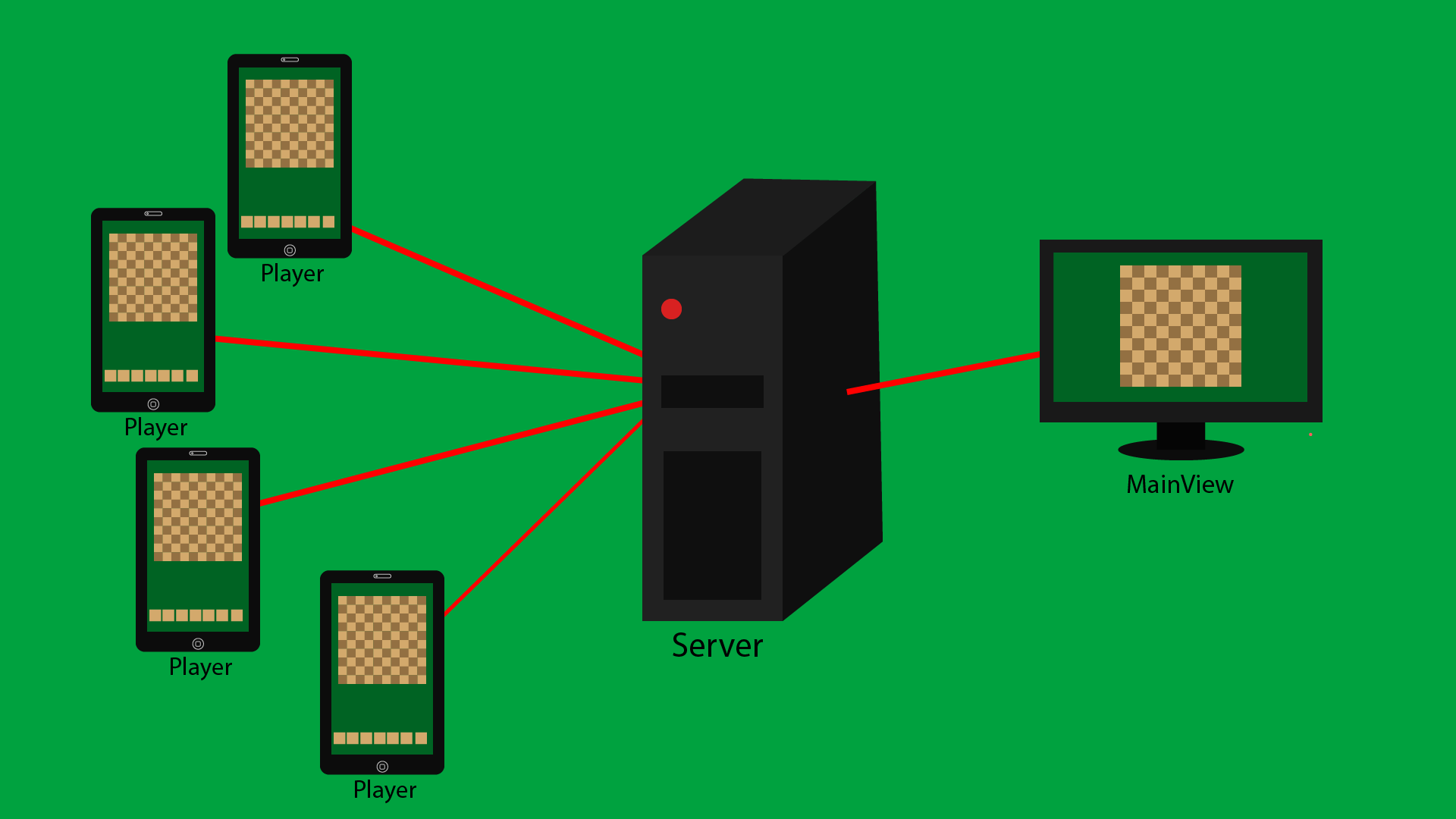
# DCU School of Computing  Assignment Submission

|  |  |
| --- | --- |
| **Student Name(s):** | **Aidan Fogarty, Eoghan Murphy, Michael Dowling, Oishin Smith, Danie Perecz, Martin Zaloudek** |
| **Student Number(s):** | **16413106** |
| **Programme:** | **BSc in Computer Applications** |
| **Project Title:** | **CA314 Assignment 1 - Analysis** |
| **Module code:** | **CA314** |
| **Lecturer:** | **Renaat Verbruggen** |
| **Project Due Date:** | **26/10/2018** |

|  |
| --- |
| **Declaration**  **I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion, or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references.**  **I have not copied or paraphrased an extract of any length from any source without identifying the source and using quotation marks as appropriate. Any images, audio recordings, video or other materials have likewise been originated and produced by me or are fully acknowledged and identified.**  **This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. I have read and understood the referencing guidelines found at**[**http://www.library.dcu.ie/citing&refguide08.pdf**](https://websvc.dcu.ie/%22http://www.library.dcu.ie/citing&refguide08.pdf/%22)**and/or recommended in the assignment guidelines.**  **I understand that I may be required to discuss with the module lecturer/s the contents of this submission.** |

# **Preface**

Since we are taking a novel approach at recreating a multiplayer Scrabble game, we feel that a short explanation is necessary. The game will (in most cases) be played in the same room. A lobby will be started on a central screen. Players will connect with their phones (or other web capable devices) to this lobby. Players can then make their moves from their devices and everything that happens will be updated on the larger screen (allowing for spectators and a more traditional feel).



An illustration of how the game will work.

## Refined requirements specification

## *Purpose:*

The goal of this project is create a networked version of Scrabble called “MyScrabble”. Upon successful completion of this project, an exciting game which keeps users engaged will be produced.

## *Intended Audience and Reading Suggestions:*

## This document is intended for the project team, developers and domain experts (The Lecturer). All members involved in the project should be familiar with this document.

## *Project Scope:*

* “MyScrabble” will be a multiplayer version of the board game Scrabble.
* Players will be able to challenge other players.
* Players will be able to challenge a computer player.
* Games will be able to support 2-4 players.
* Players will be able to create a game lobby.
* Players will be able to join a game lobby.
* The player will be able to connect to the game using a web URL.
* Players will be able to play the game on either a phone or a laptop/PC.

## *Assumptions:*

* The server will handle the game logic and communication between players.
* Players will be able to enter a lobby code to join a game.

## *Ambiguities:*

* There was some ambiguity around how networked the game had to be. Using the phone (Client) to connect to a host (Server) was found to be sufficient upon talking to the domain expert (Lecturer).

**Scenarios**

The following scenarios (2 - 8) were recorded when the group met up after lectures to physically play a game of Scrabble.

# ***Begin Game:***

* **Current System State:**

The main menu is visible on the main display screen, nothing has been selected. Each player has their phone with them.

* **Scenario:**

Mikey chooses the option “Play Game” on the main display, he then joins a lobby with his phone and waits to start the game.

* **Next Scenario:**

Game is ready to begin. “Choose Playing Order” scenario is started.

# ***Choose Playing Order:***

* **Current System State:**

All the letters jumbled in the letter-bag.

* **Scenario:**

Each player takes one letter to determine playing order. The player with the closest letter to “A”, goes first. Aidan chose the letter C, this was the lowest, he goes first. After completion, everyone puts their letter back into the bag.

* **Next Scenario:**

Playing order is chosen. “Setup Game” scenario is started.

# ***Setup Game:***

* **Current System State:**

Board is empty, letter-bag full.

* **Scenario:**

Aidan chooses 7 letters at random from the bag, this continues through the playing order previously decided.

* **Next Scenario:**

Players have their letters and are ready for the first turn. “Player Makes Turn” scenario is started.

# ***Player Makes Turn:***

* **Current System State:**

There are two words on the board (jogs and moist)

* **Scenario:**

Mikey placed [C E A E] on the board to make CEASE. Words were validated after tiles were placed. Points were totalled up for Mikey. Mikey draws 4 letters from bag to replace the ones he played.

* **Next Scenario:**

Three words are on the board (jogs, moist and cease). The scenario could lead to many other scenarios such as “Player Makes Turn” again or “Skip Turn”.

# ***Player Makes Wrong Turn:***

* **Current System State:**

Several words are on the board

* **Scenario:**

Oishin intended to play the word TOLL but misspelled it TOLE. The letters that he played were shown to other players. His letters were given back to him.

* **Next Scenario:**

Board is in same state, Oishin goes again. “Player Makes Turn” scenario is started.

# ***Skip Turn:***

* **Current System State:**

Words may potentially be on the board. Each player has their letters.

* **Scenario:**

Oishin could not think of a word to place on the board. Instead he decided to trade in all of his letters in place of a turn.

* **Next Scenario:**

Oishin now has different set of letters on his stand. This scenario could lead to “Player Makes Turn” for the next player.

# ***End game:***

* **Current System State:**

All tiles used and Aidan is out of tiles

* **Scenario:**

Everyone’s points are added up. Reduce points of unused tiles from each player.

One player used all their tiles so their score re is increased by sum of unused tiles. A winner is chosen (whoever has most points).

* **Next Scenario:**

Game displays who won and each player’s scores, final board is displayed. This could lead to the scenario “Begin Game” to start a new game.

# ***Leave game:***

* **Current System State:**

Board may have multiple words.

* **Scenario:**

Eoghan needed to leave. He put his words back into the bag. Then left the game.

* **Next Scenario:**

Eoghan no longer in game. More letters in the bag. This will lead to the scenario “Player Makes Turn” for the next player.

**Primary Class List**

1. Player
2. Lobby
3. Piece
4. Board
5. Server
6. Client

**Use Cases**

|  |  |  |
| --- | --- | --- |
| **USE CASE 1** | Begin game. | |
| **Goal in Context** | Start a game of scrabble. | |
| **Scope & Level** | Company, Summary. | |
| **Preconditions** | MainView showing menu.  Players ready to join. | |
| **Success End Condition** | Players join the game. | |
| **Failed End Condition** | Player unable to join game. | |
| **Primary,**  **Secondary Actors** | Player  Central Screen, Server | |
| **Trigger** | “Start Game” is clicked on MainView | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Player connects to website. |
|  | 2 | Enters game code. |
|  | 3 | Select play game. |
|  | 4 | Join lobby. |
|  | 5 | Wait for players to join to start game. |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 4a | Lobby is full, player unable to join. |
|  | 5a | Not enough players to start game |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 1. Begin game. |
| **Priority:** | Medium |
| **Performance** | Less than one minute |
| **Frequency** | Once per game |
| **Channels to actors** | To be decided. |
| **OPEN ISSUES** |  |
| **Due Date** | Release 2.0 |
| **...any other**  **management**  **information...** |  |
| **Superordinates** |  |
| **Subordinates** |  |

|  |  |  |
| --- | --- | --- |
| **USE CASE 2** | Choose Playing Order | |
| **Goal in Context** | Players decide the playing order so the game may start. | |
| **Scope & Level** | Primary Task | |
| **Preconditions** | Letter tiles are jumbled in the letter bag. | |
| **Success End Condition** | The game is ready to be started. | |
| **Failed End Condition** | Game cannot start. | |
| **Primary,**  **Secondary Actors** | Server  Player, Central Screen | |
| **Trigger** | Begin game (use case 1). | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Players take one letter each from letter bag. |
|  | 2 | Server checks each player’s letter. |
|  | 3 | Player with the closest letter to A is chosen as the player to make first move (blank tile is of higher order). |
|  | 4 | Letters are returned to the letter bag (an attribute of board). |
|  | 5 | Letter bag is shuffled. |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 3a | Two or more players have the same current highest order letter.   * Repeat Choosing Playing Order (use case 2) |
| **VARIATIONS** |  | **Branching Action** |
|  |  | - |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 2. Choose Playing Order |
| **Priority:** | High |
| **Performance** | 15 seconds |
| **Frequency** | 1 time per game |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | None |
| **Due Date** | Pre 1.0 |
| **...any other**  **management**  **information...** | - |
| **Superordinate’s** | Begin Game (use case 1) |
| **Subordinates** | Make Turn (use case 4) |

|  |  |  |
| --- | --- | --- |
| **USE CASE 3** | Setup Game | |
| **Goal in Context** | 7 letters are given at random to each player. | |
| **Scope & Level** | Company, Summary | |
| **Preconditions** | Board is empty, players are waiting for their initial letters. | |
| **Success End Condition** | Each player receives 7 randomly chosen letters. | |
| **Failed End Condition** | Player/Players do not receive 7 letters. | |
| **Primary,**  **Secondary Actors** | Server  Player, Central | |
| **Trigger** | All players are in ready state to receive 7 random letters after playing order is chosen. (Use case 2) | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Game selects 7 random (this can be vary) letters for each player. |
|  | 2 | Player receives letters. |
|  | 3 | Player examines letters. |
|  | 4 | Players may rearrange their letters on their stand. |
|  | 5 | Player get ready for turns (Use case 4) |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 4a | Player does not rearrange their letters  4a1. Letters stay the same on their stand. |
| **VARIATIONS** |  | **Branching Action** |
|  |  |  |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 4. Setup Game |
| **Priority:** | top |
| **Performance** | < 1 minute |
| **Frequency** | Once per game |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** |  |
| **Due Date** | Release 1.0 |
| **...any other**  **management**  **information...** |  |
| **Superordinate’s** |  |
| **Subordinates** | All cases from 4-8 |

|  |  |  |
| --- | --- | --- |
| **USE CASE 4** | Make Turn | |
| **Goal in Context** | Player makes turn to advance the game and tries to win. | |
| **Scope & Level** | Company, Summary | |
| **Preconditions** | The game board is set up and the game is ready to be played | |
| **Success End Condition** | Player makes turn, gains points, the board returns to a ready state. | |
| **Failed End Condition** | Player does not take a turn in any form. | |
| **Primary,**  **Secondary Actors** | Player  Server, Central Screen | |
| **Trigger** | The Server works out playing order (use case 2) and it is a players turn. | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Player examines state of board |
|  | 2 | Player places letters adjacent to existing tiles to form a word. |
|  | 3 | Game validates word(s) |
|  | 4 | Player is awarded points. |
|  | 5 | Player is provided replacement letters from letter bag |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 2a | Player cannot form word  2a1. Player Skips Turn (use case 6) |
|  | 3a | Server cannot validate word  3a1. Player Makes Wrong Turn (use case 5) |
|  | 5a | No letters left in letter bag  5a1. End Game (use case 8) |
| **VARIATIONS** |  | **Branching Action** |
|  | 2 | Player may  place valid word  place invalid word  skip turn |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 4. Make Turn |
| **Priority:** | top |
| **Performance** | 1 minute |
| **Frequency** | 15-20 per player |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | What if word is valid but is not in our dictionary? |
| **Due Date** | Release 1.0 |
| **Superordinate’s** |  |
| **Subordinates** | Player Skips Turn (use case 6)  Player Makes Wrong Turn (use case 5)  End Game (use case 8) |

|  |  |  |
| --- | --- | --- |
| **USE CASE 5** | Make Wrong Turn | |
| **Goal in Context** | The system handles the Player making a wrong turn. | |
| **Scope & Level** | Company, Summary | |
| **Preconditions** | The game board is set up and the game is ready to be played | |
| **Success End Condition** | Player gets another chance to make a turn. | |
| **Failed End Condition** | Player is not alerted of making a wrong turn. | |
| **Primary,**  **Secondary Actors** | Player  Server, Central Screen | |
| **Trigger** | The Player makes a turn (use case 4). | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Player examines state of board |
|  | 2 | Player places tiles on game board |
|  | 3 | Game cannot validate word(s) |
|  | 4 | Player is notified |
|  | 5 | Player takes back tiles and tries again (use case 4). |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 5a | Player cannot form word  5a1. Player Skips Turn (use case 6) |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 5. Make Wrong Turn |
| **Priority:** | top |
| **Performance** | 1 minute |
| **Frequency** | Depends on player skill level |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | What if word is valid but is not in our dictionary? |
| **Due Date** | Release 1.0 |
| **Superordinate’s** | Player Makes Turn (use case 4) |
| **Subordinates** | Player Skips Turn (use case 6)  Player Makes Turn (use case 4) |

|  |  |  |
| --- | --- | --- |
| **Use Case 6** | Skip Turn | |
| **Goal in Context** | Player Skips turn | |
| **Scope & Level** | Company, Summary | |
| **Precondition 1** | Player skips turn because they are unable to place letters on the game board. | |
| **Success End Condition** | Player receives new random letters. Or Player keeps current letters, doesn’t make a move and skips turn. | |
| **Failed End condition** | Player doesn't make it clear whether they wish to keep or swap letters before skipping turn. | |
| **Primary,**  **Secondary**  **Actors** | Player  Server, Central Screen | |
| **Trigger** | The Server gives the Player a choice of two options. | |
| **Description** | **Step** | **Action** |
|  | 1 | Player examines state of board |
|  | 2 | Player decides to skip turn |
|  | 3 | Player puts their current letters into the letter bag and randomly chooses seven new ones. |
| **Extension 1** | **Step** | **Branching Action** |
|  | 1 | Player examines state of board |
|  | 2 | Player decides to skip turn |
|  | 3 | Player must wait until their next turn to play again |
| **Extension 2** | **Step** | **Branching Action** |
|  | 3a | All players have skipped their turn twice  3a1. The game is ended (use case 8) |
| **Variations** |  | **Branching Action** |
|  |  | Player may:  skip turn  skip turn, swap letters for new ones from letter bag  skip turn twice to end game |

|  |  |
| --- | --- |
| **Related Information** | Skip Turn |
| **Priority** | Top |
| **Performance** | 1 minute |
| **Frequency** | Depends on player. |
| **Channels to actors** | Not yet Determined |
| **Open Issues** | None |
| **Due Date** | Release 1.0 |
| **Superordinate’s** |  |
| **Subordinates** | End Game (Use case 8) |

|  |  |  |
| --- | --- | --- |
| **Use Case 7** | Leave Game | |
| **Goal in Context** | The system will allow a player to leave the game. | |
| **Scope & Level** | Company, Summary | |
| **Preconditions** | The player decides that they wish to leave the game | |
| **Success End Condition** | Player leaves game and the letters they have are returned to the letter bag. | |
| **Failed End condition** | Player physically leaves but they still appear in the game / their letter pieces are still out of the letter bag | |
| **Primary,**  **Secondary**  **Actors** | Player  Server, Central Screen | |
| **Trigger** | The Player wants to leave the game. | |
| **Description** | **Step** | **Action** |
|  | 1 | Player presses leave button. |
|  | 2 | Player then puts letters back into the letter bag. |
|  | 3 | Player then leaves game |

|  |  |
| --- | --- |
| **Related information** | **7. Leave game** |
| **Priority** | Top |
| **Performance** | 1 minute |
| **frequency** | Depends on player |
| **Channels to actors** | Not yet determined |
| **Open Issues** | If all players leave before any player Makes Turn (use case 4), will the last remaining player win by default? |
| **Due Date** | Release 1.0 |
| **Any other management information** |  |
| **Superordinate’s** |  |
| **Subordinates** |  |

|  |  |  |
| --- | --- | --- |
| **USE CASE 8** | End Game | |
| **Goal in Context** | Finish the game and show player results. | |
| **Scope & Level** | Primary Task | |
| **Preconditions** | Last player move has been made, playing time is over. | |
| **Success End Condition** | Player scores displayed, game is over. | |
| **Failed End Condition** | Game is prematurely over. | |
| **Primary,**  **Secondary Actors** | Player  Central Screen, Server | |
| **Trigger** | One of:   1. Player used all tiles in an accepted move and the letter bag is empty. 2. Six successive scores of zero occur, obtained from passes, successful challenges or tile exchanges. 3. If using a time clock: A digital clock is not in overtime until –0:01 is shown. | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | One of the triggers occurs. |
|  | 2 | Game calculates all players’ scores. |
|  | 3 | Player scores are displayed. |
|  | 4 | Winner announced. |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 4a | Players with equal scores.   * Game is a tie. |
| **VARIATIONS** |  | **Branching Action** |
|  |  |  |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 8. End Game |
| **Priority:** | High |
| **Performance** | 30 seconds |
| **Frequency** | 1 time per game |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | None |
| **Due Date** | Pre 1.0 |
| **Superordinate’s** |  |
| **Subordinates** |  |

|  |  |  |
| --- | --- | --- |
| **USE CASE 9** | Update Display | |
| **Goal in Context** | Update client screens when new information is provided. | |
| **Scope & Level** | Primary Task | |
| **Preconditions** | Any game state. | |
| **Success End Condition** | Client screens are updated correctly. | |
| **Failed End Condition** | Wrong Information is sent to client displays. | |
| **Primary,**  **Secondary Actors** | Server  Central Screen, Player | |
| **Trigger** | Any change in game state (e.g. score is changed) | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | The server sends out information to all clients. |
|  | 2 | Clients screens are updated to show this information. |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 1a | Client disconnects temporarily.   * Information is kept in a cache. |
| **VARIATIONS** |  | **Branching Action** |
|  |  |  |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 9. Update Display |
| **Priority:** | High |
| **Performance** | <1 second |
| **Frequency** | 1000s of time |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | None |
| **Due Date** | Pre 1.0 |
| **Superordinate’s** |  |
| **Subordinates** |  |

|  |  |  |
| --- | --- | --- |
| **USE CASE 10** | Verify Move | |
| **Goal in Context** | Whenever there is a change to the boards state, words on the board have to be validated to ensure its a real word. | |
| **Scope & Level** | Primary Task | |
| **Preconditions** | Board can be in any state. | |
| **Success End Condition** | Scrabble board successfully validates whether the words on the board are correct or not. | |
| **Failed End Condition** | Scrabble fails to be validated. | |
| **Primary,**  **Secondary Actors** | Server  Player, Central Screen | |
| **Trigger** | One of:   1. A move is made by a player | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | One of the triggers occurs. |
|  | 2 | Server validates the word. |
|  | 3 | Server validates all other words on board. |
|  | 4 | Update display is called and displays new word on board. (Use Case 9) |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 4a | Word is incorrect.   * failure message is displayed. |
| **VARIATIONS** |  | **Branching Action** |
|  |  |  |

|  |  |
| --- | --- |
| **RELATED INFORMATION** | 10. Verify Move |
| **Priority:** | High |
| **Performance** | <1 seconds |
| **Frequency** | 50+ times per game |
| **Channels to actors** | Not yet determined |
| **OPEN ISSUES** | None |
| **Due Date** | Pre 1.0 |
| **Superordinate’s** |  |
| **Subordinates** |  |

**Class Responsibility Collaborator**

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Player | ID:  1 | | Type:  Concrete |
| Description:  A Player will be responsible for placing pieces on the board with the intention of gaining a score and winning. | | | Associated Use Cases:  1,3,4,5,6,7, 8 & 9 |
| Responsibilities | | Collaborators | |
| Each Player would keep their own separate score | | Game | |
| Can take a turn | | Piece, Board, Game | |
| Attributes | | | |
| Name | | | |
| Total Score | | | |
| Letters | | | |
| Relationships | | | |
| Owns (One to Many) | | Piece | |
| Inherits | | Client | |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Game | ID:  2 | | Type:  Concrete |
| Description:  The Game class controls each scrabble game that can be played on server. | | | Associated Use Cases:  1, 3,8 & 10 |
| Responsibilities | | Collaborators | |
| Game controls the board. | | Board | |
| Game controls the players. | | Player | |
| Handles joining and exiting of players. | |  | |
| Handles the main board view | | MainView | |
| Attributes | | | |
| ID (Primary Key) | | | |
| Game Join code. | | | |
| Game Owner | | | |
| Max-Slots | | | |
| Players joined | | | |
| Relationships | | | |
| Aggregation (has-parts) | | Client, Board | |
| Other Associations | | Server | |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Piece | ID:  3 | | Type:  Concrete |
| Description:  An object to be placed by players on the board to score points | | | Associated Use Cases:  3,4 & 5 |
| Responsibilities | | Collaborators | |
| Provide Information about itself (Letter and score) | |  | |
| Be placed on board | | Player | |
| Attributes | | | |
| Character | | | |
| Score | | | |
| Relationships | | | |
| Owned By (Many to One) | | Player, Board | |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Board | ID:  4 | | Type:  Concrete |
| Description:  The board necessary to play the game. Contains a 15 x 15 grid of squares, each of which accommodates a single letter tile. There are a few premium squares, which multiply the points awarded. | | | Associated Use Cases:  1, 2, 3, 4 & 5 |
| Responsibilities | | Collaborators | |
| Provides the game environment | | Server, Client | |
| Awards points | | Piece, Player | |
| Distributes playing pieces | | Piece, Player | |
| Keeps track of pieces in ‘letter bag’ | |  | |
| Proclaims winner | | Player, Client | |
| Attributes | | | |
| Space List (Occupied Spaces) | | | |
| Space List (Premium Spaces) | | | |
| Player Count | | | |
| Relationships | | | |
| Aggregation (has-parts) | | Piece | |

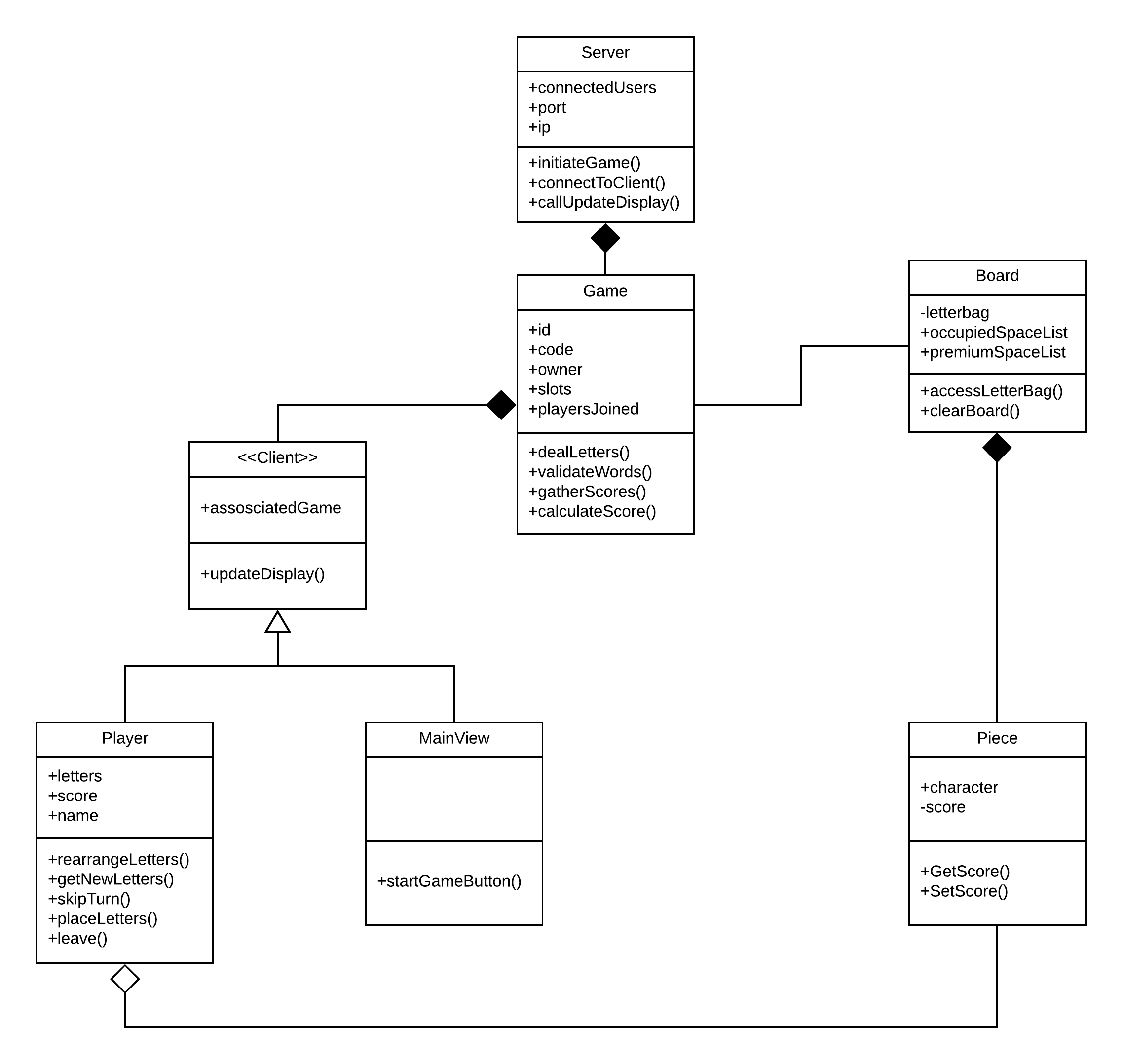
|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Server | ID:  5 | | Type:  Concrete |
| Description:  The mechanism passing information between clients and the respective game instance. | | | Associated Use Cases:  Almost All |
| Responsibilities | | Collaborators | |
| Notify of Changes | | Client | |
| Attributes | | | |
| List of games running | | | |
| Relationships | | | |
| Aggregation (has-parts) | | Game | |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  Client | ID:  6 | | Type:  Concrete |
| Description:  The object responsible to provide a view of the board to the Players and of all the changes in the game. | | | Associated Use Cases:  Almost All |
| Responsibilities | | Collaborators | |
| Update View | | Server | |
| Attributes | | | |
| Associated Game | | | |
| Relationships | | | |
| Aggregation (is-part-of) | | Game | |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name:  MainView | ID:  7 | | Type:  Concrete |
| Description:  The object responsible for providing the combined view of the board to be displayed to all the players on a shared screen. | | | Associated Use Cases:  Almost All |
| Responsibilities | | Collaborators | |
| Update View | | Game | |
| Attributes | | | |
| Associated Game | | | |
| Relationships | | | |
| Generalisation (a-kind-of) | | Client | |

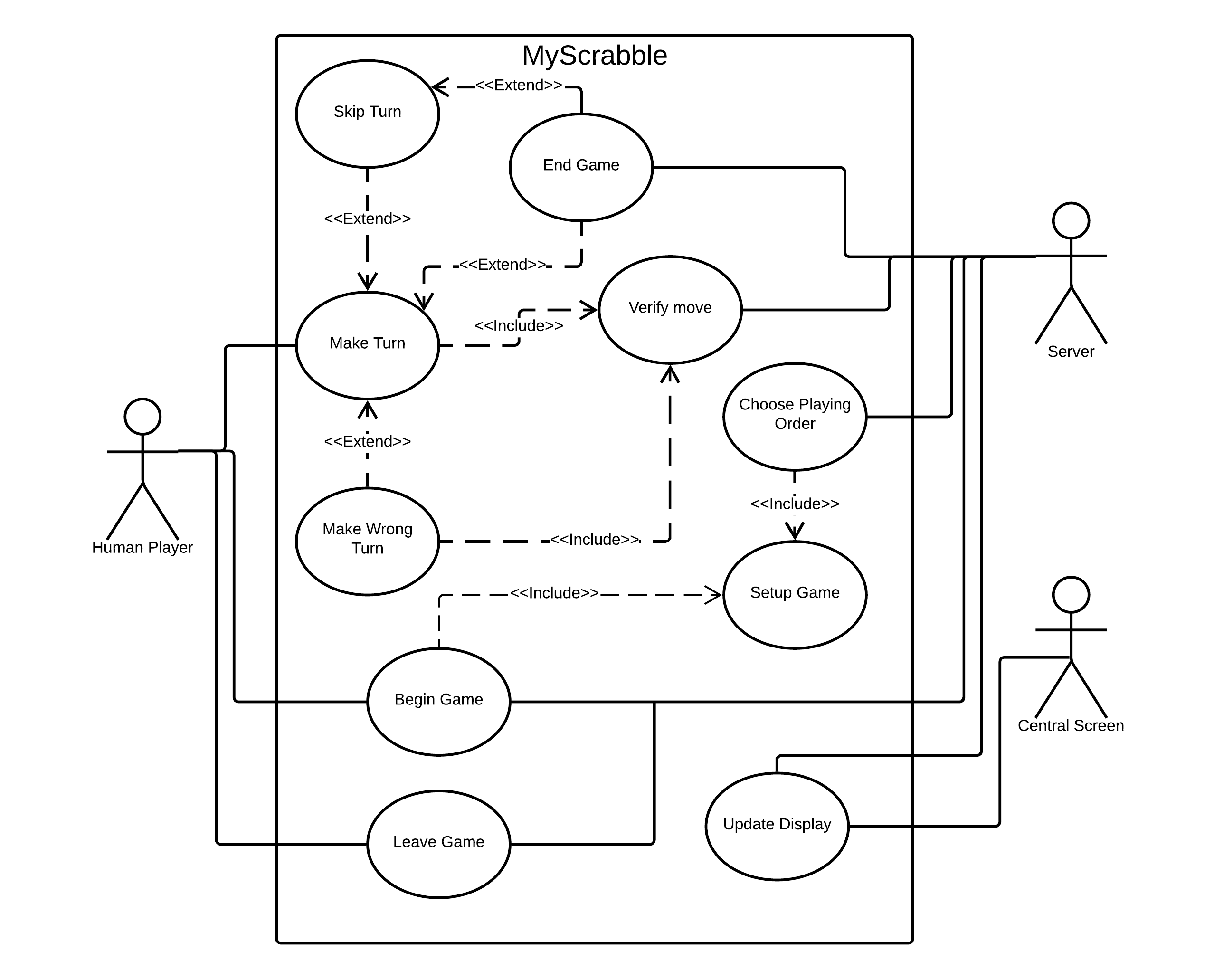
**Class Diagram**

* Diagram made with “LucidChart”.

****

**Use Case Diagram**

* Diagram made with “LucidChart”.



**Structured Walkthrough**

Classes and use cases will be in **bold.** Use Case will be shortened to **UC**

1. The System will start the **Game** and will constantly be displaying new information on the screens for **Players (UC9)**. **Mainview** will show the Menu to the **Player (UC9)**. **Players** must then connect to the website, enter the **Game** code and select “Play **Game**” to join a **Game**. The **Player** must then wait for all other **Player**s to join before starting the **Game**. Once the Lobby is full, no other **Player** may join.

**Branches to:** **UC**2 (Choose Playing Order), **UC**9 (Update Display)

1. When the **Game** is started, **Player** must decide the playing order. Each **Player** will take one letter from the letter bag. **Server** will check the letter that each **Player** has. **Player** with the closest letter to A will be chosen to make the first move, a blank tile is the most valuable tile. All the letters will now be returned to the Letter bag (Letter bag is an attribute of the **Board**). If two or more **Player**s have the same letter, they must choose a new letter each from the Letter bag. The Letter bag will now be shuffled.

**Branches to**: **UC**3 (Setup **Game**), **UC**9 (Update Display)

1. At this point, the **Game** must be set up (use case 3). **Board** will be empty and all the **Player**s will be in a ready state to receive 7 random letters from the letter bag, after playing order is chosen (**UC2)**. Each **Player** will now receive and examine their letters before making a turn (**UC**4) .

**Branches to**: **UC**4 (make turn), **UC**9 (Update Display)

1. The **Server** works out the playing order (**UC**2), and then the **Player** that has been chosen will be the first to go. Make Turn allows a **Player** to move **Piece**(s) on **Board**. Once a **Player** makes a turn, the **Board** returns to a ready state. So, a **Player** plays **Piece**s and the **Game** validates the combination of **Piece**s (words). The **Player** is then awarded points for the word(s). The **Board** then distributes new **Piece**(s) to the **Player**, so that he has a total of 7. If **Player** cannot form a word, he Skips Turn (Use Case 6). If **Server** cannot validate the word (**UC**10), it means a Wrong Turn has been made (**UC**5). If the **Player** has no **Piece** left, the **Game** is over (**UC**8).

**Branches to**: **UC**5 (Make wrong turn), **UC**6 (skip Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. The system will be validating every move that a **Player** makes (**UC**10). When **Player** decides to place a **Piece** on **Board**, if **Game** cannot validate the word(s) (**UC**10), then **Player** will be notified and will take back tiles to have a chance for another move(**UC**4).

**Branches to**: **UC**6 (skip Turn), **UC**4 (Make Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. If **Player** doesn't have any **Piece** that he could use, he has the option to skip his current turn. There are two options for skipping a turn. **Player** may choose to skip a turn to swap out all their **Piece**s for a new set of 7 random **Piece**s from the Letter bag. Option two is that **Player** may also skip a turn without swapping out their **Piece**s, if they choose to do so.

**Branch to**: **UC**5 (Make wrong turn), **UC**6 (skip Turn), **UC**4 (Make Turn), **UC**9 (Update Display), **UC**10 (Verify move)

1. **Player** has the option to leave at any point during a **Game**. Doing so will prevent him from having any interaction with the **Game** he left, and his **Piece**(s) that weren't on the **Board** will be returned to the **Piece** list of **Board**.

**Branch to**: **UC**8 (End **Game**), **UC**9 (Update Display)

1. The **Game** is over and a winner is announced. The **Board** compares the scores of every **Player** and displays the results (**UC**9). If more than one **Player** has equal highest score, the **Game** is a tie. The **Player** has no interaction with the **Board** by this point, he can only observe.

**Branch to**: **UC**9 (Update Display)

1. The **server** will be sending information to all **Client**s so that each **Client**s screens are updated to show new and old information. Information s**UC**h as scores, names, **Piece**s are stored in a cache.

**Branch to: UC1, UC2, UC3, UC4, UC5, UC6, UC7, UC8, UC10**

1. When a **Player** places a **Piece** on the **Board** (**UC**4), the **Board**’s state will be changed. When this happens, these new **Piece**s on the **Board** will form a word(s) which will be validated to ensure it’s a real word(s). The main trigger will be **Player** making a turn (**UC**4), when this happens, the **Server** will validate the word and all other words on **Board**. Update display will now be called and displays the new word on **Board** (**UC**9). If word is incorrect (**UC**5), a failure message is displayed (**UC**9) to the **Player**.

**Branch to**: **UC**4, **UC**5, **UC**6, **UC**9

## Meeting Minutes

## *Meeting 1 – Thursday October 4th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith

# **Agenda:**

* Discussed roles.
* Set up Slack.
* Set up Trello Board.
* Showed Hangman Prototype and decided on basic technology we’d be using for the project (node.js and socket.io).

# **Planned Tasks:**

* Familiarize with requirements
* Look at tutorials on Javascript and jQuery

## *Meeting 2 – Monday October 8th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith

# **Agenda:**

* Discussed requirements.
* Decided ambiguities and questions to ask Renaat.

# **Planned Tasks:**

* Complete requirements specification.
* Research into graphics software that works with socket.io.

## *Meeting 3 – Friday October 12th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith

# **Agenda:**

* Played a game of Scrabble.
* Wrote scenarios using Scrabble game as a guide.

# **Planned Tasks:**

* Create Use Cases from scenarios.

## *Meeting 4 – Thursday October 18th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith

# **Agenda:**

* Discussed classes.
* Decided primary class list.

# **Planned Tasks:**

* Complete CRCs.
* Research and complete Use Case Diagram.

## *Meeting 5 – Monday October 22th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith
* Martin Zaloudek (Only joined the group)

# **Agenda:**

* Tidied up scenarios.
* Did a rough version of the class diagram and the use case diagram.

# **Planned Tasks:**

* Tidy up CRCs.
* Finish Class Diagram.
* Finish Use Case Diagram.
* Do Structured Walkthrough.

## *Meeting 5 – Tuesday October 23rd*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith
* Martin Zaloudek

# **Agenda:**

* Redesigned the Client class to be a superclass of Player and MainView.
* Made necessary changes to accommodate this change.

# **Planned Tasks:**

* Make 2 new use cases that are required.
* Change CRC to reflect class change.

## *Meeting 6 – Wednesday October 24th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith
* Martin Zaloudek

# **Agenda:**

* Finished off Class Diagram.
* Worked on Structured Walkthrough.

# **Planned Tasks:**

* Finish structured walkthrough.
* Gather everything into one document.

## *Meeting 7 – Thursday October 25th*

# **In Attendance:**

* Aidan Fogarty
* Daniel Perecz
* Eoghan Murphy
* Michael Downling
* Oishin Smith
* Martin Zaloudek

# **Agenda:**

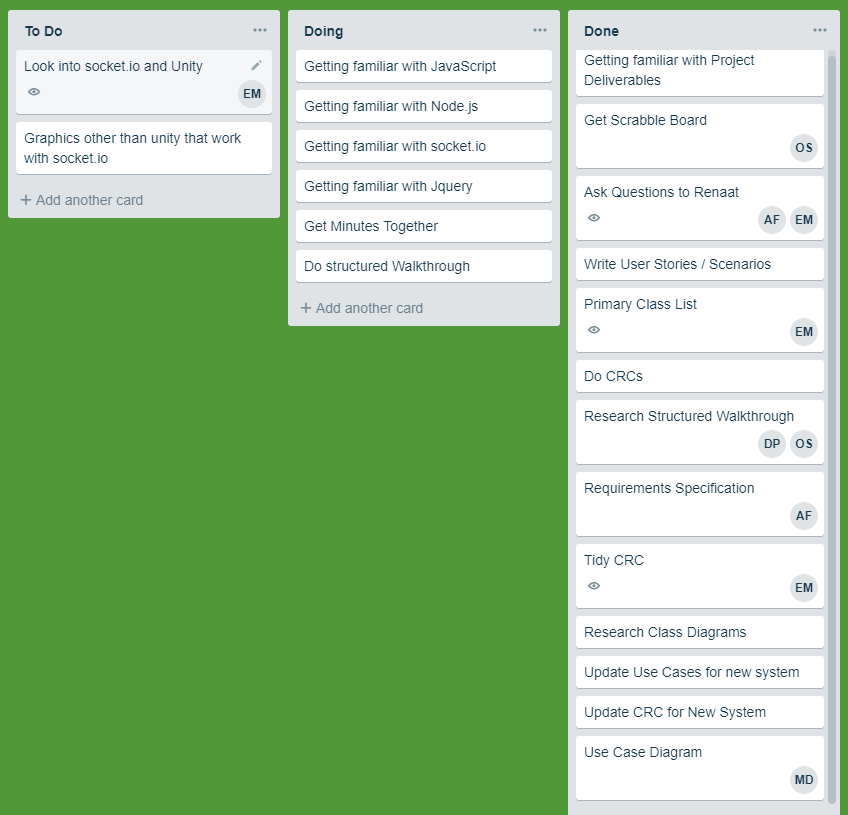
* Read over everything.
* Submitted analysis part of the project.

# **Planned Tasks:**

* Meet again next Monday to discuss the next section of the project.

Extra Information regarding record keeping

We kept a Trello board to keep track of the status of tasks:



We also had a git repository to make sure all our files were up to date:

