Non-functional Requirements

System

Send messages between two users.
Server must be available to remote clients.
Messages delivered in less than 1 second, ignoring abnormal network latency.
Communications between users are secured.

Server

Server must support at least 10 simultaneous clients.
Server must be available 90% of the time.
Server must provide a list of online contacts.

Client

Usable by users not comfortable in a command shell.
Able to chat with multiple other (remote) clients from a single local process.

Functional Requirements and Use Cases

Server Startup

Associated non-functional reqs: Server avail. to remote clients, avail 90% of the time
Actors: operator
Inputs: compiled server application
Outputs: server ready to receive messages
Normal Operation: Operator issues command to start server application. Server application starts and begins listening for client connections on a well-known port.
Exceptions: Port could be in use and server will fail to bind, operator will need to investigate or change port number. Changing the well-known port number would need to be announced to all clients somehow.

Server Accept New Connection

Associated non-functional reqs: Server avail. to remote clients, server support at least 10 clients
Actors: client
Inputs: login message
Outputs: record of active user, status message

Normal Operation: Client sends login message to server. Server receives the new connection and tracks it in a list of active client connections. Server associates the logged on username with the connection.
Exceptions: Username could already be taken, in which case server sends back and error and drops the connection.

Server Forwards Communications Between Users

Associated non-functional reqs: Server support at least 10 clients, messages delivered in <1 second
Actors: client A, client B
Inputs: chat message from user A
Outputs: chat message to user B
Normal Operation: Client sends chat message to server, addressed to another logged on client. Server looks up the receiving client and forwards the chat message to that user.
Exceptions: Receive chat message from user who is not logged on or to user who is not logged on. In either case, drop the message.

Server Recycles Usernames

Associated non-functional reqs: Server support at least 10 clients
Actors: client
Inputs: client connection closed
Outputs: associated connection removed from map and username available again
Normal Operation: Client sends logoff message to user or unexpectedly goes offline. Server notices and removes the client from collection of online users.
Exceptions: Receive logoff message for user who is not currently recorded as online. Ignore the request and close the associated connection.

Server Provides List of Online Contacts

Associated non-functional reqs: Server provides list of online contacts
Actors: client
Inputs: client request for contacts
Outputs: message to client containing list of online contacts
Normal Operation: Client sends contacts query to the server. Server interprets the message and replies with a message containing a list of all the online contacts that can be chatted with.
Exceptions:

Client Startup

Associated non-functional reqs: Send messages between users
Actors: user
Inputs: user start application, desired username
Outputs: client application ready to sent messages
Normal Operation: Client prompts user for username and uses it to log in to the server. After login client downloads list of online contacts from the server.
Exceptions: Username is taken, client should ask for another name.

Client Command Handler

Associated non-functional reqs: Client can chat with multiple remote clients
Actors: user
Inputs: user command
Outputs: appropriate action for command dictated by the user
Normal Operation: User enters a command for interpretation by the client application. The client interprets this command and performs the expected action. Supported commands should include switching conversations, logging off. Commands probably need a particular escape character to be recognized as such.
Exceptions: User enters a command that is not understood. The client should display an appropriate error and be ready for new input.

Client Send Message

Associated non-functional reqs: Client can chat with multiple remote clients, send messages to remote clients
Actors: user, server
Inputs: user-entered chat message
Outputs: message to server for dispatching
Normal Operation: Client application is in a state to receive a chat message for a particular user. User enters a chat message. The client application constructs the appropriately formatted message and sends it to the server.
Exceptions: Remote user/contact has gone offline. Client should notify the user.

Client Receive Message

Associated non-functional reqs: Client can chat with multiple remote clients, send messages to remote clients
Actors: user, server

Inputs: chat message from server
Outputs: message displayed to client
Normal Operation: Client receives a chat message from the server. It's parsed to learn the sender and displayed to the user.
Exceptions: Message is of invalid format. That'd be an internal error, so possibly display an error message to use. Drop the message.