Summary

Group name: MeetOurMakers

Group members: Fengyuan Zhang, Xiao Ling, Chia-Sen Wang, Bo Ding, Sisi Wang.

Assignment Overview

In this assignment, we implement key-value store with both TCP and UDP communication protocols. Key-value store is a data storage paradigm used for storing, retrieving, and managing associative arrays, and they can only store pairs of keys and values, as well as retrieve values when a key is known. TCP and UDP are both transportation protocol. For TCP, it’s connection-oriented protocol. If connection lost, the server will request the lost part. But for UDP, it’s connectionless protocol. Thus based on the character of two protocol, we will write the server and client differently.

Both TCP and UDP server will be single-threaded. And the socket will be used for tcp client server communication while datagramsocket will be used in udp communication. A hash map was used on server to store key value pair based on the assignment requirement.

Technical impression

In the coding part:

we have our tool package, we write a switch function for our 3 required function ‘PUT’, ‘GET’, and ‘DELETE’ executed on a hashmap. The question is how to deal with the command from the client. On one hand I use String.split() to separate the whole command into different part using a separator “//s+”, which is different from the space “ ”. The separator “//s+” can accept some other forms of space such as tab. On the other hand, it is also important to pay much attention to the bound, which means to deal with “put a b” and “put a b c” in different ways. In this point, I only deal with the orders having the right number of parts. It works, but also lead to some repeating codes that we need to make some changes in the future.

Another method in SupTool class is showMewithTime(). This is used for both TCP client and server as well as UDP client and server to show the message arrived time. In fact, as in this project, we just connect to the localhost, so we use the system time (Matt you said it’s ok).

TCPServer:

In this part, I think there is nothing special. As our textbook has offered a sample, the only thing I need to do is to understand the whole process of establishing TCP connection: Firstly, the serversocket listens on a port, and ServerSocket.accept() will block the current thread. Once it receives a request and finally set up the connection with client, I will get the input and output stream used to communicate between the client and server side.

TCPClient:

We need serverIP and server port number to establish a connection. In this project, we use localhost and a given port (wish you not have a conflict). The steps of establishing a tcp connection is written above. Here I want to talk about the encode and decode. I use the DataInputStream.readUTF() for decoding work while DataOutStream.writeUTF() for encoding, just as the textbook shows us.

For UDP:

The difference of UDP communication from TCP is that the datagram sent by UDP is transmitted without setting up a connection. If a failure occurs, the message may not arrive. So based on the UDP server and client API in our textbook, we set up our UDP communication, but we set a threshold of 3 in UDPClient, if tried more than 3 times,we will stop the sending of message. Another point is, for UDP, all arrived messages are placed in a queue for the socket that bound to the destination port, but you don’t know the order of those messages. That is where we need to make some improvement.

For makefile:

We write a makefile under out main directory. To be honest, it is a hard work since we all not familiar with the shell programming. At first we implemented it with a dirty way, writing all the compiling command one by one in the same directory. We all were not satisfied, then I searched for some examples about using makefile to manage the entire project and finally made our own “make”! Now it has 4 different functions: setting up the structure of project, building, cleaning the prior building result and running the specific app as you wish. I think it is really a breakthrough in this project.

10/24/15

Fengyuan Zhang