Name- Shivam Kumar Sareen

Student ID- 1001751987

****

About:

****

LAB 3- Vector Clocks

It is a multithreaded client/server Chat application based on a simple GUI which uses Java Socket programming.

A server listens for connection requests from clients across the network or even from the same machine.

Clients know how to connect to the server via an IP address and port number.

After connecting to the server, the client gets to choose his/her username on the chat room.

The client sends an asynchronous message, the message is sent to the server using I/O stream in java and the client doesn't wait for the acknowledgement from the server.

Java object serialization to transfer the messages.

REQUIREMENTS:

IDE used- Eclipse

1) Server:
   a) Should support at least 3 concurrent clients.
   b) Display details such as connected clients and the ongoing process.
   c) Should notify other clients about uploaded message.
   d) Allow other clients to receive the message.
   e) Logout from the server once the Logout button is pressed.
   f) Should not fail on unexpected crashes.

2) Client:
   a) Ask for username to log in. Reject if duplicate.
   b) Notify on getting connected.
   c) Sends the message using one of the messaging options (Broadcast, Multicast, Unicast)

The modes of transfer of Data(messages) within clients is through-

- Vector cocks

Receiving the message

Once the clients are connected, each client acts as a sender and randomly selects another client as the receiver between 2 to 10 seconds; updates its own clock and sends it vector clock to the server. The same clock is displayed on its GUI.

The server displays the vector clock sent by the sender and passes it to the receiver.

Receiving the message

The receiver upon receiving the message from the server, updates its vector clock by incrementing its own value. It also compares the clock of other clients with the values of other clients stored in its own clock, finds the maximum and sets the value its own vector clock and is displayed on the receiver's GUI.

****

Instructions:

****

Server

1. Run Server_GUI.java file

2. Click on the 'Start' button and Server will start on the configured localhost and port

Client

1. Run Client_GUI.java file

2. The Client will connect to the server, if the server is server was previously started, else it'll show a message saying the server is not connected.

3. Enter the Username for the client to connect to the Server

4. If the username is unique, the client will start the connection with the server, else it'll ask the user to enter a different username.

To implement multiple client chat, re-run the Client_GUI.java file

Chat

While in client console:

1. To send the message in the form of vector clocks, we connect 3 clients. As soon as the clients are connected, messaging will automatically start.

3. Click on 'Show Connected Users' button to see list of active clients

4. Click on 'Logout' button to logoff from server

Known Shortcomings

- The server is responsible for all the operations. Thus, if the server is closed before the clients, then the client status won't be able to communicate with the user. This might lock out the clients the next time the server is up until it is manually terminated.