CMPE 150 Lab #3 Design Doc

Simon Ho (sho16@ucsc.edu), Alexander Swift (afswift@ucsc.edu)

February 1, 2015

1 Connecting/Disconnecting

Before connection, a check is performed to see if another user is already using the desired listen port; if one exists, an error is thrown saying that the Address is already in use.
Upon connection, every client loads up the peers.txt into an ArrayList. Welcome text is displayed so the user knows that the program has started. Each tuple contains an IP address, a name, and port, and an optional chat color. The client then tries to make connection to everyone on the list. If connection is successful, other clients will get a message stating that the new client has connected to the chat. All the other clients now know which port the new client is listening on. Each client then updates their ArrayList to add the new information of the new client.
To disconnect, the client types 'exit' into chat. 'exit' does not display in chat, but instead, the message that the client has disconnected is displayed to the chat. Then the socket is closed and the system exits. Upon exit, the client will check its ArrayList with the peers.txt file. It will look for new indexed entries in its peer list that is not present on the peers.txt file, and add the entry to the peers.txt file. Entries that already exist will not be changed by the client. Thus, the peer list text file will always be updated on proper exit. If the chat is hard-crashed (i.e. Ctrl + C), the client does not perform this step and will not update the client list.

2 Message sending

A scanner looks for sender input, then, for every peer in the peerlist, a new socket is opened, and Printwriter sends the message through the output socket, sending first the name of the sender, then the message itself. Then the socket is closed. The receiver is always listening on their listen port, and always accepts a new message. A new thread is spawned to handle the new information, and a BufferedReader takes in the InputStream and converts it to a string via readLine. Then the message is displayed to the recipient on the screen.