COMP 4981

Assignment 3 – Chat Program

Dhivya Manohar , Dylan Blake
A00927659 , A00

March 24th 2016

Table of Contents

Summary

The purpose of this assignment was to learn about select( ) and to create a chat message emulation. Students had to create a client side that connects to a server that echoes back all messages received. Once connected, client can send messages and wait for other clients to send messages and have a conversation. Clients can see who is connected at all times. Meanwhile, the server receives all messages and only echoes back to all clients where the message did not come from.

Usage

Workspace

For the execution of this program, you need a Linux Operating System. This program will be running on the terminal (server) as well as a GUI (executable) for the client side.

Set-up

Make sure you have the Makefile and all server related files in the same folder. You also need one terminal running for the server. Also make sure you have the client executable. Doesn't have to be in the same folder as the server files.

Connect

In order to run the program, you have to change your directory to where your makefile and files to be read are. After changing directory, just type in the words 'make' on the terminal. This **only** makes the server side of the program. *Make sure to only make one server.*

To run your server, type './server'.

To run your client, just double click on the client executable. You can create as many clients as you want. Clients can connect to server by going on the menu tab and clicking connect. A new window will pop up prompting for your user name, IP address of server and port you want to connect to. Make sure to type the right IP address of the server in the format ' 000.000.000.000 '

Interpret Data

On the client side, once you press 'Send' after typing in your desired message, your message pop up on the GUI box that holds on incoming and outgoing messages. To the right of that, you will a box that shows you all the clients connected to the server.

On the server sider, you will get messages indicating what the server has done/is doing.

Disconnect

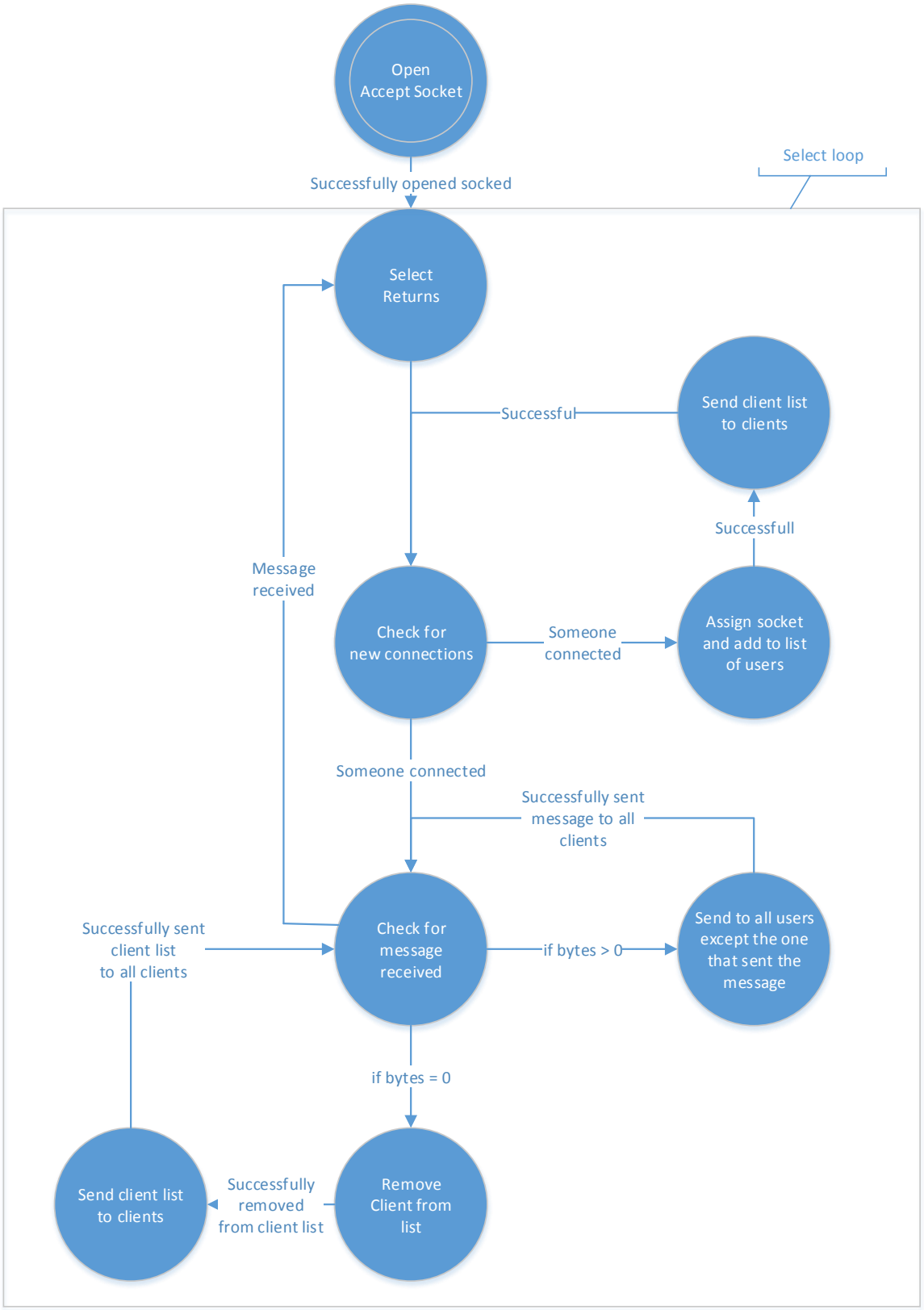
In order to disconnect from the server, just hit CONTROL + C and from the client, just click disconnect from the menu bar or close the window.

On the server side, this will close the server and kill connections to all clients connected.

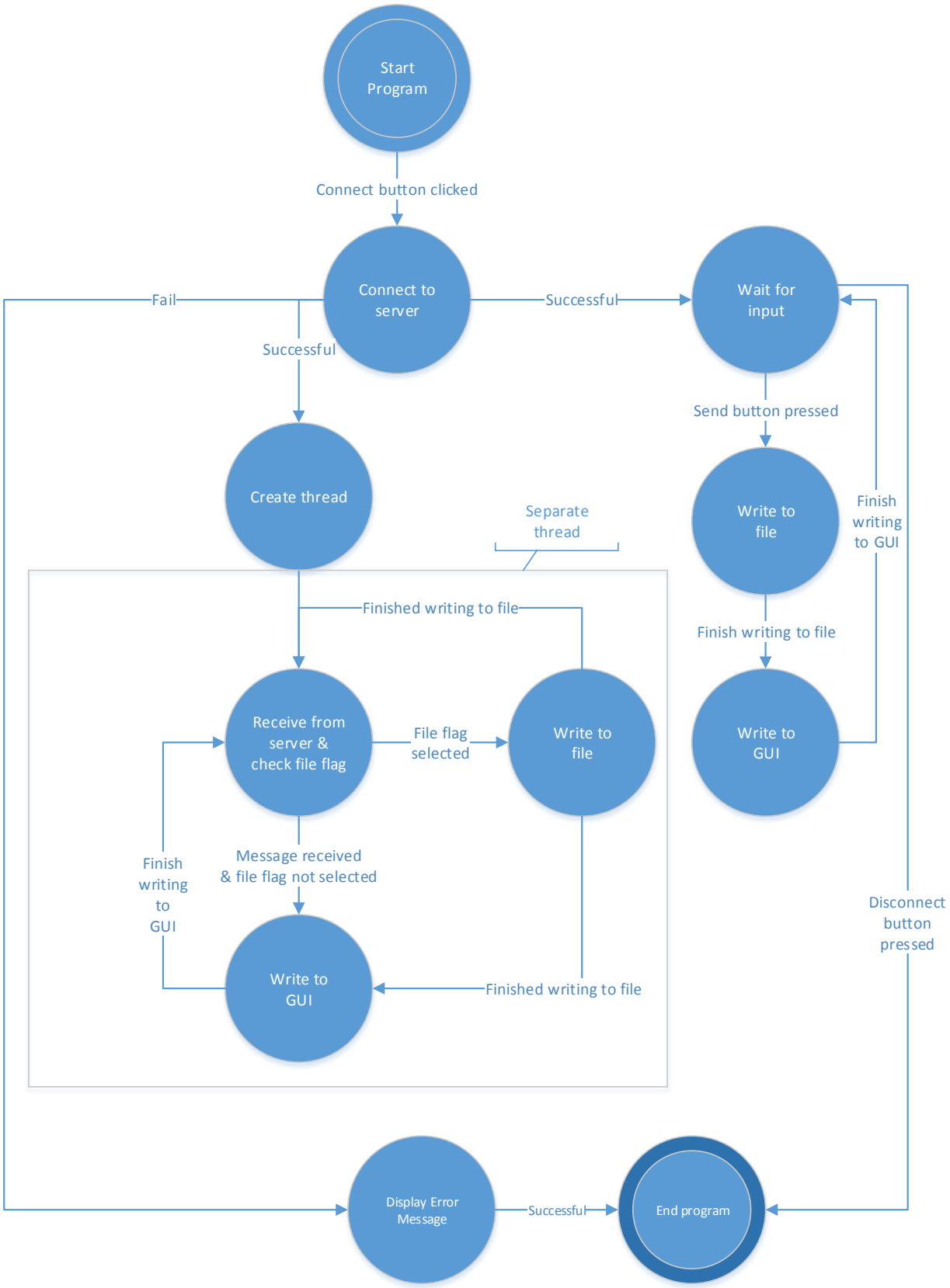
On the client side, this will close only the client, but the server is still running along with any other clients that are still open.

State Diagrams

Server



Client


Start Program
Connect button clicked
Connect to server
Fail
Successful
Successful
Wait for input
Send button pressed
Create thread
Write to file
Finish writing to GUI
Separate thread
Finished writing to file
Finish writing to file
Receive from server & check file flag
File flag selected
Write to file
Write to GUI
Finish writing to GUI
Message received & file flag not selected
Write to GUI
Finished writing to file
Disconnect button pressed
Display Error Message
Successful
End program

Pseudo Code

Server Side

Open Accept Socket:

```
Use socket() call to open a new socket
Set socket so that it can be reused and other sockets can
bind to the same port
Bind this socket to server
```

Select Returns:
Check for new connections:
Assign socket and add to list of users:
Send client list to clients:
Check for message received:
Send to all clients except the one message received from:
Remove client from client list:
Send client list to clients:

Client Side

MainThread:

Connect to server:

```
Use socket() call to open a new socket
If this call fails
     GOTO Display error message
Set socket so that it can be reused and other sockets
can bind to the same port
Use connect() to connect to server (use user input)
```

Create thread:

```
Create pthread to run the receive function
```

Wait for user input:

```
Waits for 'Send' button to be clicked
Once clicked, it checks to see if file flag was set
If flag was set
     GOTO Write to file
     GOTO Write to GUI
     Use send() call to send message to server
Otherwise
     GOTO Write to GUI
     Use send() call to send message to server
```

Write to file:

```
Create a file using the username of client
Open the file
Write message from client to file
```

Write to GUI:

```
Append to the correct GUI element
```

Display error message:

```
A pop up window saying you can't connect to server is
created
```

ReadThread:

Receive from server & check file flag:

```
Use recv() call in a forever loop. It blocks until on
the recv() call received something.
When it receives a message
     Check if it is a client list update
          If it is a client list update
               Append list to appropriate GUI element
     Otherwise Check if file flag was set
     If file flag was set
          GOTO Write to File
          GOTO Write to GUI
     Otherwise
          GOTO Write to GUI
```

Write to file:

```
Open the file to write to
Write message from client to file
```

Write to GUI:

```
Append to the correct GUI element
```

Tests

Summary Table

Test No.	Description	Procedure	Result
1	Server gets created	Run server and message will print on terminal if server was created	Figure 1 – Passed
2	Client gets created	Double click on executable and a GUI pops up	Figure 2 – Passed
3	Client able to connect to server	Click on 'Connect' menu item and type the appropriate information and click the button to connect. Displays on the server side, which client is connected. Also displays on client side	Figure 3 – Passed
4	Server able to display all clients connected	Connect multiple clients. Server will show a list of clients every time a new client connects	Figure 4 – Passed
5	Clients able to send message to server	Type message and click 'Send' and message will show up on the terminal running the server	Figure 5 – Passed
6	Client able to receive messages from server	Send messages from other clients. Messages will display on the GUI of the client that didn't send the message	Figure 6 – Passed

7	Client able to see who the message is received from	Send a message and the appropriate username will be attached to the message for all clients to see	Figure 7 – Passed
8	Clients able to display all clients connected to server	Connect multiple clients to the server and you should be able to see the same list of clients connected on all clients' window	Figure 8 – Passed
9	Clients able to disconnect from server	Disconnect client and the list of clients connected will be updated on both GUI and terminal running server	Figure 9 – Passed
10	Client able to select file dump which creates a file of the current conversation	Select file dump when connecting to server. File will show up in the same folder as executable	Figure 10 – Passed
11	Not able to send a huge message as the GUI builder (QT) restricts it	Send a huge message. Client will crash due to restriction on QT.	Figure 11 – Passed

Figure 1

Figure 2

Figure 3

Figure 4

Figure 5

Figure 6

Figure 7

Figure 8

Figure 9

Figure 10