BILKENT UNIVERSITY

CS 319: OBJECT-ORIENTED SOFTWARE ENGINEERING

ANALYSIS REPORT

DRAW IT!

Group 4 Section 1

Görkem Çamlı - 21302603

Oğuz Bilgener - 21302523

Umut Cem Soyulmaz - 21201744

Hilal Öztürk - 21000633

Course Instructor: Hüseyin Özgür TAN

Teaching Assistant:Fatma BALCI

Table Of Contents

Table Of Figures

1. Introduction - Problem Statement

"Draw It!" is an entertaining application which has lots of common points with famous "Draw Something" game. The "Draw It!" is a multiplayer game and its concept is mainly constructed on drawing the given words and guessing the drawn images in a particular time to gain points.

During this report, general pre-analysis of the project are going to be investigated. In the first part, the main point is to clearly indicate the features of gameplay including rules. During the next step, list of requirements and their analysis are explained properly. Final part of the report includes scenarios, use case models, UML diagrams and user interface figures.

2.Requirement Analysis

Requirement Analysis part contains an overview, requirements for the application, scenarios, Use Case Models and the Use Case Interface of "Draw It!" application.

2.1. Overview

"Draw It!" is a Java based computer game which requires a connection between computers. By using this connection, when a player is drawing an image, the other player is able to see the drawing at the same time. In this concept, the aim of the game is to reach the maximum points by drawing and guessing accurately the challenges. The game is played with rounds and at the end of every round, the duties of the players are replaced. During the drawing process, users are able to change the colour and size of the drawing stick to make it more understandable for the player who is guessing.

The “Draw It!” game consists of three main steps. In the first step, there is a menu with three buttons which are “Host”, “Join”, “Credits”. One of the users clicks on the “Host” button to create a new game and the only thing that the second player needs to click on the “Join” button to start the challenge. When the game is started, in the first round host player is the one who is going to draw a figure and the guest player is the one who guesses. Before starting to draw, active player, whose duty is to draw, chooses a word among three random words that the game offers. When the drawer player starts to create an image, the passive player, whose duty is to view and guess, is able to see the drawing in real time. The drawing process needs to be done in 45 seconds and after this 45 seconds, the guessing time period is started. The passive player has also 45 seconds to find the name of the object by filling the blank with letters that are located at the bottom of screen. If the passive player finds the word, the time period is over and passive player gets ten points but if they cannot make it, s/he gets zero points. After every round, the active player becomes passive player and the passive player becomes active player. The game continues until one of the players clicks on the finish button and when the game is finished, both of the players see their scores.

2.2. Functional Requirements

- Players must be able to start or join a new game over a network.
- Players must be able to use and control their mouse during the game (while drawing and guessing).
- For each round, active player should be able to select his/her own word from the given words.
- During the game, passive player should receive points according to his correct responses.
- Players must be able to exit from the game anytime they want to. In this case, game will be over for both players and the scores will be shown.

2.3. Non-Functional Requirements

- Before starting the game, the connection between two players (host and guest) should not take more than 2 minutes long.
- Active player should be able to change the colour and the size of the brush.
- 45 seconds should be given for both drawing and guessing parts of the round.
- Application should not allocate more than 1 GB of space.
- When passive player finds a word correctly, s/he should get 10 points.

2.4. Constraints

- The “Draw It!” game will be implemented in Java.
- The application should work in all operating systems.
- The application should work on the local network.

2.5. Scenarios

Participating Actors: Player (John, Valerie)

John and Valerie are both very famous painters and they decide to meet in 18th September 2015 at the John's studio. After 6 pm, they get bored and want to try something new for them. Suddenly, John finds an application which is called as "Draw It!" in his personal computer and he asks Valerie to play this game together. Then, Valerie accepts the challenge and download the game for her own personal computer. John clicks on the icon of the game on his computer's desktop as well as Valerie. John clicks on the "Host" button and he sees his personal computer's IP address on the screen and he directly tells this IP address to Valerie. Valerie clicks on the "Join" and she fills the blank space by entering the IP address of John's personal computer. When the connection is provided, Valerie encounters with a waiting screen because at this moment John needs to choose a word among three randomly chosen words. After choosing a word from the list, John starts to draw the word and the game begins officially. When John is drawing the image, Valerie can see the mouse movements of John in real time by using the connection. Every time John releases the left mouse click after a drawing movement, the drawing piece for this period is seen by the Valerie at the same time. When John finishes his drawing, he can click on the "Check" option or he can wait until the time countdown hits to zero. If he cannot finish his drawing until end of the time limit (45 seconds), his turn passes. When he finishes his work before the time is up, entire image is seen on the screen of Valerie and the time countdown starts for her. During the guessing period, Valerie can fill the word space by clicking on the given letters in the guess box to find the word that John chose. If Valerie thinks that she cannot find it, she can click on the "?" option to give up and her turn passes. In addition to this, if

Valerie cannot find the word until the time is over, his turn passes as well. If she can find it before the time is up for her, she gains 10 points and the first round is over. At the end of the first round, John and Valerie changes their roles. For the second round, John becomes the passive player and Valerie becomes the one who draws. At the end of every round, game system changes the roles of the players. The game continues, until Valerie decides to go home to cook some meal for her children and she clicks on the "Stop Game" option. After finishing the game, they both sees their personal scores and Valerie wins the challenge. Before closing the application, John wonders the names of people who worked for the construction of this game and he goes back to the main menu to click on "Credits" to see the participants of the project.

2.6. Use Case Model

We intend to make our Use Case Diagram as simple as possible in order to be comprehensible by the client. Below is our Use Case Diagram and Use Cases.

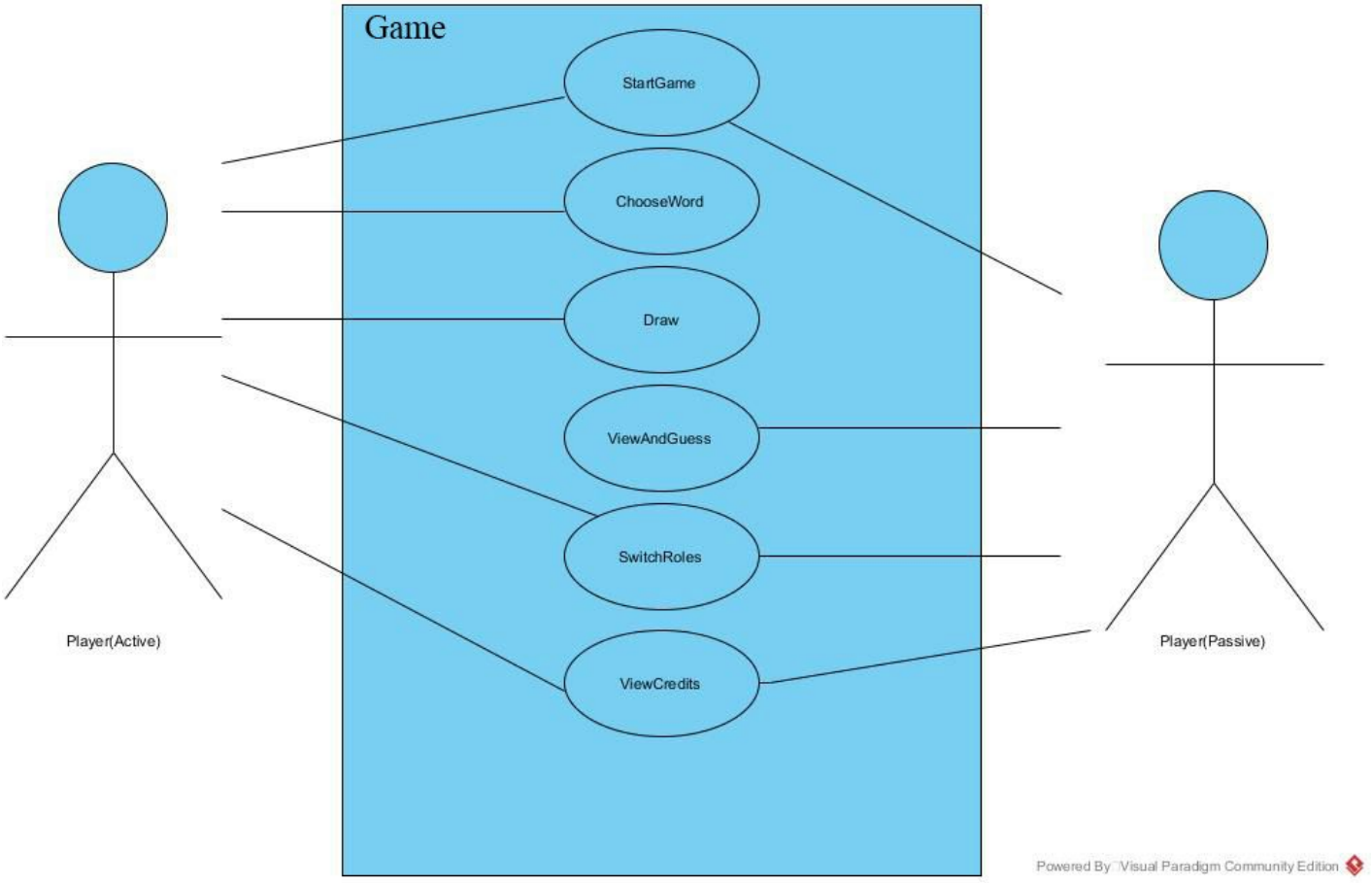

Figure 1. Use Case Diagram

2.6.1. Use Case 1: StartGame

Use Case Name:	StartGame
Participating Actors:	Initiated by Player. Communicates with another Player.

Flow of Events:

1. Player1 activates the "host" function of the main menu.
2. Game shows the IP address of the Player1 in a new screen.
3. Player1 starts to wait until Player2 enters his IP address, Player2 enters Player1's IP address.
4. Game opens choose word step to Player1, when Player2 is waiting for the first round to start.

Entry Condition:

- Right after two player opens the game .exe, Player1 clicks on the host button and Player2 clicks on the join button.

Exit Condition:

- Player2 enters Player1 IP address.

Quality Requirements:

- Player2 must click on the join button and must enter Player1's IP address at most 2 minutes.

2.6.2. Use Case 2: ChooseWord

Use Case Name: ChooseWord

Participating Actors: Initiated by Player.

Flow of Events:

1. Game shows the three words in a new window to Player.
2. Player chooses one of these word to draw.
3. Game starts the round by opening new drawing screen.

Entry Condition:

- Second player joins the game.

Exit Condition:

- Player clicks on the word he chooses.

Quality Requirements:

- -

2.6.3. Use Case 3: Draw

Use Case Name: Draw

Participating Actors: Initiated by Player.

Flow of Events:

1.Game starts the countdown of the timer.

2. Active Player (player who draws) starts to draw the word with any color and size he selects.

3. Active Player continues to draw until time is up or he clicks on the finish button.

4. Game finishes active player's part in the round.

Entry Condition:

- Active Player clicks on one the three words.

Exit Condition:

- When time is up.
- When Active Player clicks on the finish button.

Quality Requirements:

- Required time is 45 seconds.

2.6.4. Use Case 4: ViewAndGuess

Use Case Name: ViewAndGuess

Participating Actors: Initiated by Player.
Communicates with another Player.

Flow of Events:

1. Passive Player (player who guesses) sees each drawing movement of the the active player with real-time.

2. After drawing process ends, game initializes Passive Player's guess time and makes visible the guess box.

3. Within the given time, Passive Player tries to find the chosen word. If he clicks on the wrong letter he click on the Trash Bin icon. One clicked icon erases the all chosen letters and passive player should start to select them again.

4. If he thinks that he can't find the word, he clicks on the "?" button. If he finds the word or if time is up, his turn passes.

5. If passive player fails to find the word, uses box turns into red. Correct word shown. If word found correctly, guess box turns into green and player receives 10 points.

Entry Condition:

- Active player releases the mouse for the first time whilst he draws.

Exit Condition:

- Time is up.
- Passive player finds the word correctly.
- Passive player gives up and clicks on the "?" button.

Quality Requirements:

- Required time is 45 seconds.

2.6.5. Use Case 5: Switch Roles

Use Case Name: SwitchRole

Participating Actors: Initiated by Game.
Communicates with the other two Players.

Flow of Events:

1. When each round is finished, scores of the players is updated by Game Controller.

2. If player 1 is active player for the finishing round, guessing window is sent to him or vice versa. If player 2 is active player for the finishing round, guessing window is sent to her or vice versa.

Entry Condition:

- Round is finished.

Exit Condition:

- Both active and passive players takes new screen.

Quality Requirements:

- -

2.6.6. Use Case 6: ViewCredits

Use Case Name: View Credits

Participating Actors: Active and passive players

Flow of Events:

1. Players use Credits button on the main menu.
2. Game displays the contributors of the game.

Entry Condition:

- Press on Credits button.

Exit Condition:

- Press “Back” button.

Quality Requirements:

- -

2.7. User Interface

User Interface is one of the crucial thing in designing process. We care our application to be user-friendly. The images below are only rough sketches, for now.

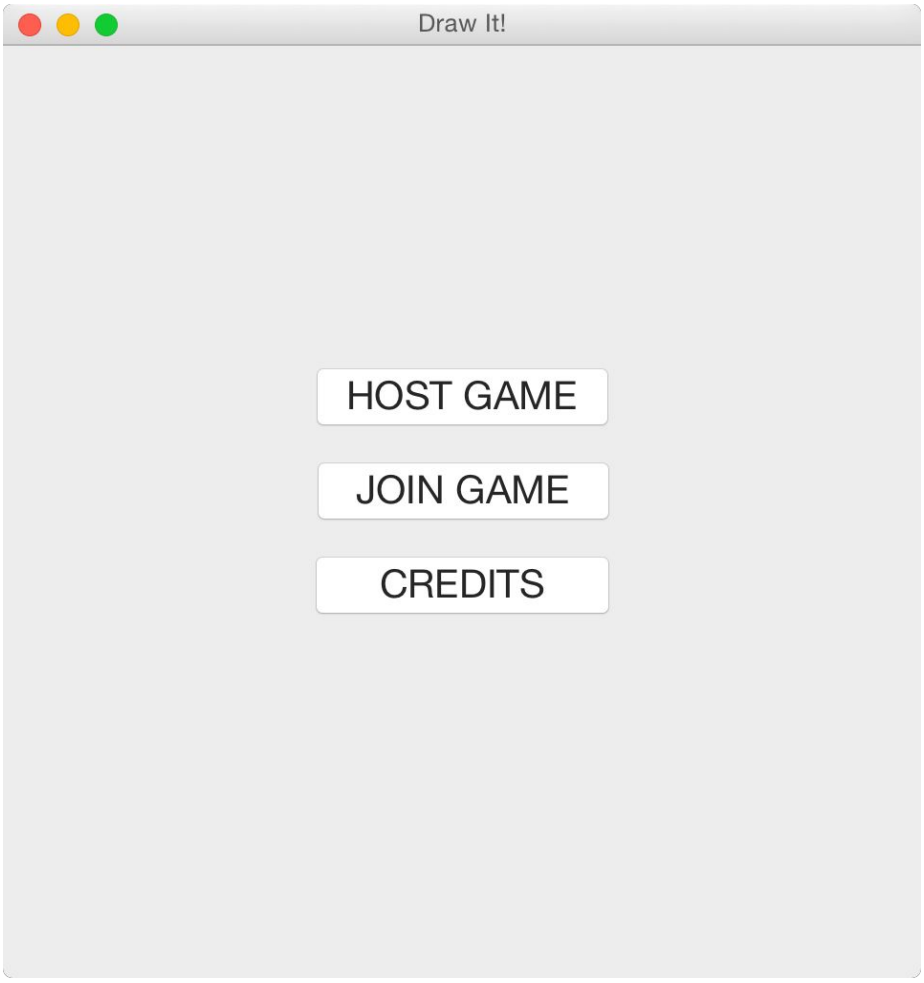

Image 1: Main Menu

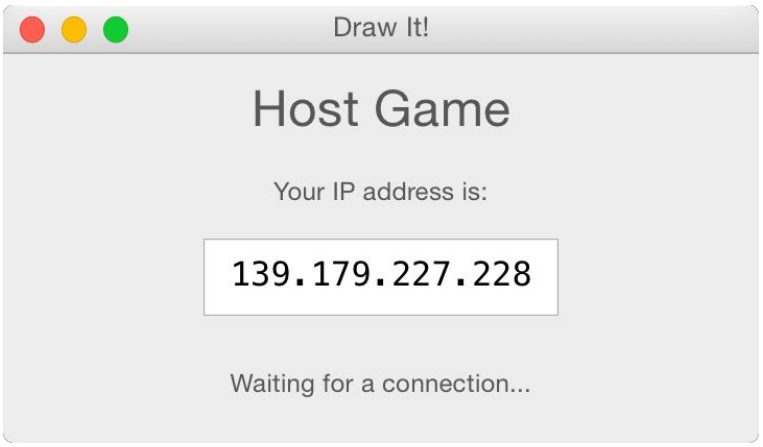

Image 2: Host Game

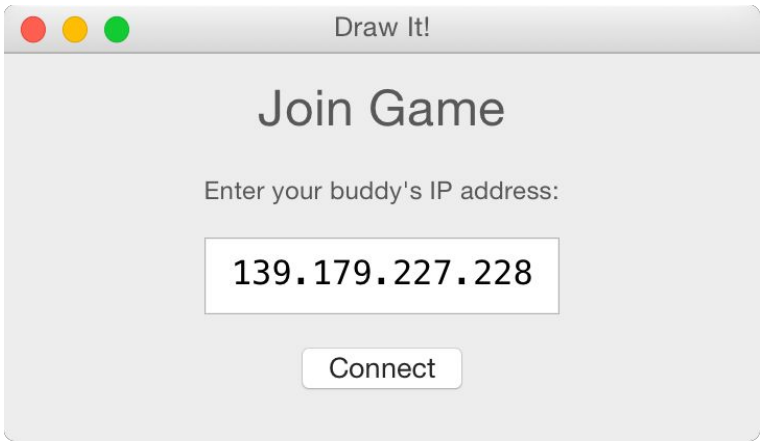

Image 3: Join Game

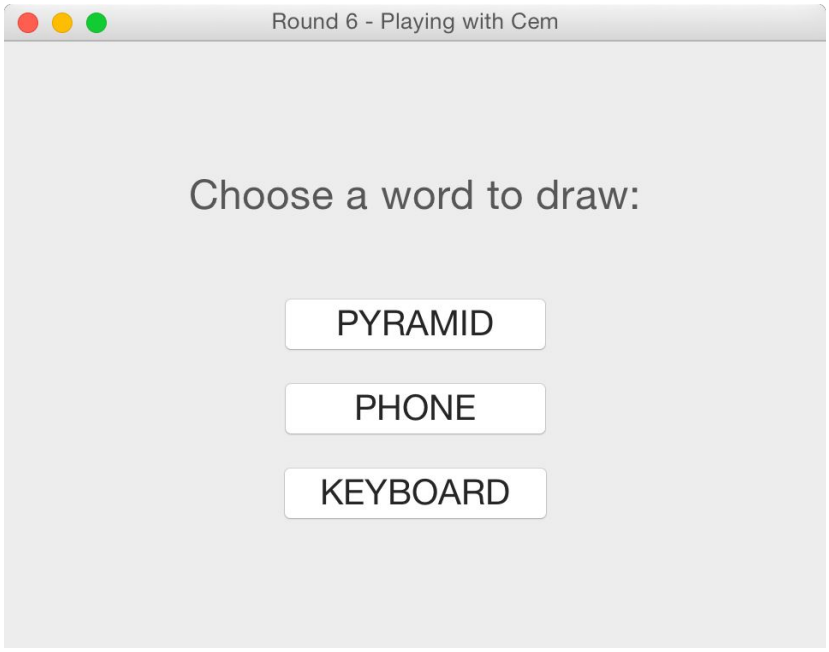

Image 4: Choose Word Screen

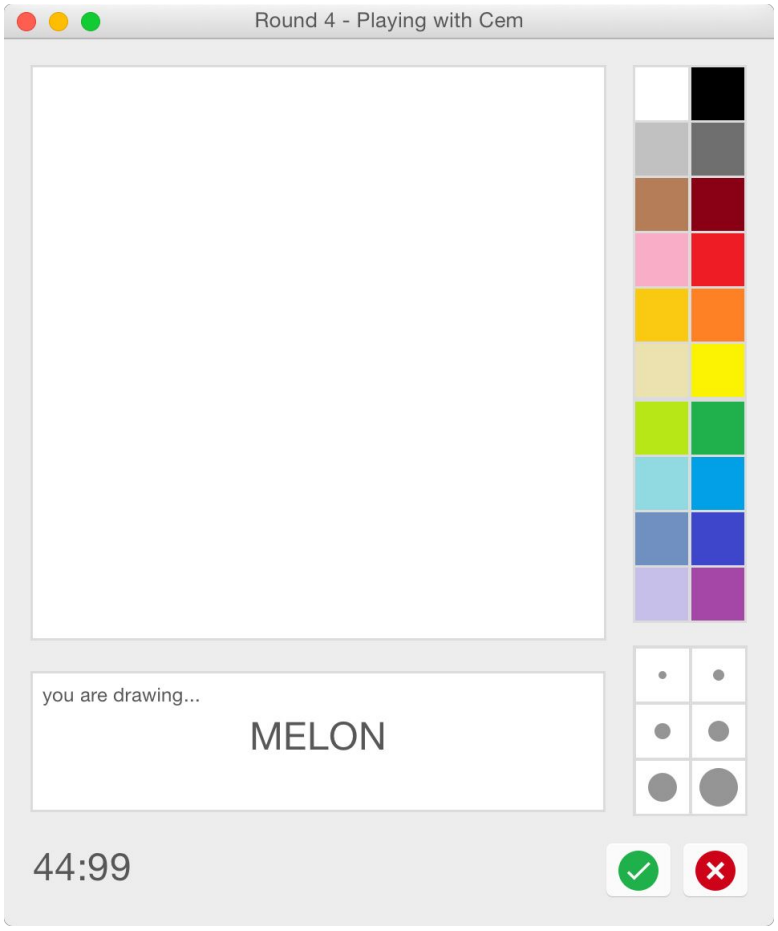

Image 5: Drawing Screen

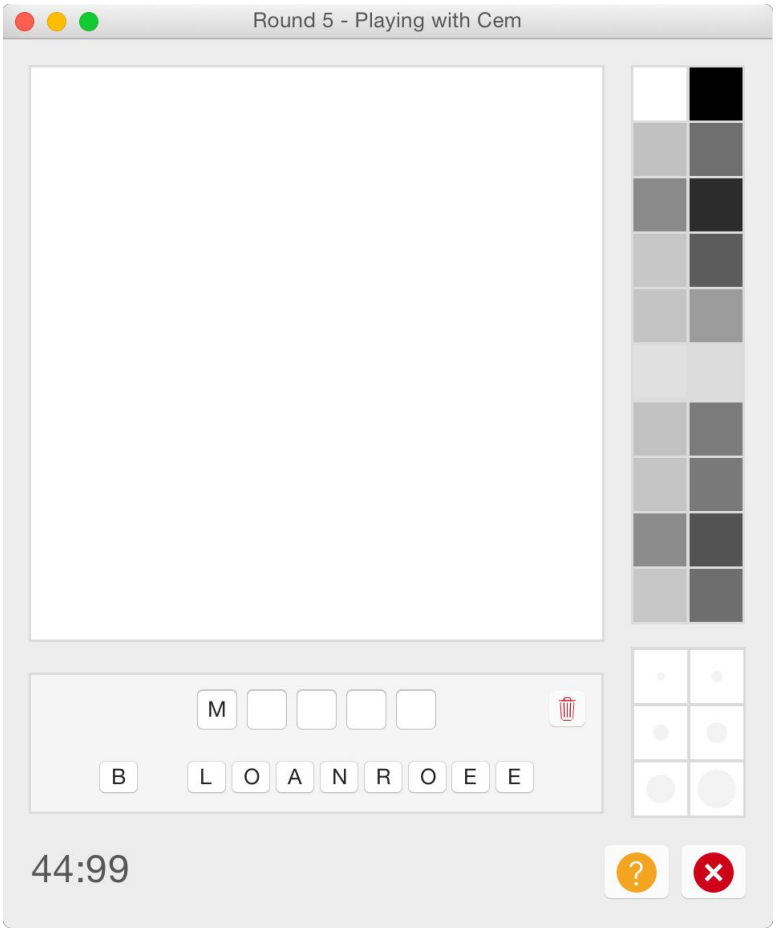

Image 6: Guessing Screen

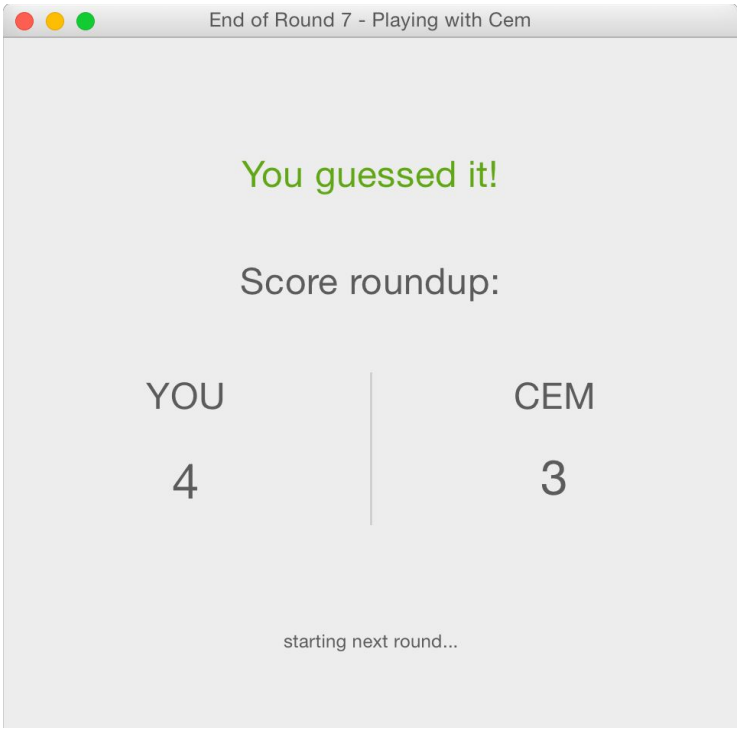

Image 7: End Of Round

Image 8: Credits

3. Analysis

Analysis part contains object model and dynamic models sections for the “Draw It!” application. More detailed explanation about “Draw It!” Game and diagrams are given in this section.

3.1. Object Model

Object Model part includes Domain Lexicon and Class Diagram. Domain Lexicon consists of the “Draw It!” application keywords’ explanation and aims to clarify ambiguous words for the user.

3.1.1. Domain Lexicon

Active Player: The player who draws in that particular round.

Passive Player: The player who guesses the word in that particular round.

Drawing Movement/ Piece: The mouse movement between mouse press and mouse release that Active Player made while drawing.

Guess Box: The box that Passive Player use while making his guess. It contains 10 random letters and n many empty letter boxes where n is the length of the chosen word.

“?” Button: This button is used by Passive Player when s/he thinks, s/he cannot find the word. When player clicks on that button s/he gave up from her turn.

Close Button: This button is used by both players when they want to quit from the game. When players press this button, they can see their total point in the game.

Check Button: This button is used by Active Player. S/he clicks on that when s/he thinks s/he finished the drawing.

Trash Bin: Passive user use this icon to clear the guessing area when he clicks on the wrong letter. Once clicked, all chosen letters are erased.

3.1.2. Class Diagram

In our class diagram, we have Player, Canvas, Piece, NetwrokLayer, TurnTimer and differen control classes such as WordDrawingControl, WatchControl, GuessWordControl and GameController class.

Game Controller class mainly responsible to showing screens and checking whether the users connected or not. It also responsible to starts the rounds and the turns, it updates the player scores as well. Player class holds a reference to basic player information such as IP address and score of the player. It has a variable called active as well, which is the information for whether the player is active or passive in that specific round. NetworkLayer class is the class which we make our connection and use for sending information. This class is for managing the network connection between the two game instances. ConnectToHost(), turnFinished(success), roundStarted(word) and sendPiece() method are parts of the NetworkLayer class.WatchControl controls the drawing watching stage of the round and it contains drawFinished(), onNewPiece(piece) and operation() methods.

TurnTimer class is the class which we keep count of the drawing and guessing process. This class checks whether the time is up. Time starts when the class's constructor is called. WordDrawingControl and GuessWordControl are using the TimerDelegate interface.

TimerDelegate interface has the onTimeout() method which notify that time is up. WordDrawingControl is the part here we control drawing. It has simple feature like selecting color and brush. Other than these, it has also isFinished(), startDrawing() and finishDrawing() method, as well as createPiece, addToPiece(point) and finishPiece() methods. On the other hand, GuessWordControl is where the part we control guessing process. This part has guess and word variables in order to check whether our guess and word matches. finishGuessing(), isFinished(), setLettter(position, letter), wordCompleted(), guessCorrect(), clearWord() and giveUp() methods.

Also, we have Canvas and Piece classes. Our Piece class is representing a piece of drawing which we call drawing movement as well throughout our report. Piece class has point, brush and color variables. Canvas is basicly for providing a surface to draw something or to see the drawn figure. Canvas class has drawPiece(piece), mousePressed(), mouseDraged(), mouseReleased and refresh() functions.

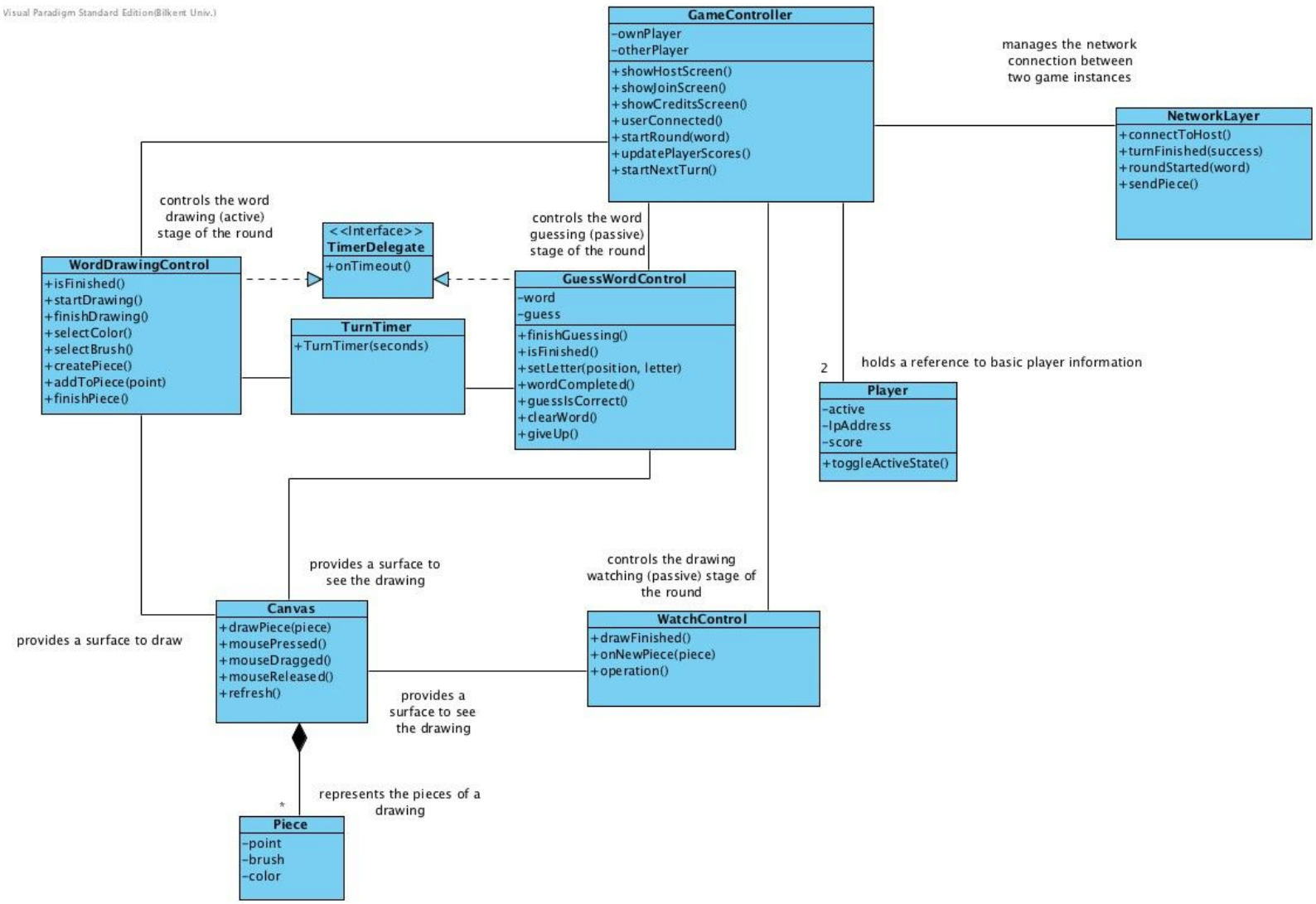

Figure 2. Class Diagram

3.2. Dynamic Models

Dynamic Models part includes the State Chart and Sequence Diagrams along with their explanation and scenarios with the aim of providing better understanding about "Draw It!" Game.

3.2.1. State Chart

We have 3 State Chart Diagrams. One is a more general state diagram which explains how role exchange occurs. The other two are the states of Active Player and Passive Player's separately.

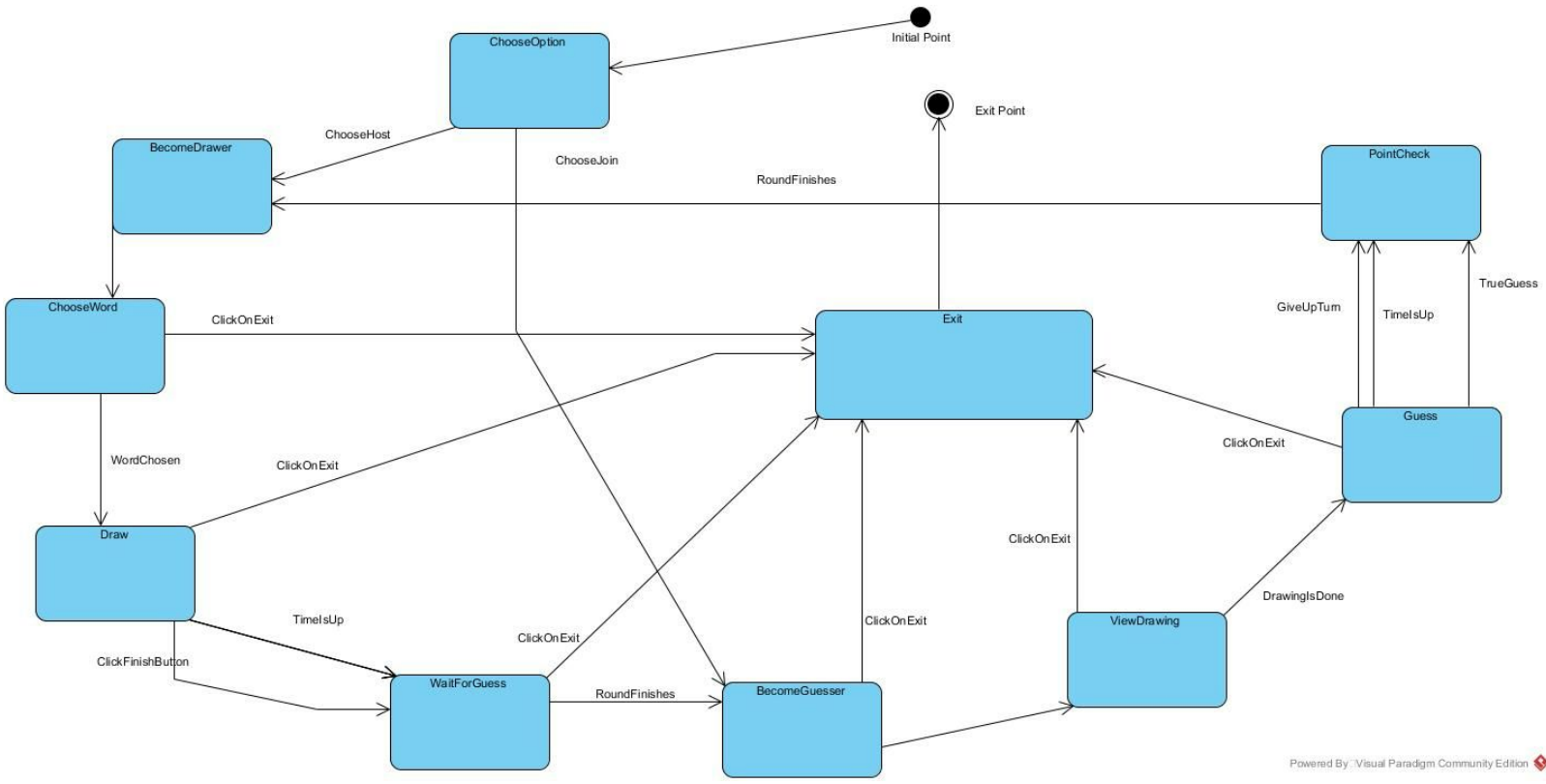

Figure 3. State Chart Diagram 1: Exchange Roles

In the Exchange Roles state chart diagram, we introduce the behaviour of a player during the game process. There are two playing options which are hosting and joining. If the player chooses hosting, he needs to choose a word in the first step of the game. After the word choosing condition is finished, the next step is to draw the word. When he finishes the drawing process, he needs to wait for the end of guessing process of the other player. When the round is over, he becomes the player whose duty is to guess the word which is drawn by the other player. In short words, there is a role exchanging process. After this step, his role is turned into being a drawer. If at the beginning of the game, the player chooses the "Join" option, his movement starts from the BecomeGuesser state of the diagram and goes with the same cycle of hosting option and this cycle continues until one of the players wants to exit from the game. Players are able to exit from the game during the ChooseWord, Draw, WaitForGuess, BecomeGuesser, ViewDrawing and Guess states of the state chart steps.

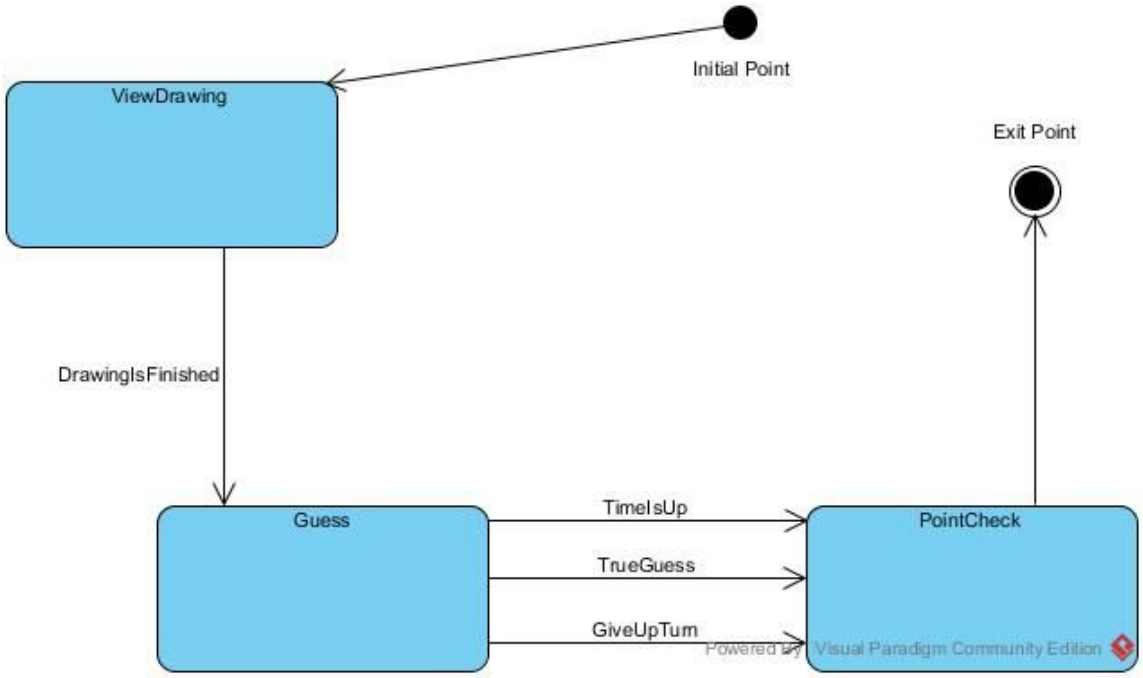

Figure 4. State Chart Diagram 2: Passive Player State

In this state chart diagram, we explain the behaviour of a passive player (the one who views and guesses the drawing). At the first step, he needs to view the drawing. When the drawing is done, in the guessing process, he needs to be successful to find the word or time needs to be up or the player needs to click on "?" option to go to the next step which is called as point check to see the current point situation. After points are shown, the lifetime of the passive player ends.

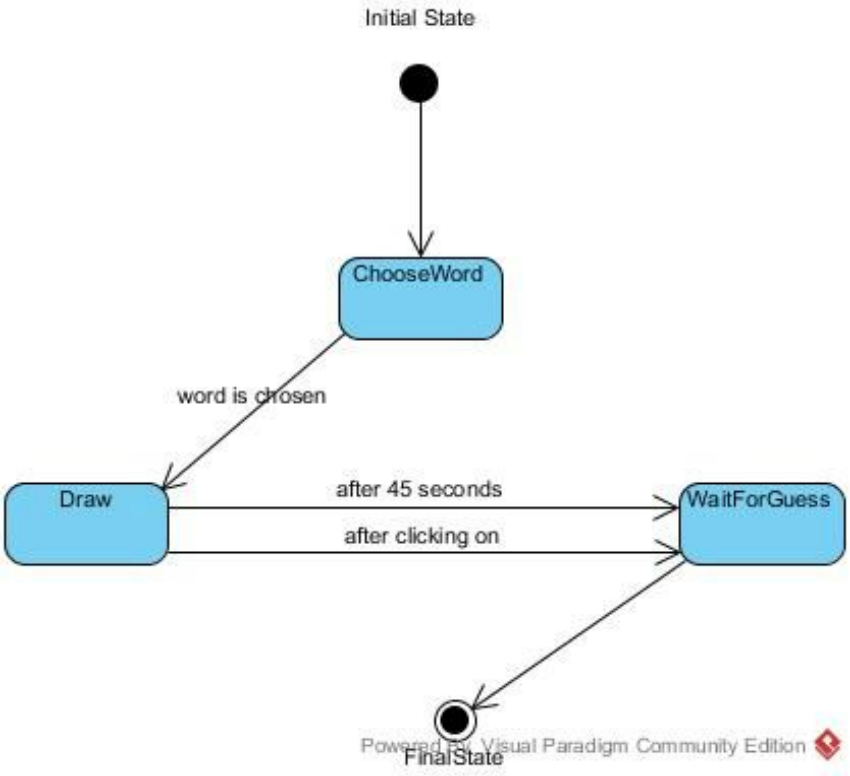

Figure 5. State Chart Diagram 2: Active Player State

In Active Player State chart diagram, we explain the behaviour of an active player (the one who is drawing). At first, the active player chooses a word to draw. After choosing a word he needs to draw it in 45 seconds or needs to click on the "Check" option to go the next step which is called as WaitForGuess. After the guess result comes, lifetime of the active player ends.

3.2.2. Sequence Diagram

This part contains scenarios and their sequential diagrams.

3.2.2.1. Scenario 1: “menu” Scenario

Scenario Name: menu

Participating actor instances: john:Player , valerie:Player

Flow of events:

1. John and Valerie both click on the icon (exe) of the “Draw It!” application and they start the execution of the program.

2. John and Valerie both encounter with the main menu which has “Host”, “Join” and “Credits” options.

3. If they want to play the game, John clicks on the “Host” option and Valerie clicks on the “Join” option. This case could be in reverse roles.

4. If John or Valerie want to see the participants of the game project, only thing they need to do is clicking on “Credits” option.

5. If John or Valerie clicks on the “Host” option, s/he starts to wait for the other player to play the game and during this waiting process, he sees his device’s IP address. This IP address is for the other player because the one who wants to join

the game needs to use it. When the other player joins the game, John finishes his work with this step and goes to the part that he chooses the word.

6. If Valerie or John clicks on the "Join" button, Valerie needs to fill the IP address space by using the IP address that is shown in the host player's screen. After filling this space, the connection is provided and the game begins.

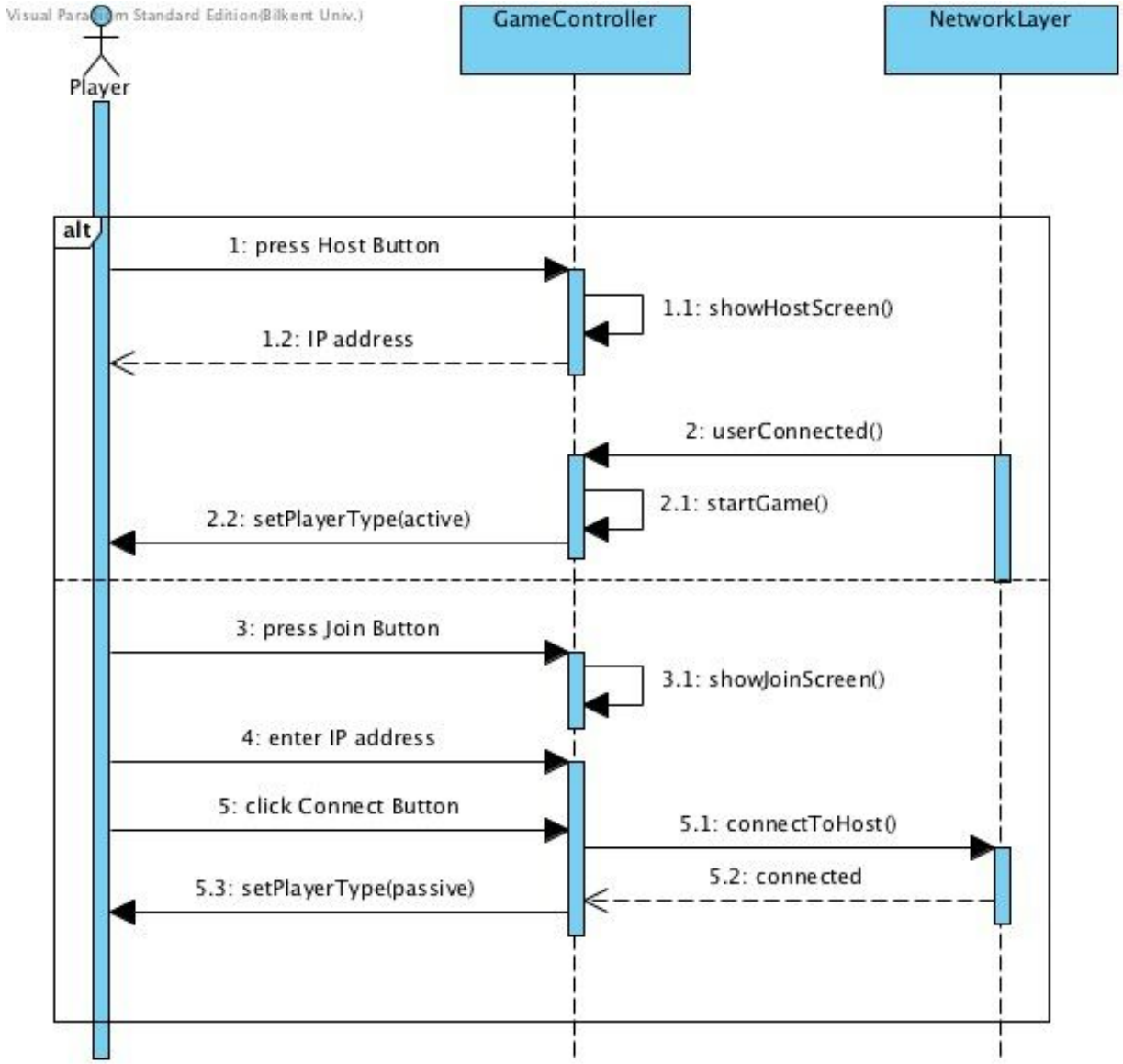

Figure 6. Sequence Diagram 1: "menu" Diagram

3.2.2.2. Scenario 2: "showCredits" Scenario

Scenario name: **showCredits**

Participating actor instances: **john:Player**

Flow of events:

1. One of the players clicks on the Credits button on the main menu.

2. The network sends new canvas which shows the names of contributors and contact information of this game to the player.

3. After choosing the "Credits" option, John finishes his work with the main menu.

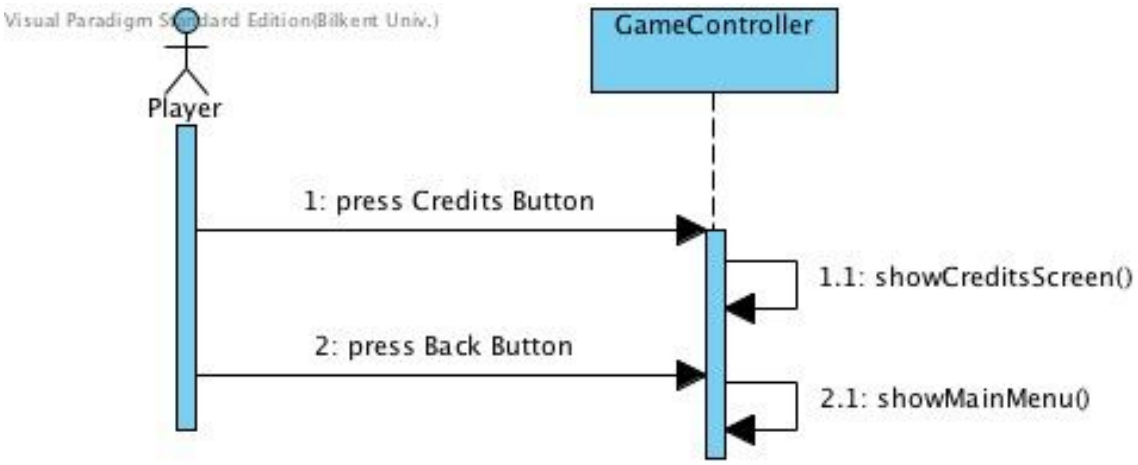

Figure 7. Sequence Diagram 2: "showCredits" Diagram

3.2.2.3. Scenario 3: “wordChoosing” Scenario

Scenario Name: wordChoosing

Participating actor instances: john:Player

Flow of events:

1. John encounters with three different words which are listed randomly.
2. John chooses one of them which he can draw.
3. After choosing one of them, John finishes his work with the word choosing step.

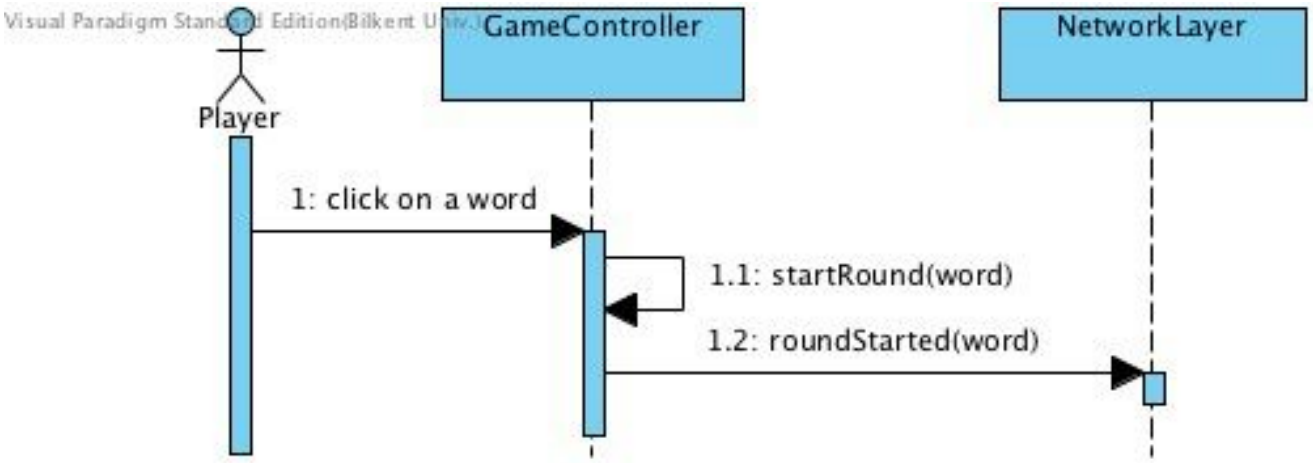

Figure 8. Sequence Diagram 3: “wordChoosing” Diagram

3.2.2.4. Scenario 4: “wordDrawing” Scenario

Scenario Name: wordDrawing

Participating actor instances: john:Player , network:NetworkLayer

Flow of Events:

1. John may select a color from color panel.
2. He presses the mouse when mouse cursor points a specific point (point where he wants to start drawing) in the canvas.
3. He draws the figure partially or completely.
4. He releases his mouse.
5. System send the point coordinations of his drawing to the network.
6. If he didn’t finish drawing, he repeats the first 5 steps.
7. If required time (45 seconds) is up or John clicks the finish (check) button, drawing word process ends.
8. The active player’s (John’s) part of the round finishes.

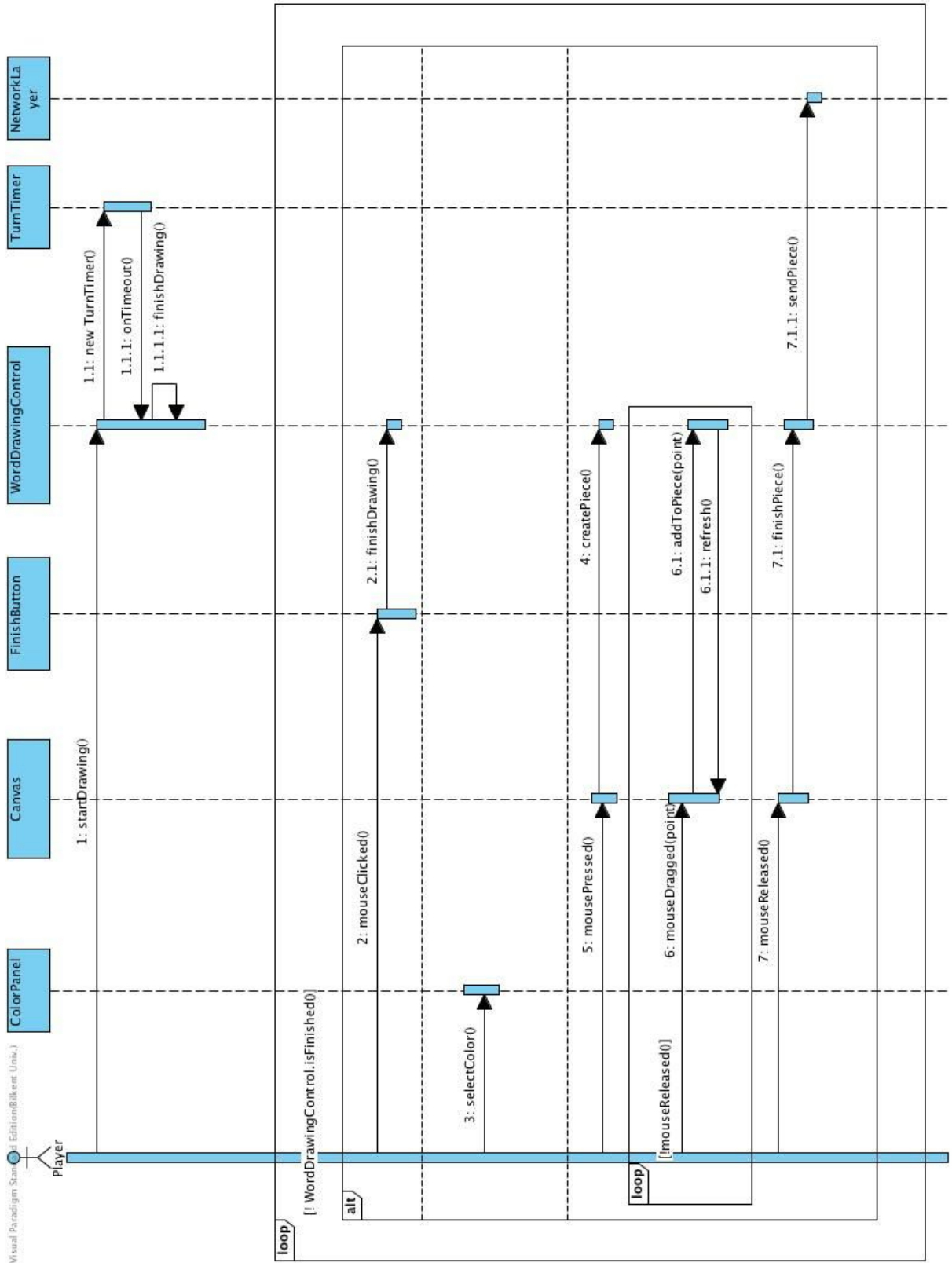

Figure 9. Sequence Diagram 4: "wordDrawing" Diagram

3.2.2.5. Scenario 5: "watchTheWord" Scenario

Scenario Name: watchTheWord

Participating actor instances: network:NetworkLayer

Flow of Events:

1. The coordinates of the drawn points come from the network to the passive player.
2. The points are drawn to the canvas as soon as they come according to their x,y coordinates and their colour.
3. If the required time, 45 seconds, didn't finish or if the active player (John, who draws) didn't click on the finish (check) button the first two step repeats.
4. If the required time, 45 seconds, finishes or if the active player (John, who draws) clicks on the finish (check) button, watchTheWord process ends and no data of points comes. Canvas stays in its last condition with the drawings.

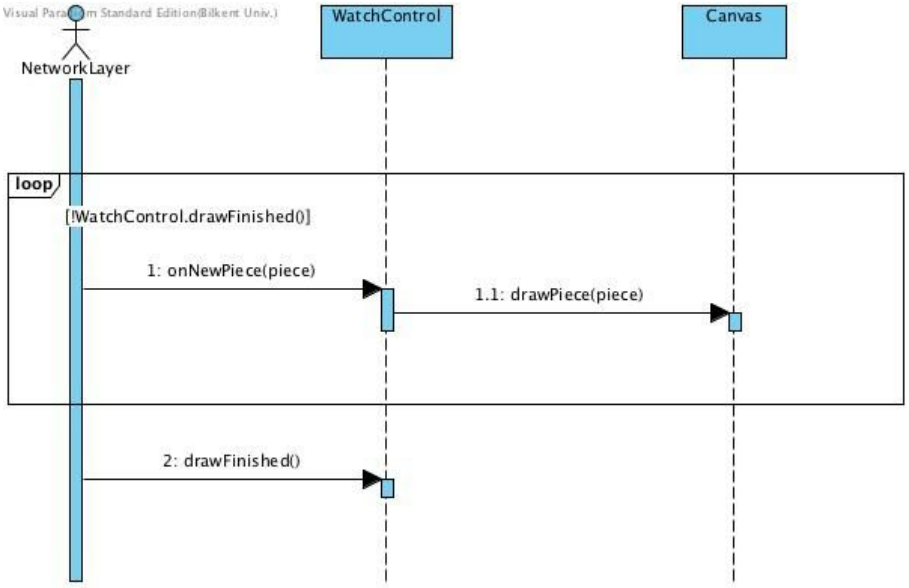

Figure 10. Sequence Diagram 5: "watchTheWord" Diagram

3.2.2.6. Scenario 6: "guessingWord" Scenario

Scenario Name: guessingWord

Participating actor instances: valerie:Player

Flow of Events:

1. System adds a box under Valerie's (the passive player's)canvas. Box consists of "_empty letter boxes for each letter that word has.

   Ex: Word: Flower -> _ _ _ _ _ _

   10 letters also given under the dashed lines. The letters have some extra letters as well as the word's letters.
2. Valerie starts to guess the word by clicking the letters with the same order as the word she guess has.
3. If she clicks on the wrong word she press Trash Bin button and clear button clear all the letters and Valerie starts the guess the word all over again!
4. If she can't find the word, Valerie clicks on the "?" button and that means she gave up from her turn. Her part ends.
5. When time is up to 45 seconds, the guessing time ends, therefore her part ends.
6. When her part ends, if the entered word correct guessing box colored to green. System gives 10 points to Valerie.
7. If it is false, first it is colored to red and 5 seconds later box colored into its original color and shows the correct word.
8. The round finishes.

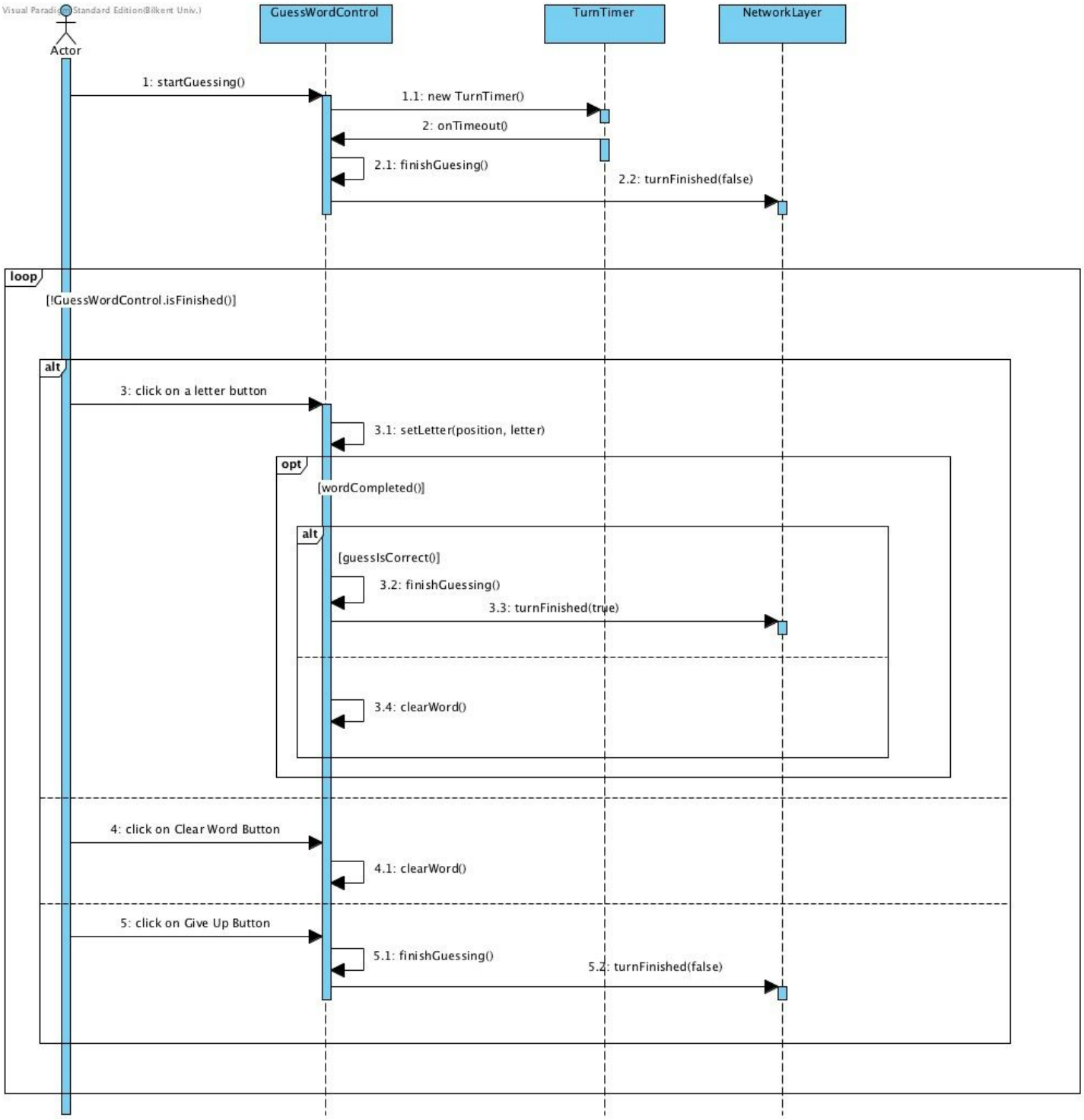

Figure 11. Sequence Diagram 6: "guessingWord" Diagram

3.2.2.7. Scenario 7: “roleExchanging” Scenario

Scenario Name: roleExchanging

Participating actor instances: valerie:Player john:Player network:Network

Flow of events:

1. At the end of each round, role of the active and passive player is changed and if passive player guesses the drawn word correctly,his/her point is updated.

2. The new canvas is sent to Valerie over the network. If she is active player, the next round she will be passive player.

3. The new canvas is sent to John over network. If he is passive player, the next round he will be the active player.

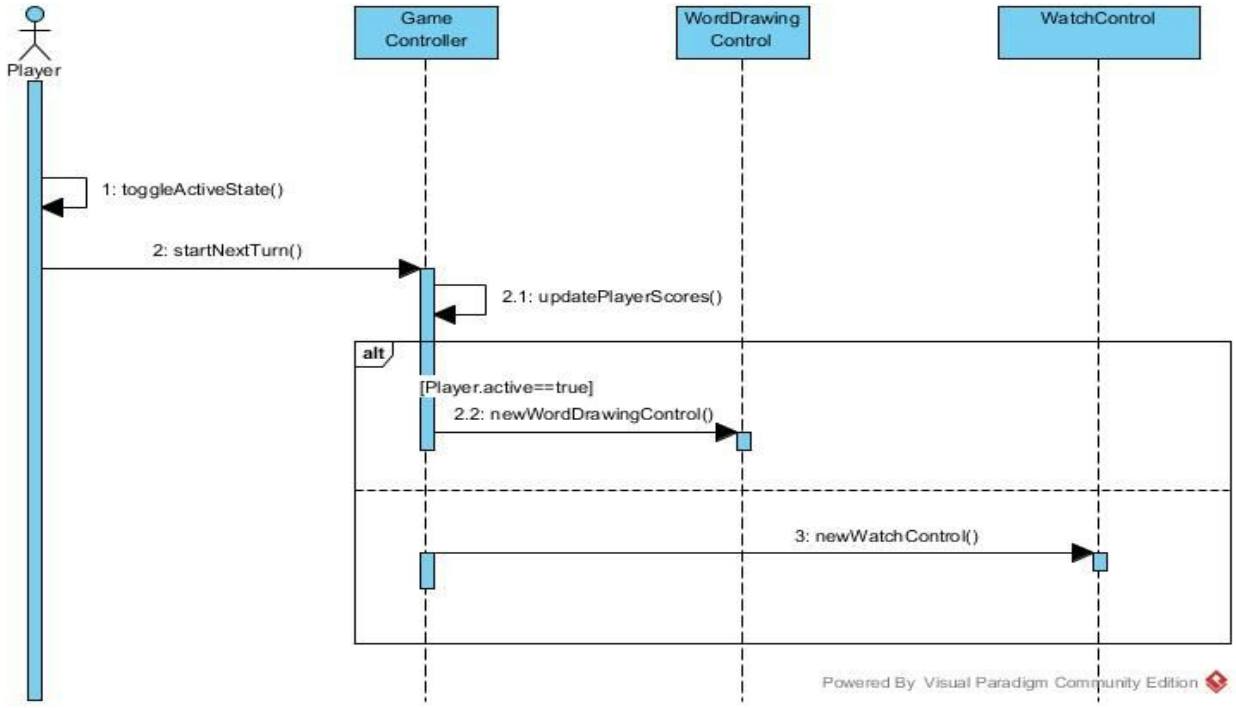

Figure 12. Sequence Diagram 7: “roleExchanging” Diagram

4. Conclusion

In the Analysis Report, we made the first step towards the Object-Oriented Design Concept which is designing the project. We prepared a detailed design for "Draw It!" application. Our report has two main parts one is Requirement Analysis and the other is Analysis section.

In the Requirement Analysis section, we first give an overview about our application. We explained Functional, Non-Functional Requirements and the application's constraints. Our criteria was to consider all possible situations for the user can perform. Therefore, we tried to find all possible situations and create scenarios based on these information. While designing our models, we always keep in mind to fulfill these requirements and scenarios we came up with. At the end of this part, we provide Use Case Model and User Interface, because the interaction between application and user is very important, we explain them as easy and as understandable as we can.

In the second part, Analysis Model, we gave more detailed explanation of our application by using Object and Dynamic Models. This part is where we gave more specific design decisions about the application. In the Object Model, we first gave the Domain Lexicon which is very essential for the understandability of our application. We tried to clarify ambious words and we noted our key words. By specifying these crucial words, we became ready to identify our design in a more detailed manner. We provide Class Diagram. In the Dynamic Model part, we gave Start Chart and Sequence Diagrams which are also very essential for the design of our

implementation. We tried to be as specific as we can in order to have a solid design that we can implement.

To sum up, our purpose to writing this report was to design and implement our application easily in the future. We tried to make it as precise as possible in order to better understand our problem and solution domains.Only by thinking deeply about the process, we can came up with a better design and solution. This report will be our future reference and guide for our following process in the “Draw It!” project. That is why we understand the importance of this report and we took it seriously and tried our best.