Design Document

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 3/11/24 | 1-4 | Initial Version | Derion |
| 3/16/24 | 1-2 | Creation of Uml class diagram (Client, Server) | Derion |
| 3/28/24 | 1-5 | Editing Message and ChatMsg info | Anya |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[**1.Design Phase 4**](#_30j0zll)

[1.1. Scope 4](#_1fob9te)

[1.2. Definitions, Acronyms, Abbreviations 4](#_3znysh7)

[1.3. References 4](#_2et92p0)

[1.4. Overview 5](#_tyjcwt)

[**2.Uml Design 6**](#_3dy6vkm)

[2.1. Client Class 6](#_1t3h5sf)

[2.2. Server Class 6](#_4d34og8)

[2.3. User/Admin Class 6](#_2s8eyo1)

[2.4. Server/Client Class 6](#_17dp8vu)

[2.5. Message Class 6](#_26in1rg)

[2.6. ChatMsg Class 6](#_26in1rg)

[**3 Interactions. 8**](#_lnxbz9)

[3.1.Class Interactions 8](#_35nkun2)

[3.1.1. Common Requirements: 8](#_1ksv4uv)

[3.1.2. Message Requirements: 8](#_z337ya)

[3.1.3. Server Requirements: 9](#_2xcytpi)

[3.1.4. Admin Requirements: 9](#_6q66152yknoj)

[10](#_1hmsyys)

# Design Phase

This document outlines the requirements for the Communication System.

## Scope

This document will catalog the different designs that will be used to help implement the different functionalities in our code

## Definitions, Acronyms, Abbreviations

* Normal user: In this document, a normal user refers to employees of the organization who are not IT staff. In other words, a normal user does not have the IT admin privileges in this Communication System.
* IT user: IT user refers to IT staff, which not only has all the features of a normal user, but also has privileges like checking server logs.
* User: In this document, “User” refers to all users in this Communication System, including both Normal users and IT users.
* Synchronous: In this Communication System, the word “synchronous” refers to the situation where two or more users are online and any messages among them are sent or received synchronously.
* Asynchronous: The word “asynchronous” refers to the case when the sender is online and the recipient of the message is offline. In this case, any messages between them are sent or received asynchronously.
* Group owner: Group owner is the owner and administrator of a group chat. Group owner can accept new people to the group, remove people from the group, and delete the group completely. The person who creates the group is automatically the group owner. Group owner can appoint another person in the same group to be the group owner. There will be only one group owner for each group chat.
* Event: In this document, event refers to all the traffic in and out of the server, including messages, login/logout requests, etc.
* Server application: server application refers to the Java application running on the server, which is responsible for distributing messages, responding to all users, and logging events.
* Client application: client application refers to the Java application with GUI that users use. Client application is mainly responsible for user login/logout and send/receive messages.
* Online status: online status refers to whether a user is online (logged in the system and has stable connection with the server) or offline (logged out of the system or loses connection with the server).
* Availability status: availability status is a status that the user can set as he wants to indicate his availability at the moment. Availability status could either be “idle” or “busy”.
* User status: User status refers to both online status and availability status.

## References

(New) Use Case Specification Document – Step 2 in assignment description

UML Use Case Diagrams – Step 3 in assignment description

## Overview

The design document provides a comprehensive overview of the messaging application architecture, components, and interactions. By following this design, developers can efficiently implement a scalable, secure, and user-friendly messaging platform.

# Uml Design

## Product Perspective

## Product Architecture

The system 3 major class implementations

Product Functionality/Features

The high-level features of the system are shown in the following uml diagrams

2.3.1 We will be providing methods and attributes of each class with the given diagrams made

2.3.2 The server application is able to log all and will use a log class to use or view those logs whether being group or regular message

2.3.3 Users will be completely separate from admin they will be limited with their functionality

2.3.4 Not introducing a blocking or kicking function for more fluidity and simplify the code

2.3.5 The server and client application should be able to pass Message objects over the network with timestamps.

## User/Admin diagrams

2.4.1 User possesses the basic functionalities of the system messaging and viewing contact information

2.4.2 Server blocks user ability to remove user or create user or view logs stored in server

2.4.3 All users cannot log into the same account from different IP addresses at the same time.

2.4.4 The system must support message delivery both synchronously and asynchronously.

2.4.5 Every user has a online and offline status that helps determines the process of messaging, showing online only when user has established a connection to the server

2.4.6 The client GUI interface should have basic functionalities of login/logout, send and receive direct messages and group chats.

2.4.7 All chat messages are only text messages. No image, video, file are supported.

## Server/Client diagrams

2.4.1 The diagram will show the interaction between the server and client.

2.4.2 Show how the code will handle the amount of clients in the server and how it will collect and store messages to the server

2.4.3 Server stores the contents of the message onto a txt file that only can be accessed by the admin

2.4.4 The messages will be logged as a string in the server class

2.4.5 Server will keep track of the status of the user (offline, online)

* 1. Client has a unique ID to represent user
  2. Connection is identified with a boolean, this attribute is for helping with connection management
  3. The server object will allow for the client to interact with server in storing/saving messages
  4. Message list allows client to show messages to user
  5. Server holds time with date object
  6. Server holds connected clients in client list
  7. Request Connection is responsible for handling the process of accepting connected clients

**2.5 Message diagram**

2.5.1 Messages utilize a “MsgType” enumerator to identify the given message’s purpose:

whether it is a login, logout, direct message, etc.

2.5.2 Messages will implement Serializable to be usable by the Server and Client

2.5.3 Messages contain a string of text, which may be used when the Message is carrying

a ChatMsg, as well as a date for the timestamp

**2.6 ChatMsg diagram**

2.6.1 Contains text for the chat message, as well as a getter for said text

2.6.2 Contains a Date type for a Date class, for timestamps

2.6.3 ChatMsg will be used in both DirectMessage and GroupChatMessage Message

types

# Interactions

## Functional Requirements

### Server and Client interaction:

3.1.1.1 All messages received in client applications will be cleared after the user logs out.

3.1.1.2 Users should be able to delete messages locally, including direct messages and group chats, such that deleted messages will not show up in the client application but the messages logged in the server will not be deleted.

3.1.1.3 Users should be able to mute direct message conversations and group chats, in a way that they will receive the messages but not get notifications.

3.1.1.4 Users should be able to unmute direct message conversations and group chats if they muted them before, and start getting message notifications whenever they receive a new message from those conversations.

3.1.1.5 Users should be able to view the directory of all users in the system.

3.1.1.6 Users should be able to view other users’ online statuses (online or offline) and availability status (idle or busy).

3.1.1.7 Users should be able to change their availability statuses (idle or busy) whenever they want.

3.1.1.8 Users’ online statuses are only based on whether they are logged in the system or not and cannot be changed by users.

### ChatMsg Module Requirements:

3.1.2.1 Users should be able to receive messages that are sent synchronously.

3.1.2.2 Users should be able to receive messages that are sent asynchronously.

3.1.2.3 Users should be able to send messages to other users regardless of whether other users are online or offline.

Users should be able to initiate a new conversation with any users in the system.

3.1.2.4 Users should be able to accept invitations to join group chats.

3.1.2.5 Users who are members of the groups should be able to send requests to invite other people to join group chats.

3.1.2.6 Users should be able to create new group chats and they will automatically become the Group Administrators of the new group chats.

3.1.2.7 Every group chat will only have one Group Administrator, who is also the person created the group.

3.1.2.8 Only users who are Group Administrators of the group chat can decide whether to let certain users join.

3.1.2.9 Only users who are Group Administrators of the group chat can remove members from the group chat.

3.1.2.10 Users should be able to leave any group chats if they want.

3.1.2.11 Every Group Administrators of the group chats can appoint one team member to be the new Group Administrator, and there will be only one Group Administrator at a time.

### Server Module Requirements:

* + - 1. All events (both inbound and outbound traffic through the server) will be logged in the server with timestamp, including all messages (including direct messages and group chats), login/logout requests and other bad requests.
      2. Before the server shuts down, all logged events should be stored in a txt file in local and can be restored after restarting the server.
      3. The server application is multi-threaded, allowing many users to connect it at the same time.

### Admin Module Requirements:

3.1.4.1 IT users should be able to view all the logged events stored in the server.

3.1.4.2 Only IT users can create new users.

3.1.4.3 IT users should be able to delete existing users and automatically remove them from all the group chats they belong to. Any attempts to send new messages to these deleted users will be rejected and those users will receive an error message.

## External Interface Requirements

3.2.1 The system must provide a GUI interface for all users.

3.2.2 The GUI supports basic functionalities, including login/logout, send/receive messages, with no animations.

3.2.3 The GUI for an IT user is the same with a normal user, except that there are a few more features like view server logs, create and delete users.

## Internal Interface Requirements

3.3.1 The server application should be able to communicate with the client application over TCP/IP.

3.3.2 The communication between server and client is by sending Message objects over the network. Different functionalities (including login, direct message, group message, directory, online status, etc) correspond to different Message types.

3.3.3 Each Message object has a GMT (Greenwich Mean Time) timestamp associated with it.