Design Process

In order to start a chat between the client and server, I had to connect the two. When they are connected, the listen socket is changed. Which is added to the client's and server's socket list. After that client and server are connected based on source port and destination port.

I started in Testsim.py where I created six different def functions for setting up the sever and client, connecting the client to the server, broadcasting a message, unicasting a message and printing out the list of users. For each of those function, it would their corresponding function in CommandHandlerP.nc. Depending on the case of setting up the server to printing out the list of users, a signal would be made to an event in Node.nc, where each function would call an event in ChatP.nc a long with their arguments.

In ChatP.nc is where the bulk of the project is at. For setting up the server, I would set the source port to 41, source address to 1 and state to listen. For setting up the client, I would set the source port to the port that was passed from Testsim.py, set address port to its certain location, state to listen, and store the username in the socket. In the connecting to the server to output "send hello" function, I would send the socket to Transport connect function and after connection the username would be in an array for the printing all users. Then I would make a broadcast function and unicast function, which is being send through Forwarder's send function.

Discussion Questions

1. In a single transport connection per client, there can only be one command per line, which would make it slower and messages won't get lost. When using a different transport connection per client, the connection would be faster and less reliable as packets can be lost or out of order.
2. TCP is a good when to connect with the client to server. It helps make a stable connection between the client and server. The problem with it is that it is a one-way connection. For example, from a chat-to-chat application, it would have to wait in the queue again and the messages won't be received quickly. Also, TCP can only accept strings, so if there are any other data types it wouldn't be able to send.
3. The transport protocol needs more functions before it can become a web server application, but it can send and receive messages. Some features will be implemented and will send packets to different nodes which isn't good.
4. I would add detail comments for each function. By saying what it does and where the information is coming from which function and/or file.