University of California,
Santa Barbara

CS48 Spring'17, G01 Project

Arcade Haven
Final Report

Alnawakhtha, Yusuf
Binkley, Jeremy Joseph
Cardenas, Jose Homero Jr
Li, Yuanqi
Qian, Zhancheng
Zhang, Fangjun

Supervised by
Tsai, Yantsey
Professor Costanzo, C. Michael

1 Project description

Arcade Haven will provide online arcade games with a user interface that allows for online discussion, online purchases, matchmaking lobby, and a user authentication feature. The arcade games will have a multi-player feature that allows two players to compete by either preventing the other player from finishing the objective of the game or score more points before the end of the game. The online purchases will consist of finite consumption power-ups that allow a player to gain an advantage over their opponent or a difficult game. The matchmaking

2 Vision

2.1 Problem statement

There are two main problems we are going to solve:

- Design a front-end arcade game emulator that has modern features (i.e., networking, multi-player, unified game platform).

  To solve this problem, our front-end group is going to implement a simple PACMAN game first to demonstrate system features.

- Design a robust and secure back-end server that supports multi-player, communication, and synchronization.

  We have another group of three working on the server. The server will have a database used to store player information. It is able to listen to user activities (i.e., user login) and make responses. It also uses non-blocking I/O to respond to all users. Public/private keys are used to ensure security.

2.2 High-level goals

- This system provides on-line arcade gaming platform for nostalgic gamers. For the purpose of demonstration, we include the PACMAN game.
- This system dedicates in making old games multi-player-playable, and it transforms old games from single-player mode into multi-player mode.
- This system facilitates communication between various players, by providing with chat box and also features like adding friends.

2.3 User-level goals

- This system provides certain players with opportunity of making friends in this vast virtual network.
- This system provides certain players with opportunity of getting hands-on experience with their favorite childhood games.

2.4 System features

- Allow users to create game accounts and securely log in through back-end server. And check user activity log, manage password, manage payment methods.
- Allow on-line instant messaging communication between players. This includes player to player private-type chat, also player to public broadcast-type chat.
- Present various games for users to choose from. Most of the games will be arcade games.
- Incorporate multi-player collaboration and competition with classical games, in this project's example, PACMAN.
- All connections are encrypted in SSL/TLS to ensure security and privacy, and make sure that the platform is not susceptible to Internet hacks/attacks.
- The Back-end server and database will be hosted on Amazon or other PaaS, to ensure best performance and robust of the whole game platform.

2.5 Other requirements and constraints

Enhanced, with details that might appear in the "Supplementary Specification" of a more elaborate project

- The game gallery needs to be highly scalable so that it will always be able to adopt more arcade games in the future, if needed.
- The database needs to be highly scalable so that it will be able to cope with a large amount of users.
- The game platform needs to go through high-capacity stress-testing, in order to check if the server could handle large streams of user activity.
- The test of the prototype should include at least 100 registered users as test members.
- There needs to be a single web-page that serves as an introduction to the game platform, and also serves as a place to download the package/user registration/check users activity.

3 Use cases

3.1 Use case 1: Login

3.1.1 Primary actor

Player: want to login the game.

3.1.2 Main success scenario

1. The player tries to login the game by typing in his account name and password.
2. Platform client retrieves information to the login server.
3. The login server checks account information in database.
4. The login server finds the account name and password matched.
5. The login server makes login record and send login success message to client.
6. Client prompt login success message and entering gaming interface.

3.1.3 Extensions

2a. No Internet connection.

    1. Client tries to connect login server and after 60 seconds without response, client prompt no connection information.

4a. Account name and password mismatched.

    1. The login server sends the client with authentication failure message.
    2. Client receives message and prompt login fail message onto screen.

3.2 Use case 2: Update user highest score

3.2.1 Primary actor

Player

3.2.2 Main success scenario

1. The player finishes his/her game, and the score is computed.
2. Platform client sends a update score message to server.
3. The game server receives user's score and finds out it is a new high score.
4. The game server updates the player's score.

3.2.3 Extensions

3a. The game server receives user's score and finds out it is not a new high score.

4a. The game server doesn't update user's score.

3.3 Use case 3: Play game

3.3.1 Primary actor

Player

3.3.2 Main success scenario

1. Player successfully logs into their account.
2. System opens up chat box and game screen onto Player's screen.
3. Player begins game.
4. Player collects all points on screen without dying three times.
5. System updates high score database.
6. Player resets game and plays again.

3.3.3 Extensions

1a. Player fails login.

   1. Server denies account access.
   2. Player attempts login again.

4a. Player dies three times

   1. Player resets game and plays again

4b. Player pauses game

   1. Player pauses game.
   2. System stops game and timer while the game is paused.
   3. Player starts game.
   4. System starts game and timer and everything in game continues.

6a. Player selects next map

   1. Player chooses to move to next map.
   2. System generates new map for the Player.
   3. Player begins to play on new map.

3.4 Use case 4: Communicate with others

3.4.1 Primary actor

Message sender, message recipient

3.4.2 Main success scenario

1. The message sender inputs a message in his/her client and clicks 'send' button.
2. The servers receives a message from that player, and sends the message to the message's recipient.
3. The message recipient receives a message from the server, displays that message on his/her client.

3.4.3 Extensions

2. The server receives a message from that player, but finds out its recipient is not online. The server will discard the message, and sends a system message back to the sender telling him/her the recipient is offline.

4 Project schedule

Time	**Target**
Week 3	Finish authorization and IM server Finish game logic
Week 4	Refine database schema and support message history Finish game engine
Week 5	Finish client-server message transmission Finish game controller
Week 6	Alpha preview Synchronize client and server Finish game art design
Week 7	Design in-game items
Week 8	Implement items shop and Item class in game
Week 9	Beta preview Support item purchase
Week 10	Necessary modification to the game based on TA's and professor's feedback

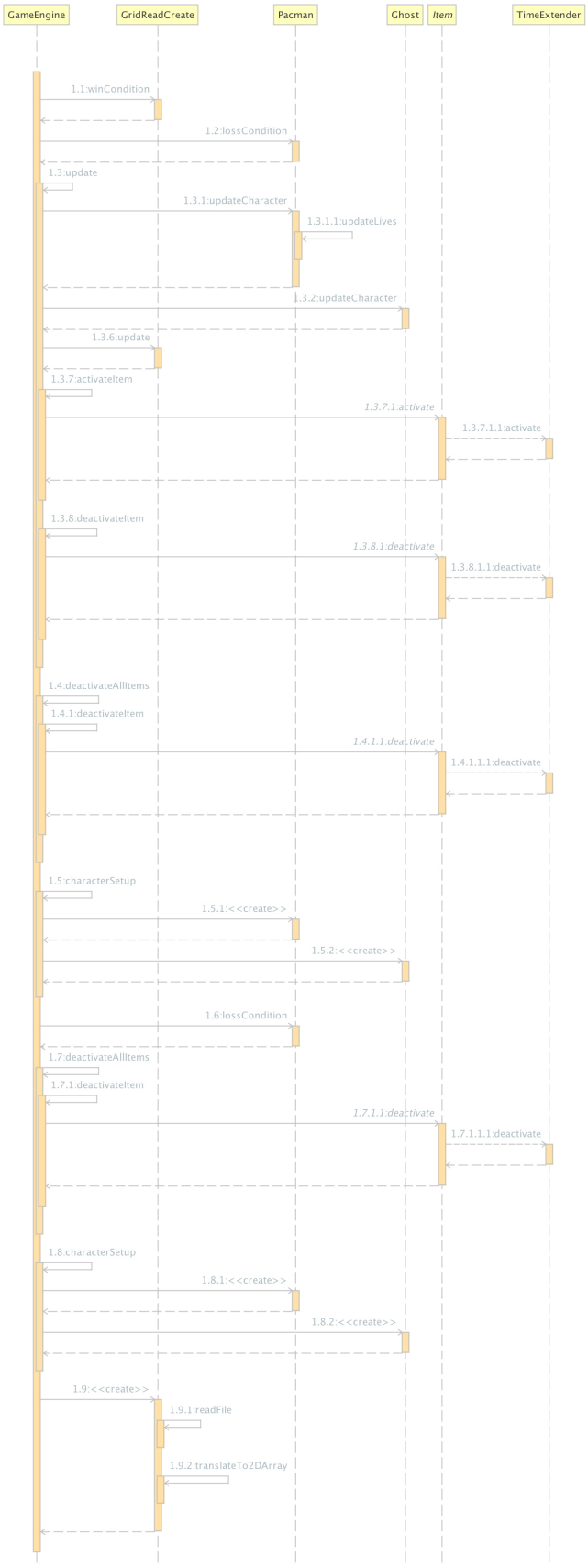

Figure 1. Sequence diagram: Game engine sequence.

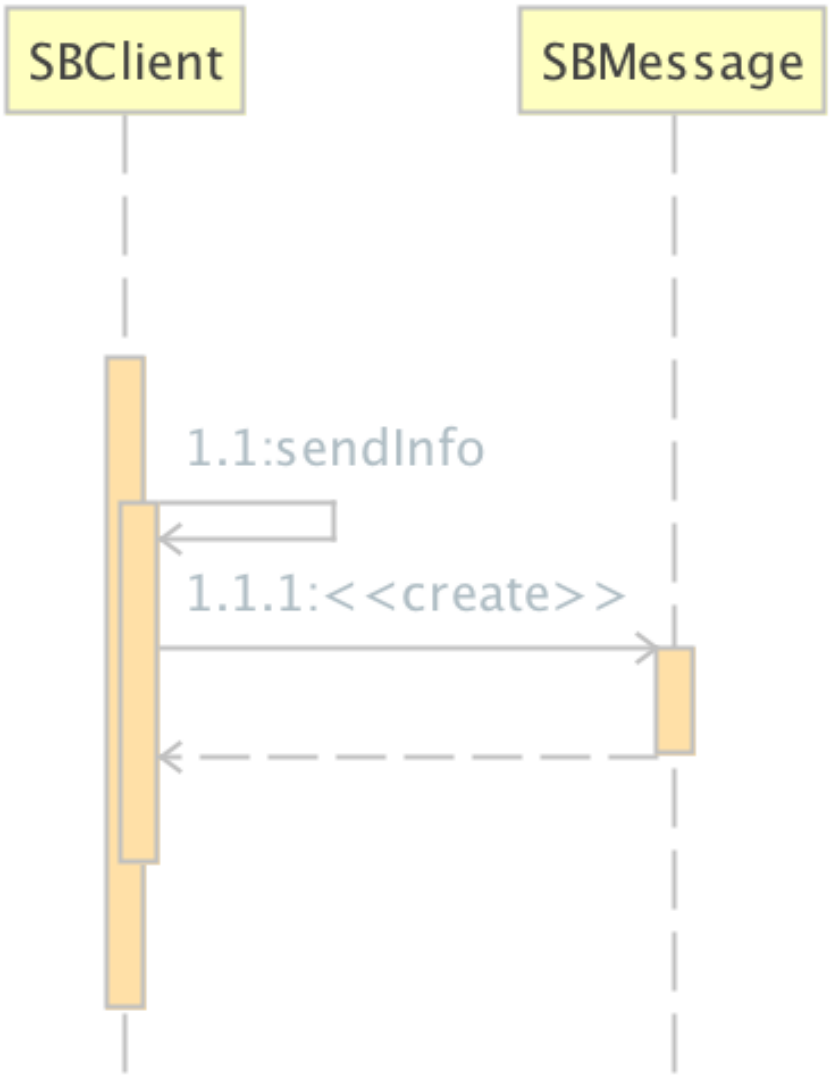

Figure 2. Sequence diagram: Client connects to server.

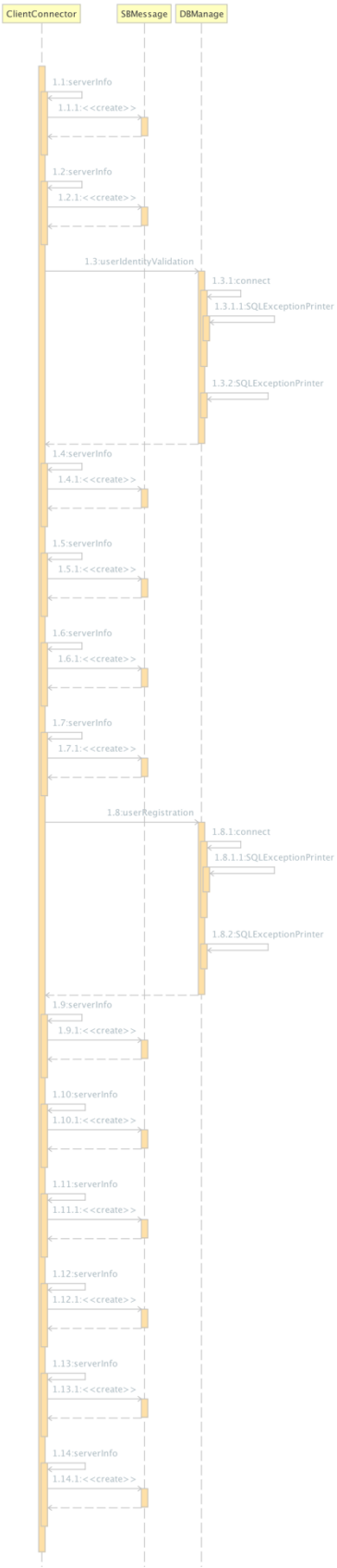

Figure 3. Sequence diagram: Server responses to a client.

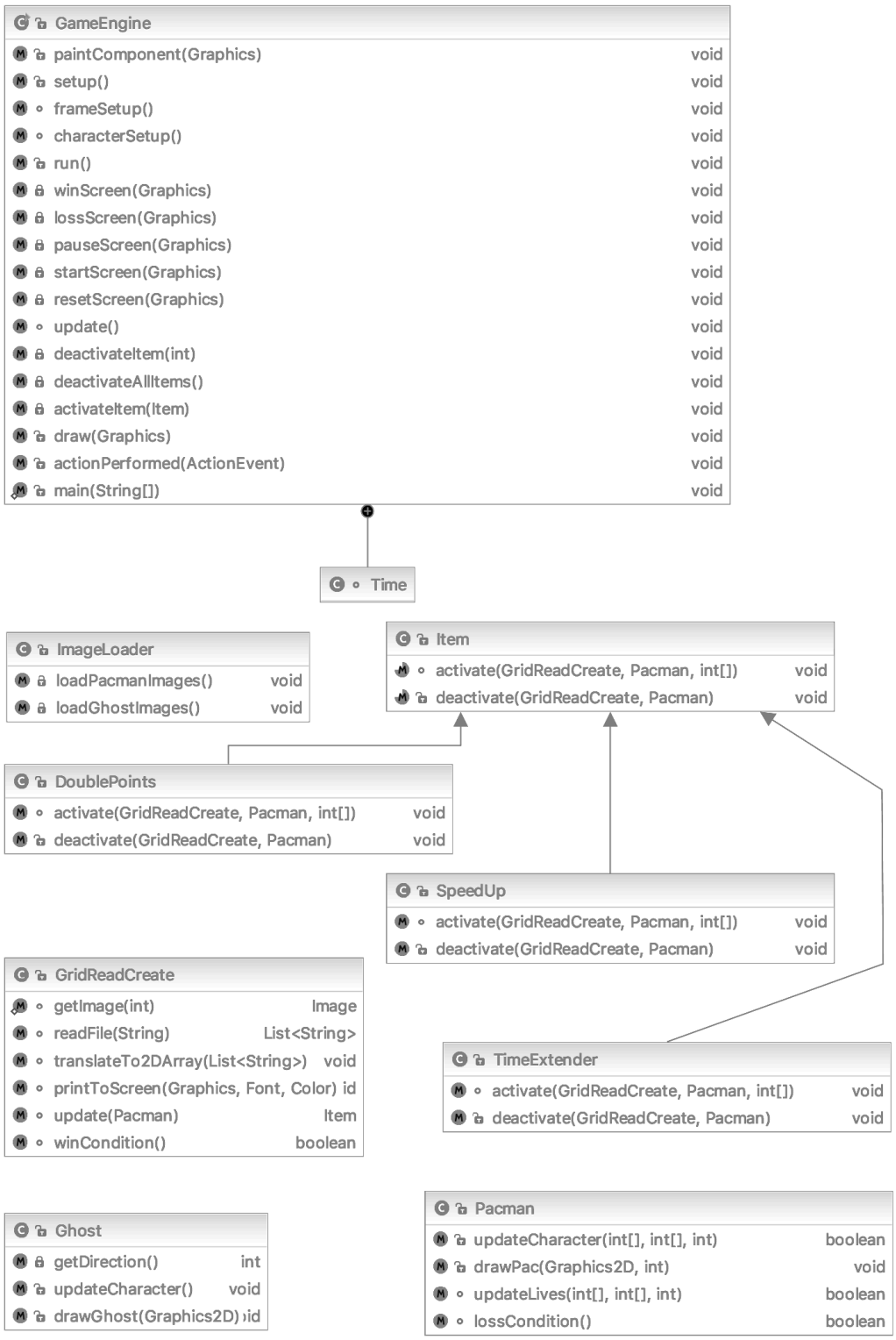

Figure 4. Static class diagram: PACMAN package.

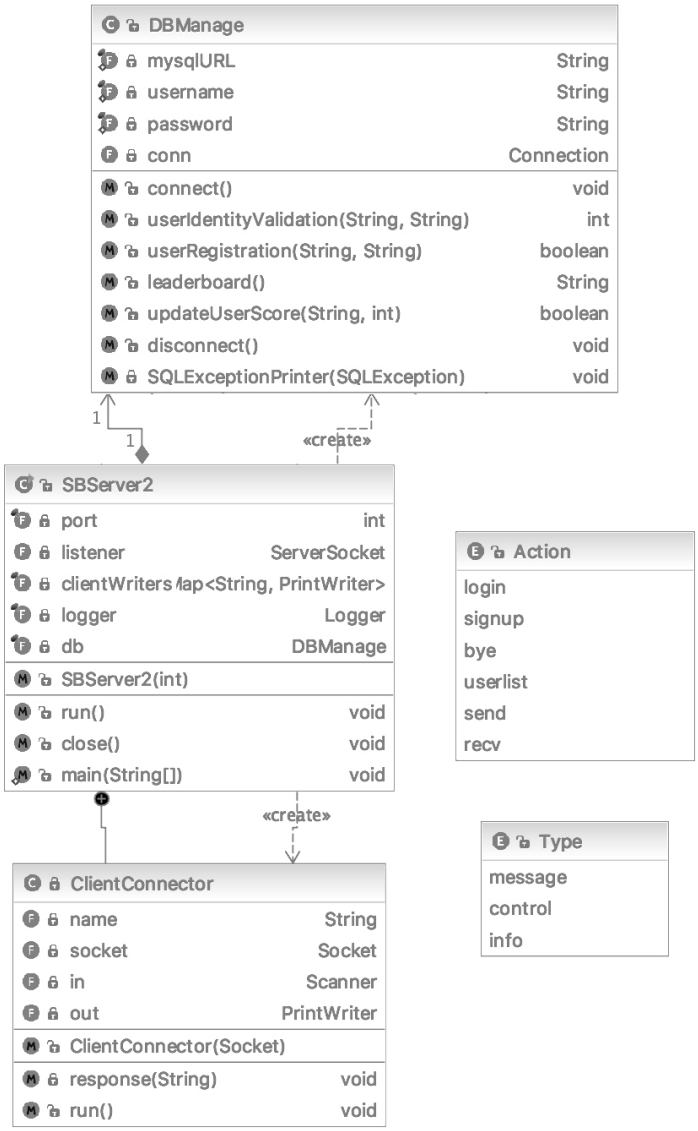

Figure 5. Static class diagram: networking package.

Figure 6. Static class diagram: GUI package.

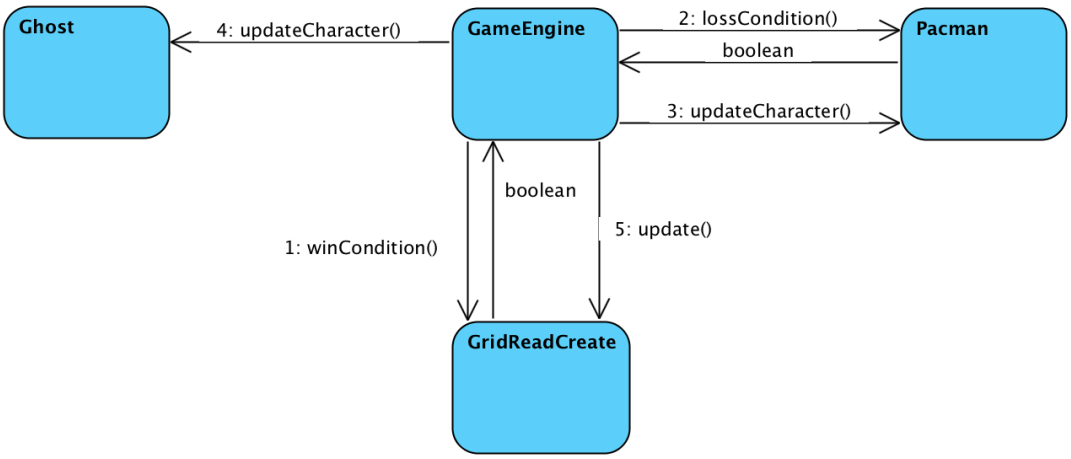

Figure 7. Interaction diagram: Game engine diagram.

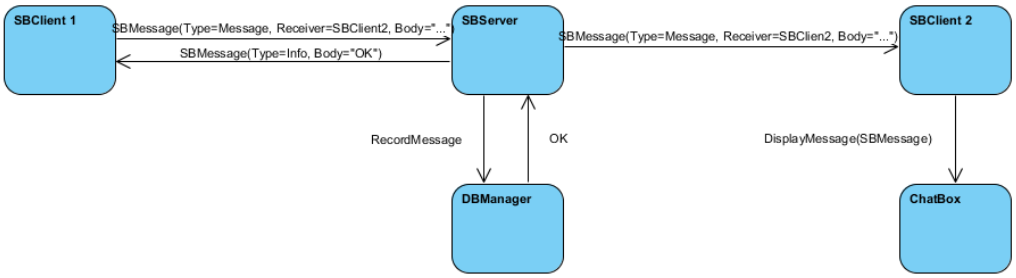

Figure 8. Interaction diagram: Client communication diagram.

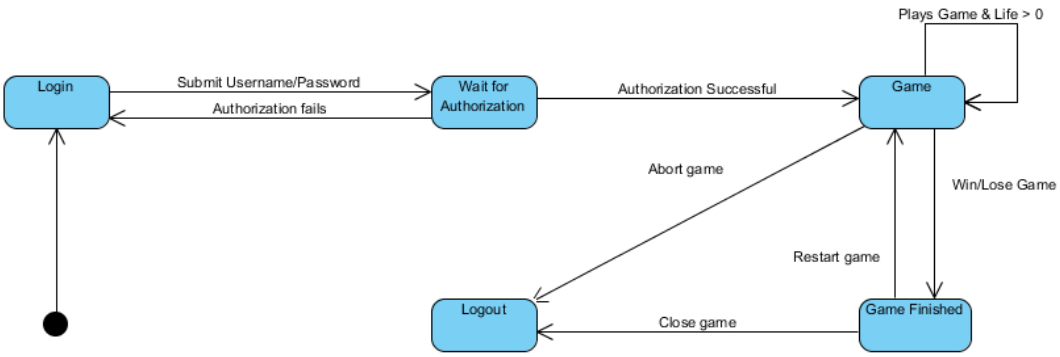

Figure 9. State diagram: Game play state diagram.