Project 2

1. Proxy and Server

1.1 Data

Proxy keep two FILE(customized class) array which contains fd, path, raf of a single file, one is used for public file another is used to keep track of private copy. Every time a file is open, proxy allocate a FILE to it. An hash table is also kept to map <Path, Version> of a file. Next, I create a class VersionList which contain a List Node, bool inUse, int version for the LRU. Last, A Linked list is also kept to store the LRU table.

Data is Server is much simpler, which only keeps a hash table for <Path, Version>.

1.2 Function

Server has few major function to communication with Proxy.

a) getFileFromServerToProxy
b) uploadFileFromProxyToServer
c) rmFileFromServer
d) getVersion

And Proxy handle all the “system call” function and also whole file caching.

2. Working flow

When Client open a file. The first thing Proxy do is to judge if he has permission to that directory and then, I call a function to simple the path ensuing there are not two different path point to the same file.

Then, I process a fd to it, compareVersion or initializing the version to determine whether it’s needed to get latest version from server. After all these, according to the open option, Proxy decide whether to give it a private copy like (CREATE, CREATE_NEW, WRITE) or just the shared one(READ)

To ensure the consistence of concurrency request. I chose to simply assign another name to the private copy instead of using Locks. For example, change the name to path +”by”+fd. After all the writing finished, just replace the private copy with the shared one and push to server.

3.LRU

For LRU part, I designed a circle shaped Linked List which means the tail->next is head. Node contain <previous, next, data>, pre and next makes it easy to delete an element. All the recent used file are put at tail and least used file are put at head. When Miss, just append a new node. When hit, just delete the original one and append a new one at tail.

In the situation of not enough space. I use a while loop to keep dealloc space from the head of the list. Until it can fit.