NETWORKS ASSIGNMENT 1:

Alexander Murie; Nathan Esra; Thabo Kopane

FEATURES

Broadcasting a message: The client is able to broadcast a message to the all clients connected on the socket. This feature forms the basic functionality of our chat room.

Broadcasting a file: The client is able to broadcast a file to the entire chat room.

Private message: The client sends a message to another client such that only that client can see it. In order to send a message to one (only one) particular client, one would ordinarily have to connect to a "private" socket with that client to chat. However, this feature allows this while still in a larger group chat, and thus was a worthwhile inclusion.

Privately send a file: The client sends a file to one particular client. This allows for client discretion in sending files and prevents possible security leaks. This also accommodates low bandwidth users in that they won't be forced to download *every* file that's broadcast.

Block a client from receiving a message: The client sends a message to the entire chat room, *except* one particular client (the inverse of a private message, in a sense). This was included for the discretion of the clients as well as allowing, in a sense, a "subgroup chat room" within a larger chat room.

Block a client from receiving a file: The client sends a file to the entire chat room, except one particular client (inverse of privately sending a file). This allows for a file to be sent to multiple people at once while not necessarily sending to everyone.

PROTOCOL SPECIFICATION

Starting a Server from the command line: java Server [port number]

Starting a Client from command line: java Client [client number] [port number] [Server IP]

Message structure:

broadcast message: broadcast message "[message]"

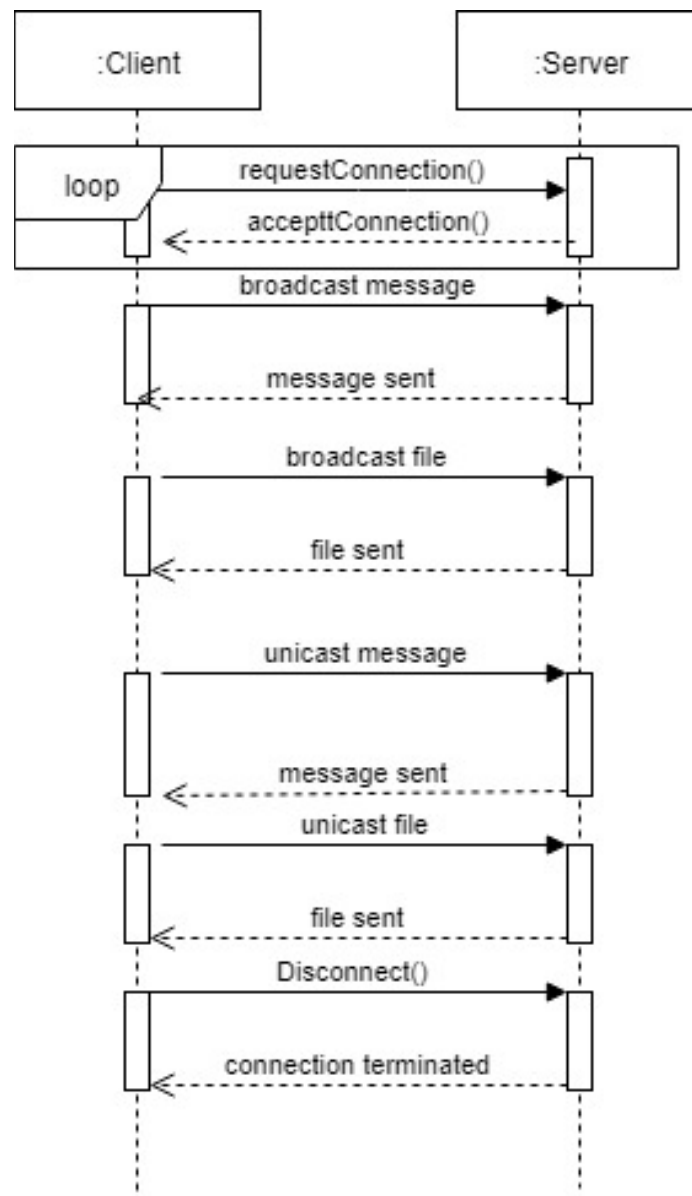**broadcast file:** broadcast file "[file path]"

private file: unicast file "[file path]" [target client]

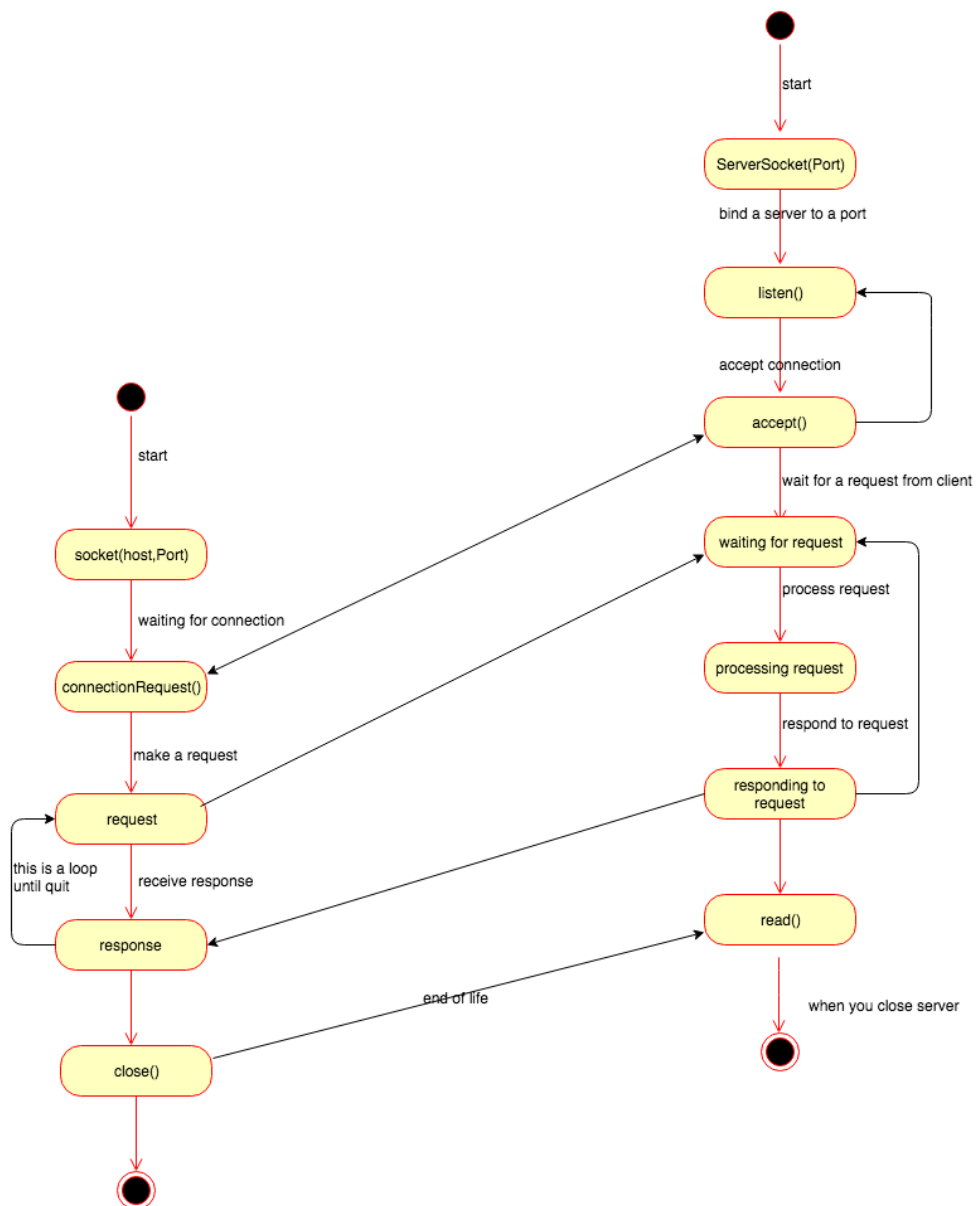**private message:** unicast message "[message]" [target client]

block cast message: blockcast message "[message]" [blocked client]

block cast file: blockcast file "[message]" [blocked client]

DIAGRAMS: A system Sequence Diagram of Client-Server Communication.

DIAGRAMS: A state machine

start

ServerSocket(Port)

bind a server to a port

listen()

accept connection

accept()

wait for a request from client

waiting for request

process request

processing request

respond to request

responding to request

read()

when you close server

start

socket(host,Port)

waiting for connection

connectionRequest()

make a request

request

this is a loop until quit

receive response

response

close()

end of life

DIAGRAMS: A Sequence Diagram showing the full communication