COMP90015: Distributed System – Assignment 2

Scrabble Game

ShaoChuan Luo No.952179
Wendong Chen No.931018
Yuming Lin No.883717
Peiyang Li No.897054

Tutor: Lakshmi Jagathamma Mohan

1. Introduction

The main goal of this report is to record game development content and game rules in detail. The report firstly records the UML class diagram, interaction diagram of the program and give the explanations of the diagram. In addition, the architectural design of the server, as well as the communication protocol and the transport information format, are recorded. Finally, this report documents the implementation details and execution steps of the server, client, and game rules.

The Distributed Scrabble game consists of a server on the client, and the game can be opened only when the server is turned on. The game's interface is a 20x20 checkerboard grid that allows players to play on different computers. The game has a minimum of two players and allows more players to play in a game at the same time. Additionally, the function of simultaneously playing games in a plurality of game rooms is also implemented.

Each player has a unique username. Players can see the current room list and online players, then perform the invitation game, join a room and create a room. Each player can only join the game at the beginning of one game. If the game is in progress, players cannot join the middle. Words can only be 26 English letters, and new words can only be placed next to the grid of existing letters. When a word is placed in a grid of different colors, you can get more score depends on the color. Finally, based on other users voting to determine the correctness of the word.

2. Architecture

The UML diagram of the system is shown in Figure 1. It consists of six classes. The server part is composed of Server, ClientConnection and ServerState. Server is the server main body, which establishes a ServerSocket and Whenever a legitimate user requests a connection, it creates a thread (ClientConnection) to serve the user. Among it is the main method, which is the entrance to the entire server. ClientConnection is the connection thread to the client and established for the

connection of new clients. ClientConnection instances provide client with various services including accepting and sending a variety of information, update game state, hall or specific room state, and so on. After receiving the client's close command, the thread corresponding to the client is automatically destroyed. The role of the ServerState is a thread manager who records how many connection threads are currently working on the server. Which thread was created and which thread was destroyed. And it manages two data sets, one is a data set containing a list of all current online rooms, and the other is a data set of a list of rooms in the game state.

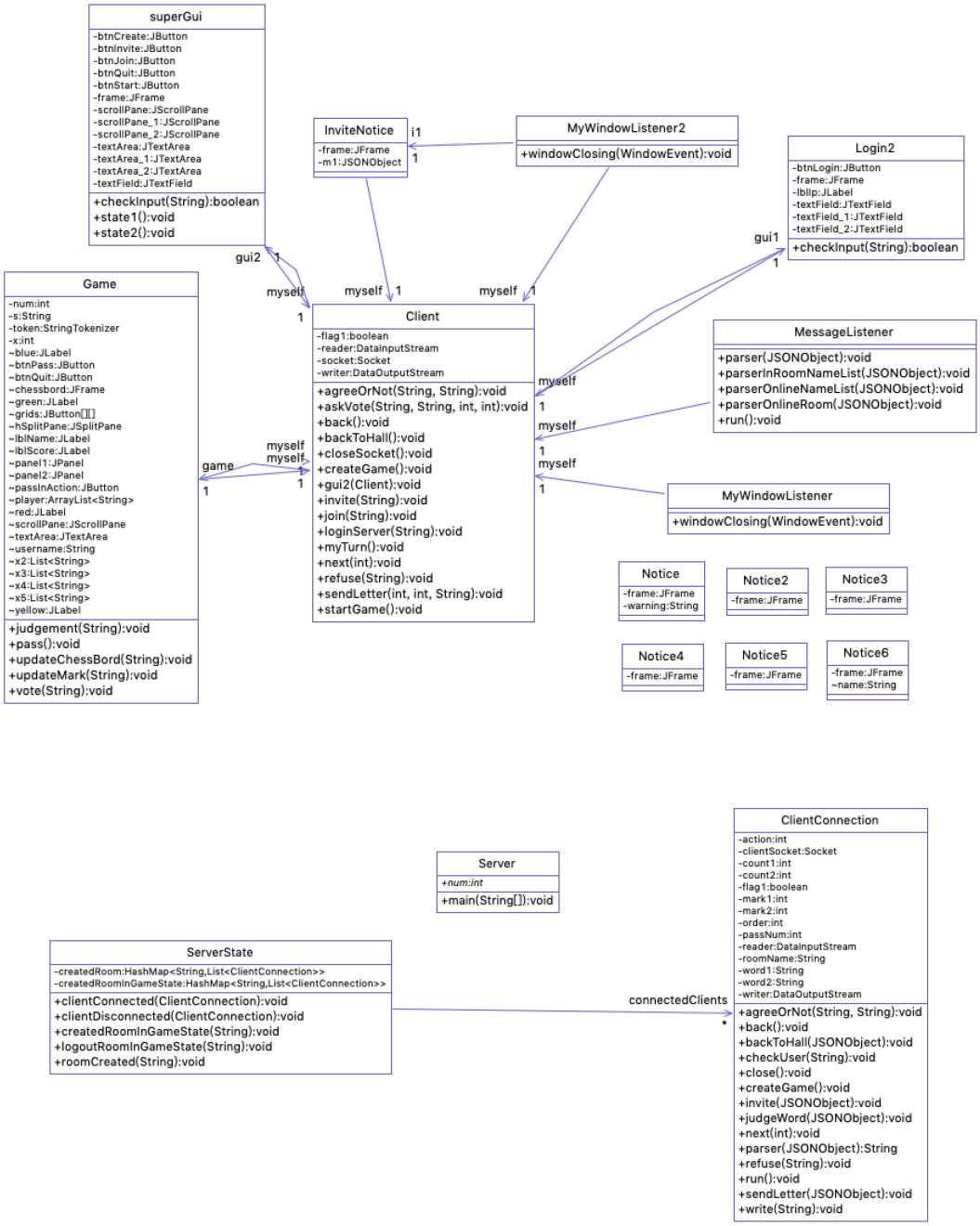

Figure 1: UML Diagram of the System

The client part consists of fourteen classes. The core classes of the client are Game and Client. The game class is a game entity that includes the client GUI in game state and various game rules. After user logged in successfully, the superGui class is the GUI interacting with the user when client is in a non-gaming state, the user can issue an invitation, create a room, and start a game through the interactive interface. MessageListener is a thread class. Whenever a new user successfully logs in to the server, the client process creates an MessageListener thread to listen for various messages from the server. At the same time, the thread is also responsible for processing the information. The thread is destroyed when the client is closed. Login2 class is the login interface after the user starts the server. The user can confirm whether his or her name is unique at this time. InviteNotice class is the prompt GUI that pops up when the user receives the invitation. The user may have accepted or rejected the invitation through the interactive interface. From the Notice class to the Notice6 class, are various prompt boxes for different situations such as disconnection. The MyWindowListener class and MyWindowListener2 class are for the user to click the "x" key of superGUI (disconnected from the server) and the user clicks the "x" key of the invitation prompt (reject invitation). The client class is the soul of the entire client, its main role is to accept calls from a number of human-computer interaction interfaces and send information to the server. It is also called by the listening thread to process the information sent by the server.

3. Protocols and Messaging Description

In this project, we use the JSON format to deliver and receive messages. I will introduce the types of messages and their meanings from perspectives of the client and server respectively. The content corresponding to the msg key indicates the message type and the corresponding situation of the client.

Protocol of client’s perspective (non-game state):

Msg: 1
This means that a user has invited you to join his room. The message contains the user's name and his room number.

Msg: 3

This means that the invitation failed. The reason may be that the user does not exist (the message contains the "noExist" key); the user is not in the hall and is in the game state or has entered the room (message contains the "name" button); or the invitation is rejected (message contains the "Name" button).

Msg: 4.

This means that the server allows the user to set up a room. The message contains a list of new online people, a list of people in the room, and a welcome message.

Msg: 6.

This means that the user has successfully logged in and entered the hall. The user will receive a welcome message, a new list of online people, and a list of online rooms.

Msg: 7

This means that the user will update the list of people in the room.

Msg: 8

This means that whether the user can join the room. If the content of the key "permission" is "noRoom", then the room does not exist; if the content of the key "permission" is "no", the user in the room is already in the game state and cannot join the room; except above cases, the user is allowed to join the room and get a new list of users in the room.

Msg: 9

This means that the user will update the list of people on the server.

Msg: 10

This means that the user will update the online room list.

Msg: 11
Received a server warning. There is already a user with the same name in the server.

Game part (message format is "00X"):

Msg: 001
If the message contains the "permission" key, that is the number of people in the room is not enough to start the game, otherwise the user enters the game.

Msg: 002
This message contains two strings and the user is asked to determine if the two strings are words.

Msg: 003
This message tells the user that other users have updated the board.

Mag: 005
This message means that the game has entered this user's turn.

Msg: 006
This message means that the scores of everyone in the room will be updated.

Msg: 007
This message means that the user exits the game and returns to the hall, the user gets a message to end the game, a new list of online people, and a list of online rooms.

Msg: 008
The user updates the online person message as well as the online room message.

Msg: 009
This message tells the user which user left the game.

Protocol of perspective from server (non-game state):

Msg: 1
The server receives an invitation request from the user. This message contains the name of the invited user. The server will process it.

Msg: 3
The server received a message that this user rejected the invitation from another user.

Msg: 4
The server receives the request from the user to create a room.

Msg: 5
The server receives the request that the user wants to disconnect.

Msg: 6
The server receives the user's login request and checks if the user's username is unique.

Msg: 7
The server receives the request from the user to return to the hall from the room.

Msg: 8
The server receives a request from the user to join a room. This message contains the name of the room.

Game part (message format is "00X"):

Msg: 001
The server receives a request from the user to start the game.

Msg: 002
The server receives a request to initiate a vote and contains two words about the vote.

Msg: 003
The server receives the request from the user to update the board.

Msg: 004
The server receives a response from the user about whether a string is a word.

Msg: 005
The server receives information that the user completed its round.

Msg: 007
The server receives the request from the user to return to the hall from the game.

Note: In the communication agreement, a part of the message number is skipped.

4. Design

For this project, we choose Server-Clients structure and Socket programming. The module design of client and server bases on Socket, Thread and JSON, read-write streams. Users enter the login interface by running Client.java. Since the user creates the login page, client sets up a thread in server and creates Socket listening message from server persistently. Server and client use JSON to exchange messages. Client use GUI to exchange messages with players.

There are several different functions in the system, but all of them can be described as the following process. The players provide commands via different GUI. In each action listener, it links to server with the specific method in client. And then Client takes different actions to response users, including send messages to server. After that, server disposes commands and gives feedback to client. The message listener disposes the JSON messages received from server and shows feedback to users via GUI. The whole process is shown in Figure 2.

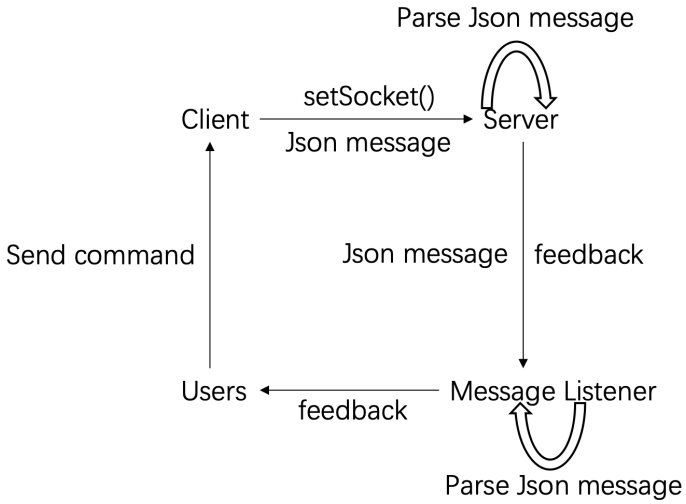

Figure 2: Working Process of Server-Client Structure

Login function is a good example to explain this process. If user logins successfully, his name will be sent to server via socket. Server will name this user with the message client sent. In the login process, server does not offer feedback actions or message to client, so the interaction between them completed in these simple steps. After login, client will create a new GUI for users. It is the main functional interface before starting a game. It has information about the number of online people, people in current room and messages between server and current user. In this page, user can choose to create a new room or join the other room (as showing in Figure 3).

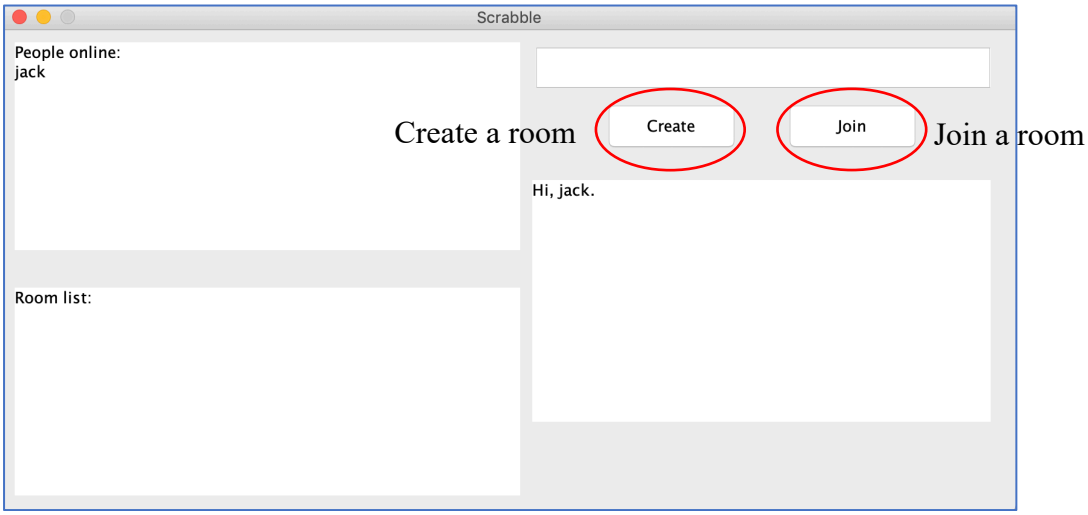

Figure 3: The Main Functional Interface

The functionality of sever in game is explained as following. Before the user starts the game, first run the Server class to start the server. When the game server is turned on, the server listens for the sockets sent from the client. If a connection message is sent, the server accepts the connection request. Then receive the incoming socket through the loop accept(). When a new socket is accepted, a new ClientConnection thread is created, and more details such as one user and the user's IP are displayed in the server.

The ClientConnection Class extends the Thread class. This class is used to process different message sending by the client and then execute the corresponding methods. Firstly, the constructor creates a data stream input and output, and then get the command sent by the Client. There are different methods in the ClientConnection Class class. The corresponding operation instruction is obtained by parsing the message sent from the players. Therefore, the program can call different methods and perform different operations.

We use the ServerState class to store the connected client information. ServerState is equivalent to a small database. We use an Arraylist and two HashMaps to record connected Client numbers, created rooms, and rooms in the game. Because of the ServerState class, the server can monitor the connected threads and complete the creation of the room, invite players and other functions.

The rules and policies of the game are called by the client, and each client that starts the game calls the Game class. The Game class initially generate a game board, the game board consists of a 20x20 JButton. There are 8 red squares, 16 green squares, 28 yellow squares and 40 blue squares on the board. Representing five times the score, four times the score, three times the score and twice the score. Players can only select one grid at a time and put an English letter. If you enter more than one letter or non-alphabetic characters, the system will let the player re-enter. In addition, players can only enter new letters around a lettered grid. Whenever a new letter is entered by the player, the system automatically captures horizontal and vertical words until it finds a null grid. Next, the system asks the player whether to vote. If the player chooses to start voting, the system will send the words obtained by the horizontal and vertical to other players and accept the voting information. If the vote is successful, the corresponding word score is calculated, otherwise it is not scored. If the player refuses to start voting, the next round is will begin.

In each round, players can choose to skip this turn. If the player chooses to skip, the game round of the next player is started. If all players choose to skip, the game ends. If all the grids are filled, the game ends.

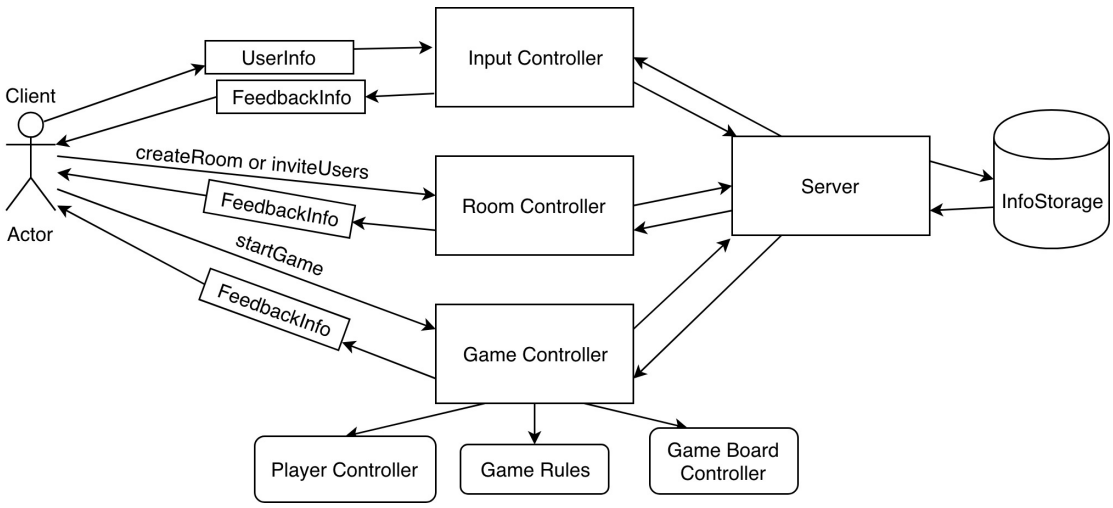

Figure 4: Interaction Diagram

The interaction diagram is showed in Figure 4. There are three parts interacting with client, including input controller, room controller and game controller. Input controller acts on checking the validity of user input. If the messages user input

is valid, it will send the information to server. If the messages are invalid, the controller can publish prompt messages via GUI for users. For example, input controller is used in verifying the existence of input username when client login the system and verifying the validity of input string. Room controller acts on creating rooms and inviting players. If client sends a command of creating a new room, room controller sends it to server in order that server responses accordingly. The other function of room controller is that accepting the command of invitation. Once client enters a room successfully, he can invite another online user to join this game. Room controller sends the information of this application to server, and then server broadcasting it to the appointed user. If the invited user accepts the invitation, server will add him into the current room. Otherwise, room controller will send notification to the applicant.

Another functional part is game controller. It acts when client sends out the commands about game. Game controller includes player controller, game rules and game board controller. Player controller is used in recording the information about players, such as the score of each player and the status of players. Game rules contains all the potential rules of the scrabble game. It can provide players with instructions and calculate score for every player. Game board controller is the GUI for game. It is special because the GUI of game board is designed as 20*20 grid button. Thus, this controller can control the editability of each grid button and change the properties of the button to realize visualization of input character.

The information of online users and online rooms is stored in information storage. When server needs the previous information, it can pull data from the storage. In this project, the profile stored in information storage will be rewrite after each action. After that, server can broadcast the new profile to all the users.

The GUI of the system is designed with Java Swing. The main output plug-in is TextArea and the input plug-in is TextField. The GUI of login, room page, invite page are similar, which are normal user interaction interface. The grid

board of game is combined with 400 buttons. All the buttons are editable at first. If a button is clicked and player changes the value of it, then the button will be changed to be no editable.

5. Contributions

Name	Which part	Contribution rates
Wendong Chen	Game	25%
Shaochuan Luo	Server+Client+report	25%
Yuming Lin	Game+report	25%
Peiyang Li	GUI+report	25%

Appendix

The following appendix is the guide of user basic operations.

- Login

After the user starts the client, he should enter the name, IP and port of server to log in to the system (IP: IP of server computer, port: “4444”). The id cannot be null and the content cannot contain whitespace characters. If a user with the same name already exists in the system, the new user is denied to log in with that name and a pop-up warning is issued.

- Hall and Room

1. After successfully logging in to the server, the user obtains a list of the latest online people and a list of online rooms. The user can choose to join or create a room now.

2. After the user creates the room, he gets the latest list of people in the room and a list of all online people. Users can make invitations based on the list of online people. The invitation bar cannot be null and the content cannot contain space characters.

3. The people in the rooms can exit the room at any time to return to the lobby (returning to the lobby means that the user is in a state where they can create a room or join the game) and get a list of the latest online people and online room list. The users in the room will update the user situation in the room.

4. The users in hall can choose to join the room according to the online room list. If the players in the room are already in the game state, they cannot join the room and a pop-up warning appears. If the room does not exist, a pop-up warning will also appear.

5. After the user joins the room, he gets a list of the latest online people and a list of people in the room. Users can make invitations based on the list of online people. The invitation bar cannot be null and the content cannot contain space characters.

6. The person in rooms can send an invitation to the online staff in the lobby at any time, and the invitee will receive a pop-up reminder.

7. Players who receive the invitation can choose to refuse the invitation and the inviter will receive the rejected information. If the player accepts the invitation, he will enter the inviter's room.

8. When the number of people in the room is more than 2, any user in the room can choose to start the game.

9. If a user disconnects from the server, all other users will update their online staff list. If the user is in a room, others in the room will update the list of users in their room.

10. If the number of people in the room is less than 1, the room will be cancelled and all the users in the lobby will update their room list.

11. If the server is disconnected from the user, a pop-up prompt appears and the client will be closed.

12. Once a new user connects to the server, all non-gaming users will update their online staff list.

13. Whenever a new user enters the room, everyone in the room gets a list of people in the room.

14. If the invited user does not exist in the system, he will receive a reminder from the system. If the invited user is not in the lobby (in their own or other

user's room), the invitation cannot be sent and the inviter receives the prompt.

- Game

1. After all users enter the game, the system will randomly select a user’s turn to start game.

2. If a user enters his turn, he can choose to skip his turn and the game enters the next player's turn.

3. The first user can place a letter on the board. Otherwise the user can only place the letter next to the grid where the letter has been placed.

4. After successfully placing the letters, the user can choose whether to initiate a poll. If he gives up the poll, he will go directly to the next user's turn.

5. If the user chooses the poll, other users in this game will receive two prompt dialogs, one asking whether the string consisting of horizontal letters is a word, and the other asking whether the string consisting of vertical letters is a word. If one of the other users thinks that a horizontal or vertical string is not a word, the system determines that the horizontal or vertical string is not a word, for which the system does not add points to the user who initiated the poll. If all other users think that a horizontal or vertical string is a word, then for the string, the system will add points to the user who initiated the poll, the score is equal to the length of the word (general case), and if the voting rights are abandoned(Click on the top right “X” key), it will be considered as a veto.

6. After the poll, the game enters the next user's turn.

7. If a user quits in the game (you can use quit button or the "X" button in the upper right corner), players in this game will receive a pop-up prompt. If

the number of players left in the game is greater than or equal to 2, the game continues. If it is less than 2, the game will be terminated and the room will be cancelled. The users in all the hall will be updated the online room information. Users who quit the game will be returned to lobby to get a list of online people and a list of online rooms. The last gamer will be forced to quit the game.

8. If all the users in a game pass their own rounds or all the grids in the board are full, the game results will be obtained and the game ends. The corresponding room is cancelled. All the players in the game will return to lobby to get a list of online people and online list of rooms.

9. If the server is disconnected from the client(game state), a pop-up prompt will appear and the client will be closed.