# CMPUT-379-Assignment-2

<ul>

<li>zhizhou</li>

<li>ry</li>

</ul>

Design:

Our goal is to implement a client/server application. The server here works as a repository (with no directories) to store the files upload from the clients. Multiple clients can list, upload and download the files from the server at the same time.

Before we built the server and client, we need to create a new folder as the repository.

We create server first, using IP and set port. The IP we used for testing is 127.0.0.1. Then we can connect to it through setting the client using then same port number and the folder name of the repository folder we built.

At client, we can enter:

“l” to list all the files we have uploaded.

“u Filename” to upload the file to the server.

“d Filename” to download the files from the server.

“r Filename” to delete the files in the server.

If we want to check the XML file which recorded the hash and other elements of the file, you can enter “cat FolderNameOfRepository /.dedup” to check.

For some special situations:

If the client uploads the file with name which has already existed in server, the upload will not be implemented but return the message:” file already existed”.

If the client uploads the file which has already existed in server (i.e. they have the same content, and the same hash name) but has distinct filename, the server will maintain only one file but remember that the same contents are available under two filenames.

If the client uploads more than one file with distinct filenames but same content (same hash name), the file with this hash name will be deleted only when the client deletes all the filenames of this file.

Questions:

1. What is the internal data structure you use to map file names to file contents?

We use a XML file named “.dedup”. In this file, we have three elements: “hashname”,” saveas” and” knownas” for each file we uploaded. Files with same content will have the same” hashname”, and are recorded under the same file term. “saveas” will contain the name that this file is actually named in the repository. “knownas” will contain the list of filenames belong to this file when it’s uploaded.

1. How do you ensure that the server is responding efficiently and quickly to client requests?

We use a while loop to receive the message. When we get the message, we will handle the event immediately, thus we can ensure the process it’s efficiency and quick.

1. How do you ensure changes to your internal data structures are consistent when multiple threads try to perform such changes?

We set a global variable named “accessing”, initialized as “0”. When we access the XML file, a function named “lock” is implemented. The function “unlock” will be used to set it to 1. The process will keep waiting until the “accessing” is equal to 0. Then the accessing is equal to 1 when an event is handled now. The function “unlock” will be used to set it to 1. For each thread, it will implement “lock” before it accessing the XML content and invoke “unlock” after it done the accessing.

1. How are the contents of a file which is being uploaded stored before you can determine whether it matches the hash of already stored file(s)?

The file is already stored in repository when it’s being uploaded. It will be deleted or kept after it matches the hash of the already stored file(s).

1. How do you deal with clients that fail or appear to be buggy without harming the consistency of your server?

Once the clients fail or appear to be buggy, the tread will end, back to process that waiting for the tread.

1. How do you handle the graceful termination of the server when it receives SIGTERM?

When it receives SIGTERM, server will store the current pointer, XML file named .dedup, which contains the mapping between stored files in the repository before it quits.

1. What sanity checks do you perform when the server starts on a given directory?

It will do the check that if the .depup exists. It will create this file if .depup doesn’t exist.

1. How do you deal with zero sized files?

We treat zero sized files as normal files, which can be uploaded, downloaded and deleted.

For the working process, we worked together and discuss different ideas and methods we thought. One of us put the method into implementation and the other test the code and found the errors.

We run and test the code on official VM.