Group 3

Maximilliano Cabrera
Tanmay Narang
Maria Ramos Garzom

Software Requirements Specification

Table of Contents

1. Purpose

This document outlines the requirements for the Multi Person Chat System (MPC).

1.1. Scope

This document will catalog the user and system requirements for the MPC system. It will not, however, document how these requirements will be implemented.

1.2. Definitions, Acronyms, Abbreviations

Instant messaging (IM)
Private message (PM)

1.3. References

Use Case Document

1.4. Overview

The Multi Person Chat System (MPC), is designed to allow for users to connect to chat rooms and communicate with one another via instant messaging. Users can also private message one another.

2. Overall Description

2.1. Product Perspective

The system will be open-source. It is a web based system that implements the client-server model. It enables simple text communication between users.

2.2. Product Architecture

The system will be organized into 2 major modules: the Server module and the Client module. System architecture will follow standard OO design practices.

2.3. Product Functionality/Features

It will support the following features:

- Chat rooms
- Direct messaging from one user to another

2.4. Constraints

Will be developed using java.

2.5. Assumptions and Dependencies

2.5.1 Assumptions

There must be a minimum of two active users for a chat to begin

2.5.2 Dependencies

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements: System should have users login. They can only join a chat room or logout after a successful login. Users can join existing chat lobbies. Chat lobbies will allow for users to IM a group of people and from the people in the chat room they can start a private chat. Users can disconnect from these lobbies at any time.

3.1.2 Client Module Requirements: Client should allow users to login. Users should be able to create their username. This module should handle all the program to user interaction.

3.1.3. Server Module Requirements: The server will deal with the client requests and back end processing. Handling most of the processing will go to the server, determining what information goes where and which client will receive what data needs.

3.2. External Interface Requirements

3.2.1 Provide an interface for users to login with their credentials.

3.2.2 Provide users with an interface to PM other users separate from the chat room.

3.2.3. Provide users with an interface for chat rooms connecting, disconnecting, and messaging

3.2.4 Provide a graphical user interface for the user

3.3. Internal Interface Requirements

3.3.1. Process incoming user login information against a list of usernames

3.3.2. Handle PM data going to appropriate users both to and from both users.

3.3.3. Handle the active chat rooms and clients connected to them.

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.2. Environmental Requirements

4.3. Performance Requirements

Use Cases Document

Actors – User, Operator

Use Cases -

- Chatting with individuals: the application will create a query window for chatting with users
- Chatting in channels: users can chat in channels with several other users
- Login of users with specific usernames
- Operator can shutdown the server or kick users offline through command line

Use Case ID: 1
Use Case Name: Group Chat
Relevant Requirements: N/A
Primary Actor: User
Pre-conditions: The user is logged in and connected with a unique username.
Post-conditions: The user can send and receive message to several channels or users.
Basic Flow or Main Scenario:

1. The user logs in with a unique username.
2. The user joins a channel with a specific command or through a GUI prompt.
3. The user can send and receive messages to that channel.
4. The user can join other channels.

Extensions or Alternate Flows: N/A
Exceptions:

1. A user logging in with a non-unique username.
2. A user starts spamming the server and it crashes.

Related Use Cases: Individual Chat

Use Case ID: 2
Use Case Name: Individual Chat
Relevant Requirements: N/A
Primary Actor: User
Pre-conditions: The user is logged in and connected with a unique username.
Post-conditions: The user can send and receive message to one or several users directly.
Basic Flow or Main Scenario:

1. The user logs in with a unique username.
2. The user starts a direct message with another user without joining a channel of users.
3. The user can send and receive messages to that user specifically.
4. The user can join other channels.

Extensions or Alternate Flows: N/A
Exceptions:

1. A user logging in with a non-unique username.
2. A user starts spamming the server and it crashes.

Related Use Cases: Individual Chat

Use Case ID: 3
Use Case Name: Server Shutdown
Relevant Requirements: N/A
Primary Actor: Operator
Pre-conditions: The server is up and running.
Post-conditions: The server will be shutdown.
Basic Flow or Main Scenario:
1. The operator logs onto the server hosting the chat daemon.
2. The operator shuts down the daemon.
Extensions or Alternate Flows:
1. The operator runs a specific shutdown command with an authentication key.
2. The server shuts down.
Exceptions:
1. The operator forgets their authentication key.
Related Use Cases: N/A

Use Case ID: 4
Use Case Name: User Kicked
Relevant Requirements: N/A
Primary Actor: Operator
Pre-conditions:
1. The server is up and running.
2. The user is on the server and the operator knows their username.
3. The operator can login to the console of the chat daemon or has an authentication key.
Post-conditions: The user will be kicked off the server.
Basic Flow or Main Scenario:
1. The operator logs onto the server hosting the chat daemon.
2. The operator issues a command to kick the user off the server by their username.
Extensions or Alternate Flows:
1. The operator runs a specific kick command with an authentication key.
2. The server kicks the user out.
Exceptions:
1. The operator forgets their authentication key.
Related Use Cases: N/A