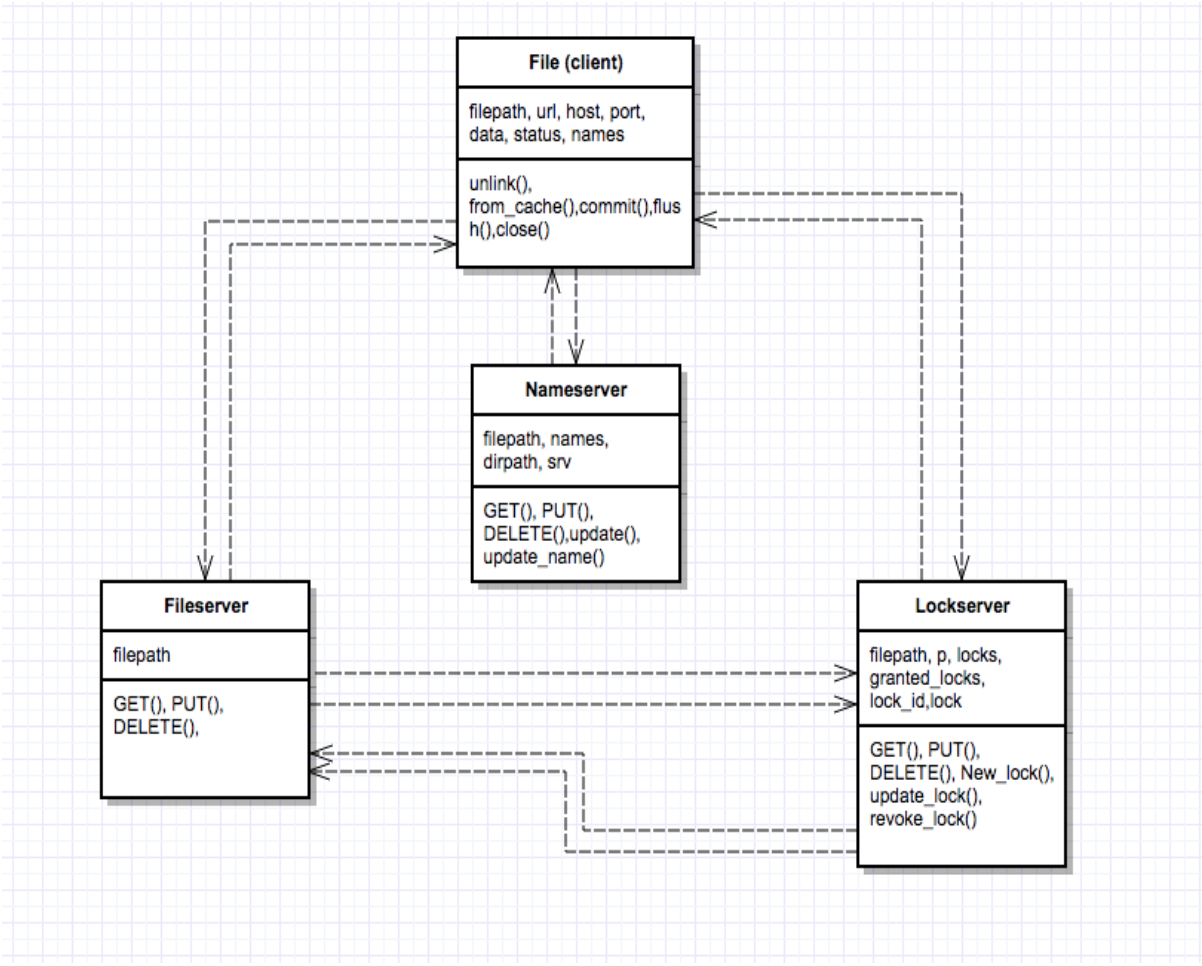


There are total 3 servers in the architecture i.e. fileserver, name server, lock server and cashing is done on the client side. And there is an implementation of the REST services to access the file and perform necessary tasks.

REST services are implemented in the file server to perform the basic functionality and make the connection between client and server. Three main function is implemented in the file server i.e. GET , PUT , DELETE. In the server open(filepath) is used to open the file and read, or edit the file.

Cache is implemented in the client side and client.File.from_cache can be used to retrieve the file from cache and that file is stored in the Temporary File pool. If the size of the Client.dfs.json is more than maximum size than it will be sored In the local disk of the system otherwise it will store in the RAM.

If file opened will check for the lock in the server and then it will check and then grant the access to write in the file.

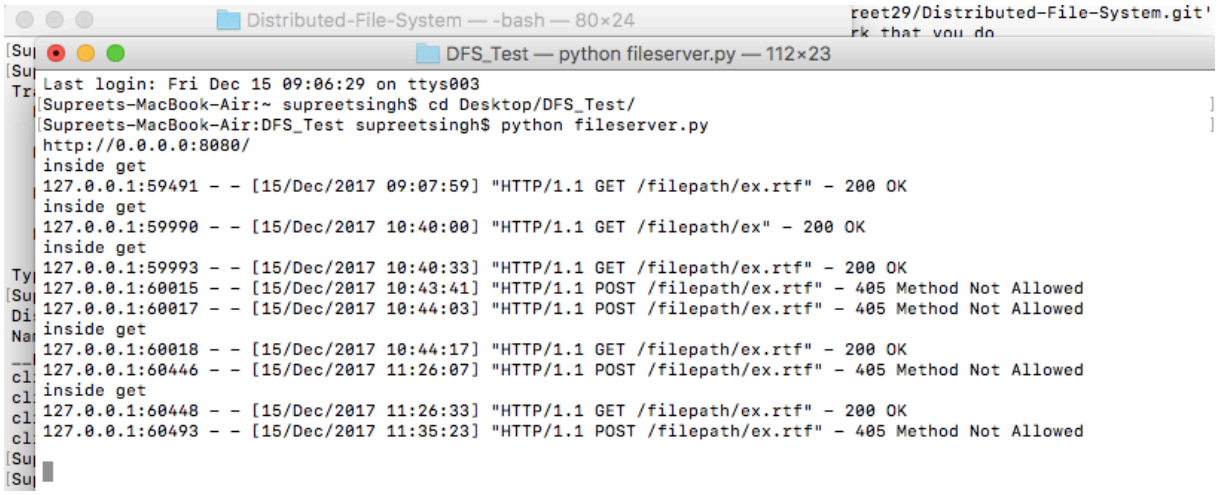


UML diagram of the distributed file system

FILE SERVER

File server is used to server the file with GET (read the file), PUT(Create or modify the file) and DELETE files. Each file is accompanied with the lock in the file, which is used in

the file to lock and be sure that user is authorized to read/make changes (edit)/ delete the file.



Client output

```
======================================================================
Welcome to Distributed FileSystem
======================================================================
1. Read File
2. Write File
3. Delete File

Select the option to:-
1
Enter the file name:
```