## Design document

Design

The design of this application is robust and functional, with its core functionality executed in a command-line environment.  The application comprises two interlinked programs that will run together to create a chat application: a server file and a client file. The application should let the user create a username to be displayed on the command line when messages are sent.

The application will use java.net.Socket to communicate between two nodes. java.net.InetAddress to get the IP address. According to Schildt (2019),

“The InetAddress class is used to encapsulate both the numerical IP address and the domain name for that address. You interact with this class by using the name of an IP host, which is more convenient and understandable than its IP address.”

The socket represents the socket that both the client and server use to communicate with each other. The ServerSocket class is used by server applications to obtain a port and listen for client requests. The connection is identified by its IP address and port number. The application should prepare to communicate. The getInetAddress() will return the address if a socket is connected, if not a null is returned. Which will add to the functionality and help with any congestion issues. A socket can be created if the server address and port can be established.

java.net.SocketImpl is used to accept server connections using server.accept connections on the port using socket.getPort and to close them using the InetAddress. The socket uses an input stream and an output stream to display content. getPort()  will return the number of the port to which the socket is connected, if it is not connected to a port, it returns -1.

The port number is defined in the program and can be changed from there to make operations more seamless. PrintStream provides output from the system. According to Schildt (2019), “*PrintStream is an output stream derived from OutputStream, it also implements the low-level method write( ). Thus, write() can be used to write to the console.”*

BufferedReader uses an input stream reader(Sytem.in) to take input and uses a print stream to the socket.

Schildt (2019) states, *“TCP/IP sockets implement reliable, bidirectional, persistent, point-to-point, stream-based connections between hosts on the Internet.”* To follow onStallings 2007 notes: “*Stream sockets make use of TCP, which provides a connection-oriented reliable data transfer. Therefore, with stream sockets, all blocks of data sent between a pair of sockets are guaranteed for delivery and arrive in the order that they were sent.”*

The socket is closed using /q. The connection will be released. The users will not be able to send messages until a new connection is established. The message is read in a string called ‘message’. The two nodes will communicate until the socket connection is closed.

The client will be able to connect to the socket and read or write to the socket through the established connection.

Rationale

The outcome of this application is to communicate between nodes. The ChatServer should first be run, and then the client can connect from the ChatClient. Both users can pick usernames. The program just accepts text from the command line. The user connects through a socket connection and uses an IP address and TCP port to connect two nodes.

The message is passed from the inputstream to the outputstream using the socket connection.

The socket connection allows communication between the same machine or different computers on the same network. Schildt (2019), *“A socket can be used to connect Java’s I/O system to other programs that may reside either on the local machine or on any other machine on the Internet.”*

It refers to writing programs that execute across devices, in which the devices are all connected to each other using a network.

Socket programming can be easily implemented for general communications. It causes low network traffic. Socket based communications allows packets of raw data to be sent in-between applications. Both the client-side and server-side must provide mechanisms to make the data supported.

Java has an independent platform and network libraries to suit the application. The application can be maintained and is scalable. The application is suitable for communicating between nodes in a distributed system and processing clients making requests from the server.

The application should be compiled and run in the command line. Each file needs to be compiled. The server needs to be started first and will need to wait for the client to connect. The client can connect when the client file is compiled and run on the command line. After these steps, the functionality can be met.

References

Schildt, H. (2019). *JAVA : the complete reference.* S.L.: Mcgraw-Hill Education.

Stallings, W. (2007). *Data and computer communications*. Upper Saddle River, N.J.: Pearson/Prentice Hall.