**CSE 5306 Distributed Systems Spring 2022 Project-1**

Name: Adithya Venkatesh

ID: 1002032198

Email: [axv2198@mavs.uta.edu](mailto:axv2198@mavs.uta.edu)

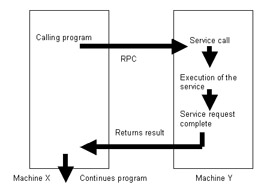
**PART-1 (10 pts)**

In this assignment we were asked to setup the Linux operating system in our host operating system (MAC M1). I have installed a virtual machine called UTM and in UTM I have setup Ubuntu. Since my host operating system is M1 MAC it was difficult to install a virtual machine and after few research I found a VM called UTM and with UTM I was able to setup a VM and install Ubuntu

**PART-2 (20 pts)**

In this part we have to implement a multi-threaded file server that supports UPLOAD, DOWNLOAD, DELETE, and RENAME file operations. Use different folders to hold files downloaded to the client or uploaded to the server.

I have  used XMLRPC in python to establish a RPC client server model.



Explanation:

1. Client connects to port **8000** and server accepts the same.

2. At Client, we provide 4 options to user: UPLOAD, DOWNLOAD, DELETE, RENAME

3. For **UPLOAD** command, client needs to enter the filename that it is going to upload it to server. If file is present then it will upload or it will ask us whether we want to create the file and upload. If yes it will create a file and upload it.

4.For **DOWNLOAD** command, client needs to enter the filename that it wants to download from server. The server will check whether a file is available at the server end and if it is available it sends back the file to client.

5.For **DELETE** the client needs to enter the filename that it wants to delete in server. When client enters the filename, it is sent to server to check whether it is available at the server folder and if it available it deletes the file from server as well as from client.

6.For **Rename** the client as for the file name to be renamed and the name which it has to be changed if file is available in server and client it renames in both the folder and if it is not available it send a message that it is not available

**PART-3( 30 pts)**

In this part we implemented a multi-threaded file server which supports basic operations as: UPLOAD, MODIFY, DELETE.

I have used a file watcher method called Watchdog which keeps on monitoring the client folder and if any of the operation happens same is reflected back in server.

UPLOAD: if a file is created in the client folder the watchdog triggers created event then then same file is placed in server without any delay.

MODIFY: if a file is modified in the client folder the Watchdog triggers an event modified and with that event current file is transferred to server folder.

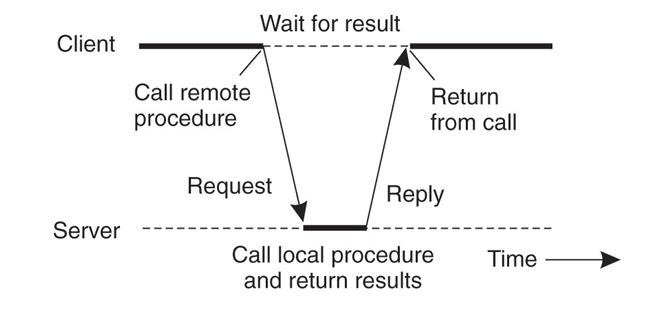
Delete: when a file is deleted in the client folder. The watchdog triggers a deleted function and with that function server is called and file which is deleted is passed to server and that particular file is deleted from server.

**PART-4(40 points)**

In this part Implemented a computation server to support add(i, j), sort(array A) operations using synchronous, asynchronous and deferred synchronous RPCs

Synchronous:

The RPC protocol allows the construction of client-server applications, using a demand/response protocol with management of transactions. The client is blocked until a response is returned from the server, or a user-defined optional timeout occurs.



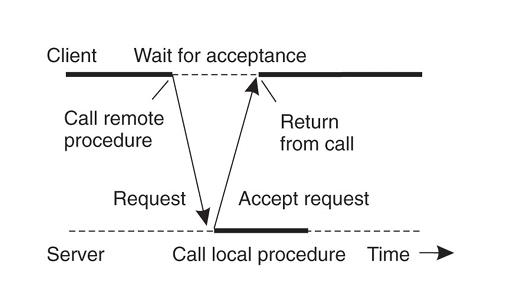
**Execution Process:**

* The client prompts with three options and we select the synchronous option.
* The client gets input for addition of 2 numbers and array length for sorting and the data to be sorted
* In synchronous, Addition is called first and the sorting function is called after addition
* I have given a pause of 5 second in addition function in server side and the client waits till the first function returns the result and then the next function sorting is called
* Hence the above is synchronous operation where 2 functions are called one after other and the second one is executed only when the first function returns a result

Asynchronous:

For asynchronous RPC, the server immediately acknowledges an RPC call before it actually performs a computation. The result of the computation is saved in a table on the server, which can be looked up by the client for the RPC result. The design of the client will be slightly different from that in the synchronous

RPC. Instead of waiting for a synchronous RPC to return, the client using an asynchronous RPC switch to other computations and queries the server for the RPC result at a later time.

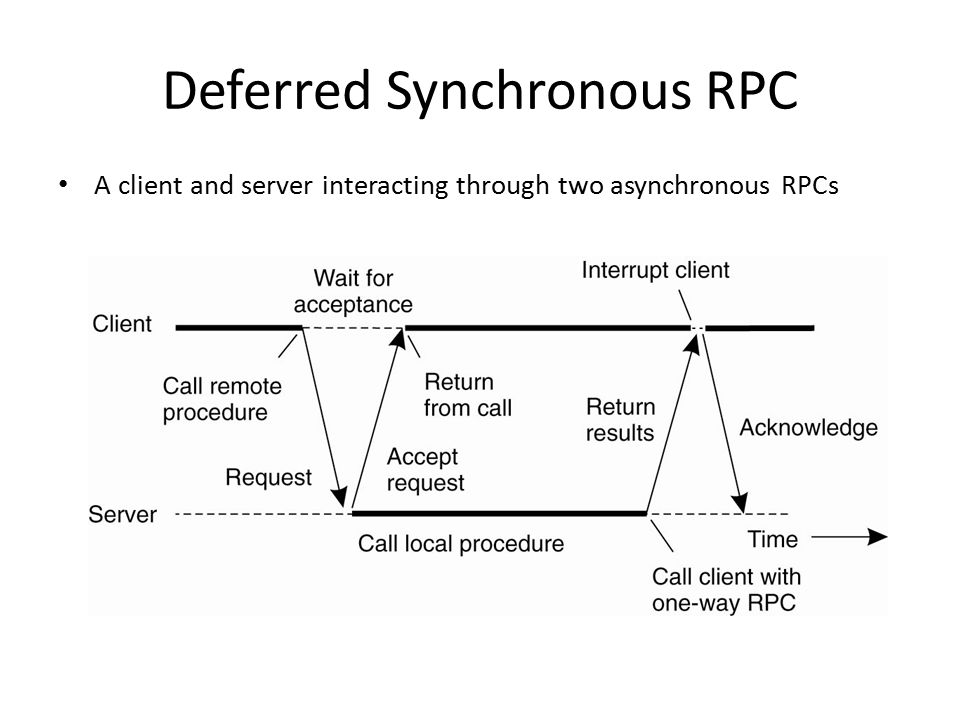


**Execution Process:**

* I have used asyncio in Python for async operations.
* asyncio is a library to write **concurrent** code using the **async/await** syntax.
* asyncio is used as a foundation for multiple Python asynchronous frameworks that provide high performance network and web-servers, database connection libraries, distributed task queues, etc.
* In Asynchronous, Addition is called first and the sorting function is called after addition.
* 5 second pause is given in addition function and then sorting is called.
* The Client does not wait for the addition function to return the value and it continues with the sorting function and prints the result returned from sorting first
* Then the addition result is printed
* Hence the above is asynchronous as the client does not wait for the first function result in turn it prints the result of the second function and prints the first result when it is available

**Deferred Synchronous:**

In deferred synchronous RPC, we devise a mechanism to interrupt the client to return the RPC result to the client when the server has completed its local computation.



**Execution Process:**

* RPC client does prompt with three options and we select the deferred synchronous option
* In Deferred synchronous, Addition is called first and the sorting function is called after addition.
* Meanwhile the client will be performing some task. I have given a for loop which prints text
* And when client is printing the client is interrupted by server with server output of add and sort
* Whatever is returned first is printed in the console and then client continues with execution.
* When the next server return the client is paused and server result is printed and client continues with rest of the job
* Thus Deferred synchronous is showcased as the client is interrupted by server to return the computed result, and client continues with its process.