P2 Design Report
Siying Jin
siyingj@andrew.cmu.edu
3/2/18

Protocol between proxy and server:
Using RMI, the **Proxy** can communicate with the **server** using the interface **RMIserver**.
It has the following methods:

```
/**
Generate a private path for each new client ==> ensure concurrent client to read/write at
separate paths.
*/
int getClientID() throws RemoteException;
void setClientID(int clientID) throws RemoteException;

/**
Update version number on the server after each modification to ensure cache freshness.
*/
int getVer(String path) throws RemoteException;
void setVer(String path, int ver) throws RemoteException;

/**
Enable file transfer between proxy and server
*/
byte[] fetch(String path) throws RemoteException;
void write2server(String path, byte[] bytes) throws RemoteException;
```

Consistency Model:
Check on check.
Idea: Push modifications to the server when closing and increment server's version number. On read, check if proxy's version number matches server's version number. If it does not, fetch the whole file from server.
Implementation: on close(if modified) server.setVer(curVer+1);
on read (check:) if (version != server.getVer()) fetchFile()

Cache Freshness:
I used the data structure CacheEntry to record the current cachepath, proxy version number, whether it's modified or being opened.

LRU:
Idea: Always add files to LRU cache unless there isn' enough space, in which case evict the LRU cache entry which is not being opened.
Implementation: Doubly linked list to model the cacheEntries. Map<String, CacheEntry> to map pathname to cache entry in O(1).
It has the following public methods to communicate with Proxy:

```
/**
Add path to cache with c. Evict older cache entries if necessary
*/
set(String path, CacheEntry c)
/**
Move the cache entry associated with the give path to MRU(head of DLL).Return the cache entry
*/
CacheEntry get(String path)
```

Work Flow:
On read:
The proxy will 1.simplify the path; 2 check if the file is in cache: on hit, get the latest version as with the server if necessary; on miss, download the file from server and put it in cache; 3. make a separate copy of the file(except with READ option) for current client to ensure concurrent read/ write don't interfere; 4. mark cache entry as being opened so that other clients won't delete or evict it.
On close:
The proxy will 1.copy private file's content to cache; 2.push modifications to server and update server's version number.
On unlink:
The proxy will 1.delete file from cache(if it's in cache) 2. delete file from server