A Group Chat application using java sockets presented in a GUI

B00071601  
B00067349  
B00062302

Assignment 1 | Network Distributed Systems | March 18, 2016

# Introduction

In this assignment we were tasked with creating a group chat application using Java techniques and sockets to send the data. We attempted to first create a pleasant GUI for the client and implement the data structures afterward but we found that this approach would require us to rewrite much of the code we already had so we decided to start again and focus on the sending of data rather than the graphical interface.

We were happy with our finished product but felt that time constraints hindered the inclusion of several extra features we had in our original program such as the ability to change usernames, save the chat log &c. &c.

# The creation process

We first started by deciding on our group and separating the main tasks evenly, we found however, that this was not a realistic way to complete our tasks as our code was dependent on the code of others so we then decided to attempt and do as much of our work as possible in a unified group.

This lead to us doing more work quicker but was limited to certain time periods when all members of the group could agree to get together and work on the project. We still did some work in our own time but this was code that was not necessarily dependent on the code of others.

# The program

The program uses some of the code we used in previous lab exercises modified to work in a GUI. We combined our code with some very useful resources we found online which aided greatly in our creation process.

We have created GUI’s for the client and the server, on the server side, the server must start the program and open the port (1500 in our case) to ready it for data transmission, the server then continues to monitor the application, displaying the server events and the messages being sent between clients. The server also features some nice little customization options; the font size can be increased or decreased and the colour of the GUI can be changed for ease of use. These same options are present on the client side.

When the client is opened, he can join the session but only after the server has started the session. The client must login with a username and he is stuck with that username – against our better wishes. This is by design as once the user logs on, the session is created – if we wanted to change the username, a new session must be created, disrupting the original session.

Messages are sent as a new object, according to our research, this is intrinsically easier to do in java. We create the message object and sent it across to the server where it is broadcast to the other clients.

The client side program also allows for the client to log off from the session, broadcasting a message to the other users – at any time, the client may check who is logged on to the group chat using the “Who is in” button at the bottom of the GUI.

Some of these design decisions were not ours but we found them during our research and felt we should implement them regardless. It made the writing of the code simpler for all members of the group.

# Conclusion

Overall, we were happy with our end product. We felt that all members of the group benefited the group and did their tasks.

We hope that the code is up to scratch and is of high quality. We also hope that this report is sufficient and explains our vision for this assignment well.