Robert Tedesco

Professor Kermani

Computer Science 453

24 March 2017

My TCP client is mean to be started first , on port 57772. It waits on a client in infinitely (while(true)), and then tries to read in the line. I then parse the message, delimiting by commas, and store the op code and operands in variables. If I encounter an s as an oc, the server quits. Originally I thought about sending a message back to the client that it quit successfully, but I thought it better if they were more agnostic of that behavior. The oc is sent through a switch statement so that if it doesn't hit a case defined, the error message is received. The message is then build off of the response result (calculated in the switch statement), and sent to the client. The client takes 3 operations from standard input builds a message and sends to the server. If the client gets an s oc it stops taking input builds a message with the s, and sends to server. Then it closes the connection and quits regardless of what the server does next. I handle exceptions in both, and continue execution. The server sends and error message back when this occurs. I throw IO Exceptions however. My reliable UDP works in a similar way, except implemented with datagrams, and doesn't have the intial 'accept' handshake. For Unreliable UDP I used Math.random to generate a random boolean, and drop requests in the server half the time. In the client I built an additional try-catch to catch when the server drops (the client will experience a timeout and catch this as an exception). The timeout increases If this happens and the program will loop, continuously trying to send the message. After timeout reaches 2 seconds the client gives up and quits. I handled the user asking to quit here by having the client wait on a server response even when quitting (in the case that the server drops the message that says to quit).

I tested this program by giving different inputs of characters at each of the 3 parts, including alphanumeric where there were supposed to integers. I also tried dividing by zero. To test this program use:

Javac TCPServer.java or UDPServer.java or UUDPServer.java

Javac TCPClient.java or UDPClient.java or UUDPClient.java

Then (where XXX is TPC UDP or UUDP)

java XXXServer &

java XXXClient