Distributed systems Assignment 4 Designing and implementing a chat system for multiple clients

Aino Räkköläinen

Part 1: Design of the system

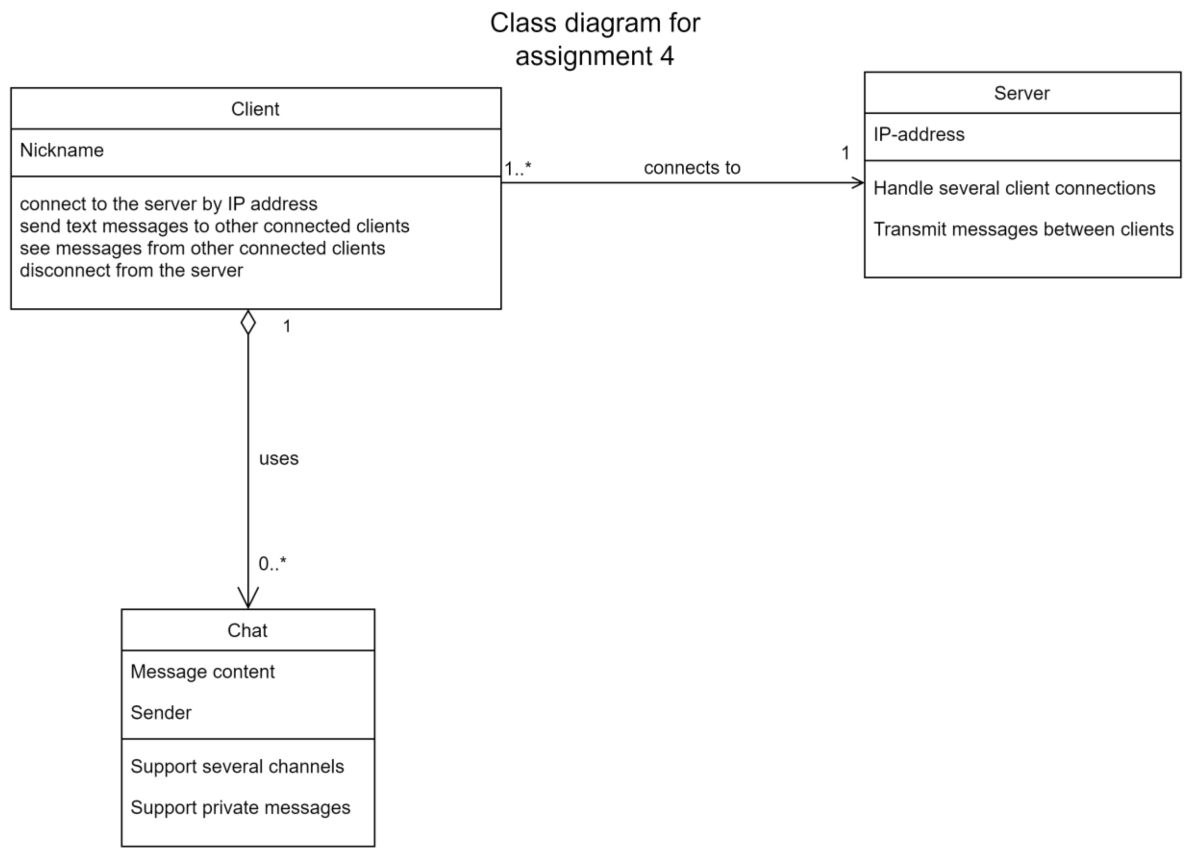
Class diagram for assignment 4
Client
Nickname
connect to the server by IP address
send text messages to other connected clients
see messages from other connected clients
disconnect from the server
1..*
connects to
1
Server
IP-address
Handle several client connections
Transmit messages between clients
1
uses
0..*
Chat
Message content
Sender
Support several channels
Support private messages

Figure 1. Class diagram for chat system

Figure 1. represents the class diagram for chat system developed for this assignment. The overall architecture of this system has these three classes, one for client, one for server and one for chat. In the system itself, those classes are just files, not classes themselves because this assignment is implemented with JavaScript. Additionally, Chat is just an abstract way to present the messages that are sent between clients. Thus, the actual Chat class or file is not implemented.

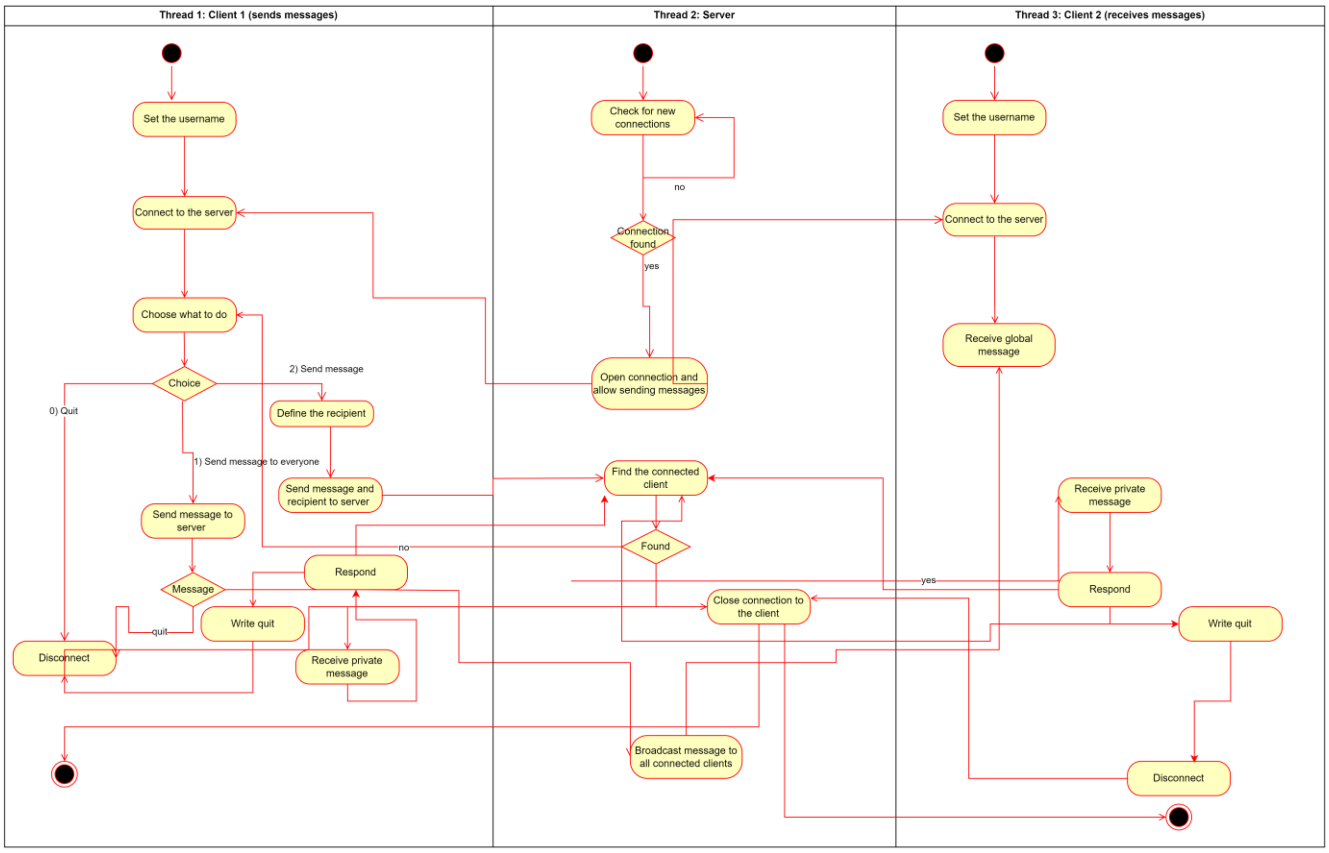

Figure 2. Activity diagram of two clients and server

Figure 2 demonstrates with an activity diagram how the system works when there are two clients connected to the server Activity diagram was used because it is typically used to demonstrate how different activities are used to implement a service (Visual paradigm, 2024). For simplicity, in this case only client 1 sends messages to everyone and in private to client 2 which client 2 can respond to. In the system itself, all clients have the same options to send messages to every connected client or one particular client. Server simply checks for connections and when those are found, it handles them and opens chat connection which is closed when client chooses to disconnect. Server also transmits the messages between two clients and also broadcasts all messages that are sent to everyone. The connection server opens are intended to be TCP connection because it can remain open until client chooses to close it and also if sending message is failed, it can be retrieved.

Server is in the middle of two clients in figure 2, because I thought that it implements the transparency of this distributed system. Client 1 doesn’t need to know how the server creates the connection or how it forwards the messages to client 2.

Part 2: Developing the system

System I developed in this assignment, is a terminal-based chat system which is mostly developed according to the tutorial video created by WittCode (2022). The tutorial by WittCode (2022) was used because it provided a good example of how to make a simple multi-client chat system and how to broadcast messages to all clients connected to the system. The private chat option works a bit similar with a difference in that way, but it checks from usernames list whether the recipient client is connected and if it exists, it sends messages only to that client, not everyone. However, both clients must have chosen option 2 to send and receive private messages. The idea is that clients might meet each other in the global chat and there decide to start chatting privately. A future improvement could be implementing a chat history system to save previous messages.

Few assumptions about development phase:

Currently, if username is already taken, the program closes, and the client cannot open the chat. I know it is not ideal that it requires always restarting if the given username is not accepted, but I didn't manage to make it work to ask for new username even though I tried many ways. Thus, for now clients have to restart the program until they find a username that is accepted to join the chat. It would be future improvement to make it ask for new username until one is accepted. I acknowledge that this would be really frustrating if it was a real chat application. Additionally, no chat history is maintained at all after clients are disconnected.

Note: Closing the program is a bit different in real implementation than in the design figure 2, but idea is still the same.

References:

Visual paradigm. (2024). What is Activity Diagram? [online document]. [Referenced 28th March 2024]. Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/

WittCode. (2022). Node Multi Client Chat. [YouTube Video]. [Referenced 28th March 2024]. Available at: https://www.youtube.com/watch?v=-rVxORKWzv0