Bilkent Univeristy

Computer Engineering

CS319 – Section 2

Object Oriented Software Engineering

Section TA: Gulden Olgun

Supervisor: Bora Güngören

Analysis Report: First Draft

Bubble Popper A Project By Group 2-I

Serhat Hakkı Akdag

Orkun Alpar

Mustafa Mert Aşkaroğlu

Faaiz Ul Haque

Table of Contents

Table of Contents

Bubble Popper – Analysis Report (CS319-2) – Group 2I – 10/7/2017

Bubble Popper – Analysis Report (CS319-2) – Group 2I – 10/7/2017

Analysis Report

1. Introduction

For our project we have decided to create a two player action game called Bubble Popper. The game will be entirely java-based, and will be playable on desktop only. This report will provide a comprehensive summary of the specifications of the game. We will introduce the topic by stating the basic rules and features involved in the gameplay. In specific, we will explain the overview, function requirements, non-functional requirements, pseudo functional requirements, use-case scenarios, dynamic models, class models, object models, user interface and screen mock-ups of the game. This report aims to represent a layout for the ongoing development of our project.

2. Overview

In our multiplayer game, players will work together to survive as bubbles drop down on them. Upon touching a bubble a player will lose a life. Players must try to evade the bubbles while attempting to pop the bubbles. Each player is equipped with a special weapon which can be used to pop the bubbles. The two players will share a number of three lives at the start of the game. Upon one

+player being hit by a bubble, one life will be deducted and the users will restart from their current level. Every third level the players will be granted with one extra life. Players can accumulate points by popping bubbles. Upon reaching a certain number of points a player's gun will automatically be upgraded at the end of the current level. The point system will be individual, so the players can compete in a sense to try and get better guns faster than their partner. As the levels progress the difficulty level will increase, as in the bubbles will grow in size and number. The object of the game is to complete all the levels of the game without losing all the given lives. A high-score system will be able to be viewed, in which the top three players with the most points will be listed.

2.1 Initialization Of The Game

Players can navigate through the game using their mouse, and select the different options on the starting screen: 'Play Game', 'Hiscores', 'Options', 'Controls', 'Rules', 'Credits', 'Exit'. Upon starting up the game the players will be given three lives and start with the basic gun. The points will start at 0, and each time the game is start up, the players have to begin from level 1.

2.2 Gameplay

During the game one keyboard will be sufficient for both players. Three keys will be used for each player: left navigation, right navigation and a shoot button. The pause button can be pressed by either user. Player 1's keys will include left arrow, right arrow, and the up arrow. Player 2's keys will include 'W', 'A', and 'D'. Players must move around the map while trying to avoid the balls as they fall in a constant motion. The balls motion will be altered as levels get harder using mirrors. The feature of the mirrors will also be changed depending on the difficulty of the level. The players will also attempt to pop the bubbles using their equipped weapons. Players' weapons will be upgraded at the end of reach around if they have reached a certain point threshold. Players can gain points by popping bubbles. Different colored bubbles will represent different speeds and different point awards each time they are popped. The players will share lives, and lose a life each time a player comes in contact with a bubble. The players must survive all the levels in order to complete the game.

2.3 Bubbles

There will be three different type of colours: red, yellow and green, which will be worth 10, 20 and 50 points respectively. The red bubbles will be the standard ones present in every level. The yellow ones will move a bit faster as compared to the red and will be more frequently seen as the game progresses in levels and difficulty. The green bubbles will only appear 10 times in the game, and be randomly placed in the levels.

2.4 Mirrors

To avoid a monotonous behavior of the balls motion, mirrors will be placed in the levels to allow the balls to divert in their paths. This will create a challenging effect in the game and allow a more enjoyable experience. The number of mirrors will increase as the levels get harder. Their positioning for the most part will be

randomly placed. The size of the mirrors will remain constant. The mirrors will have different types of features. There will be three type of mirrors: transparent, opaque and white. The transparent mirrors will allow the ball to pass through directly, in different random speeds. This will confuse the users and they may expect the ball to divert in path when it will not. The opaque mirrors are unpassable and allow the ball to rebound off the mirror and change its course. The white mirrors will always be placed in pairs. When the ball hits one white mirror it will come out of the second one. The mirrors may be stationary, or moving depending on the level difficulty to make it more challenging.

2.5 Weapons

In the first level users will initially start out with the basic arrow and bow to shoot the bubbles. If an individual player reaches 100, and 300 points their guns will be upgraded. When a player reaches 100 points their arrows speeds will be increased allowing them to shoot faster and pop the bubbles more quickly. When a player reaches 300 points, the arrow will shoot two arrows at a time parallel to each other. This will allow an easier target on the bubbles, and also allow the possibility of popping multiple bubbles at a time. Since the point system will be individual, the players will only upgrade weapons upon reaching the point thresholds on their own.

2.6 Lives

In this game the two players will share a set number of lives. The players will initially start with a total of three lives. If one player gets hit by a ball, both players will suffer and will be reduced by one life, while they have to restart the current level. If the players lose all their lives, they have to start from level 1. After every three levels completed the players will gain one extra life.

2.7 Levels

The game will consist of ten levels and will increase in difficulty as the game progresses. The mirrors, guns, bubbles will all change depending on the level number. The number of bubbles will also increase as the levels get harder, and their types will change (red to yellow), more frequently. After the completion of each level the user will be prompted to a screen in which they can take a break and view their current scores, lives and type of gun. Their guns will also upgrade in this time period depending on their number of points. Users can click 'continue

playing', a menu which will prompt them to the upcoming level. If the players complete all ten levels they wRill win the game.

2.8 Points

Players can accumulate points by popping bubbles. Unlike the joint life system, the point system will be individual. This will allow competitive play in the game, where players will try their best to pop more bubbles so they can increase their individual scores. If a player reaches a certain bench mark of points they can upgrade their weapons at the end of every round. To allow for a more enjoyable experience, if the players complete a level without losing a single life they will both be awarded with 50 points each.

3. Requirement Specifications

3.1 Functional Requirements

3.1.1 Play Game

This option will automatically direct the two players to the first level of Bubble Popper. The players will start with the basic guns and a shared total of three lives. The players' individual points will initialize at 0. The bubbles will start falling after a delay of three seconds for the players to get ready to start playing the game using the keyboard.

3.1.2 Controls

This menu will display the navigation controls and shooting buttons as displayed on the keyboard for each player respectively.

3.1.3 Rules

A summary of the gameplay and basic rules will be stated. Different type of bubbles and their points will be shown with the aid of pictures. The weapon system and life system will also be explained briefly. This will allow the players to get a quick idea of how the game works before beginning.

3.1.4 Settings

Players can adjust the sound and music options. There will be two different types of sound, one will be the background music of the game which will be constant throughout gameplay and there will be additional sound effects produced by the shooting and popping actions that will occur in the game.

3.1.5 Pause button

Players can press on the pause button in the top right corner to take a break or save their progress and continue later. Additionally the 'P' character on the keyboard can be used to pause the game.

3.1.6 High scores

A list of the current top three players will be constantly and automatically updated in the high scores menu. The high scores will be of an individual player, and not a joint score of the two players. This feature introduces a more competitive experience for the players.

3.1.6 Credits

The developers of the game will be listed here along with their contact information. Players can contact the creators to learn about the development and process of creating the game, and also provide positive or negative feedback.

3.1.7 Exit

Immediately closes the game and redirects out of the window.

3.2 Non-Functional Requirements

3.2.1 Response timings

In the start of every level there should be a minor three second delay before the bubbles motion begins, in order to give the players a chance to dodge the bubbles and ensure they do not immediately fall down upon them. There also should be a delay between when the gun is shot and the bubble is popped. There will be a visible arrow so the players can learn the current speed of their guns. This will create a more

challenging experience and allow players to develop a skill for being accurate in the game.

3.2.2 Game Performance And Graphics

Although specific, the graphics of the game will be kept simple to prevent lags or glitches that could occur in the game. The navigation of the players and shooting will all be created in high performance to ensure the smoothness of the gameplay. The bubbles popping will have a simple sound effect and graphic splash accompanied with it. The weapons will also be kept simple but clearly visible to distinguish between types of upgrades. The mirrors will be simple wall structures with different colors. There will be minor sound effects with shooting, and also background music throughout the game play.

3.2.3 User Friendly Interface

The menus in the initial screen will be natural and easy to understand. The game will be appealing using the suitable colors that attract the eye. The rules and other explanations will address the players with simple and coherent English. The other icons during the gameplay will be created with the utmost simplicity for the users to understand with convenience, as in the number of lives, and scoring systems.

3.2.4 Extendibility

Extendibility is an important tool used in the software engineering and programming world. The game will be incorporated with both concepts of extendibility and reusability of the game. This will allow for updates that could result in future and more improved versions of the game, while taking account the user feedback provided to the game developers as seen in the credits menu.
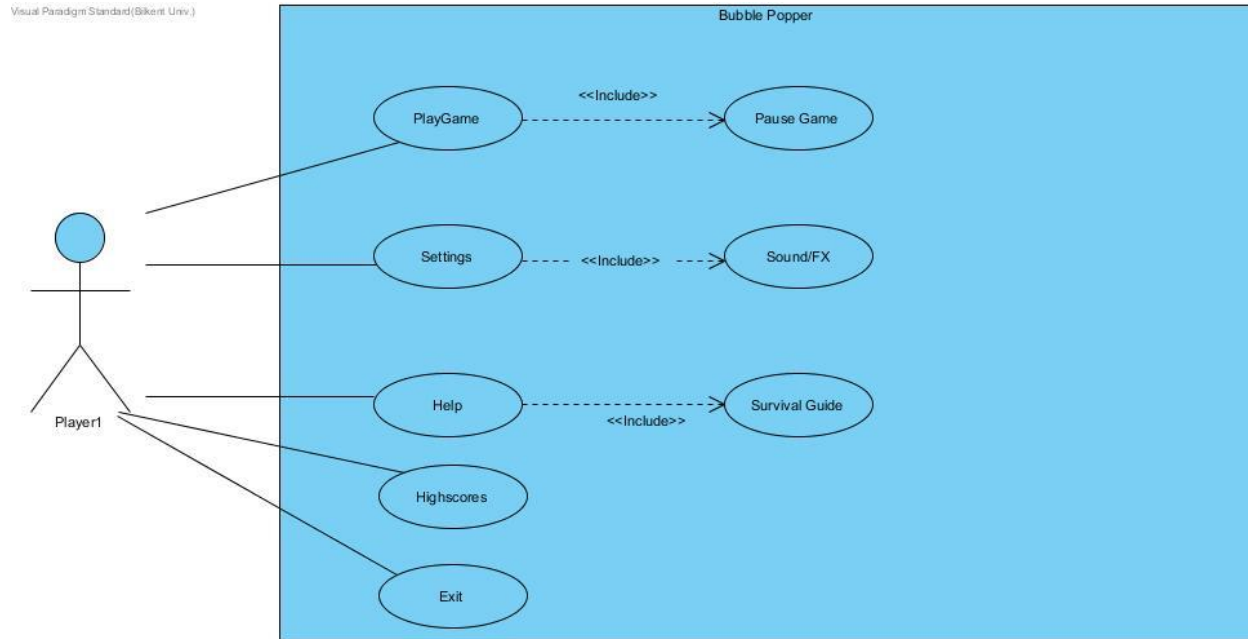
3.3 Pseudo Requirements

The entire game will be java-based. The coding of the classes will be implemented in java and the graphics will be created using the aid of java FX. The game will be playable on a desktop.

4. System Model

4.1 Use Case Model

Below is the use-case model in which the player is shown and its relations with the functionalities of the game.

4.1.1 Play Game

Participating actors: Players

Stakeholders and interests: Player wishes to play the game

Entry condition: Player had the ability to launch the game using his/her mouse and is now on the main-menu page, with two players ready to play the game.

Exit condition(s): 1. Players have successfully passed all levels and completed the game.

2. Players lose all their lives before being able to finish the game.

3. Players return to the main-menu after pausing the game.

Main flow of events: 1. Game starts from level 1

2. System constructs the game by providing players with lives and loads the maps.

3. Players finish all the levels successfully, by popping bubbles in every level and maintaining at least 1 life.

4. Scores are displayed of the players, and they will enter their names

5. System checks high scores, and compares with current scores. If the scores are higher than the top 5, they will be replaced.

6. Returns players to the main menu

Alternative flow of events: 1. Players lose all their lives

2. Players exit the game from pause menu

4.1.2 Settings

Participating actors: Players

Stakeholders and interests: Player wishes to alter the sound of the game

Entry condition: Player had the ability to launch the game using his/her mouse and is now on the main-menu page.

Exit condition(s): Return to main menu

Main flow of events: 1. Player loads the game.

2. Player is directed to main-menu page.

3. Player selects change settings.

4. Player adjusts volume and fx values

5. System updates the player's inputs.

Alternative flow of events: No settings change, i.e default settings chosen. Player can also be redirected to the main menu after losing or completing the game.

4.1.3 Help

Participating actors: Players

Stakeholders and interests: Player wishes to learn how to play the game

Entry condition: Player had the ability to launch the game using his/her mouse and is now on the main-menu page.

Exit condition(s): Return to main menu

Main flow of events: 1. Player loads the game.

2. Player is directed to main-menu page.

3. Player selects help

4. Player(s) reads and views how to play the game and learns the default controls

5. Once understood, player returns to main menu

Alternative flow of events: After step 3, players changes his/her controls. **Player can also be redirected to main page after losing or completing the game.**

4.1.4 Highscores

Participating actors: Players

Stakeholders and interests: Player wishes to view the highscores

Entry condition: Player had the ability to launch the game using his/her mouse and is now on the main-menu page.

Exit condition(s): Returns to main menu

Main flow of events: 1. Player loads the game.

2. Player is directed to main-menu page.

3. Player selects highscores

4. Player views highscores

5. Player returns to main menu

Alternative flow of events: No high scores able to be displayed since no one has played the game yet.

4.1.5 Exit

Participating actors: Players

Stakeholders and interests: Player wishes to exit the game

Entry condition: Player had the ability to launch the game using his/her mouse and is now on the main-menu page.

Exit condition(s): Game must be open again

Main flow of events: 1. Player loads the game.

2. Player is directed to main-menu page.

3. Player selects exit

4. Window closes, game ends

Alternative flow of events: Players loses or finishes the game and is redirected to main-menu from where they exit the game.

4.2 Sequence Diagrams

4.2.1 Main Menu Navigation

Scenario: Player1 opens the game by clicking the game icon from computer. Then main menu appears with options "New Game", "Settings", "Help", "Credits", and "High Scores". If clicked to "Settings", program takes the changes that player had, changes them and saves, then by "Return to Main Menu" button goes back to main menu. If clicked to "Help", rules are shown and by clicking "Return to Main Menu" button navigates back to main menu. If clicked to "Credits", developers of the game and their contact mail addresses shown and by clicking "Return to Main Menu" button navigates back to main menu. If clicked to "High Scores", top 5 scores achieved by previous runs shown and by clicking "Return to Main Menu" button navigates back to main menu. If clicked to "Start New Game", players starts playing new game.
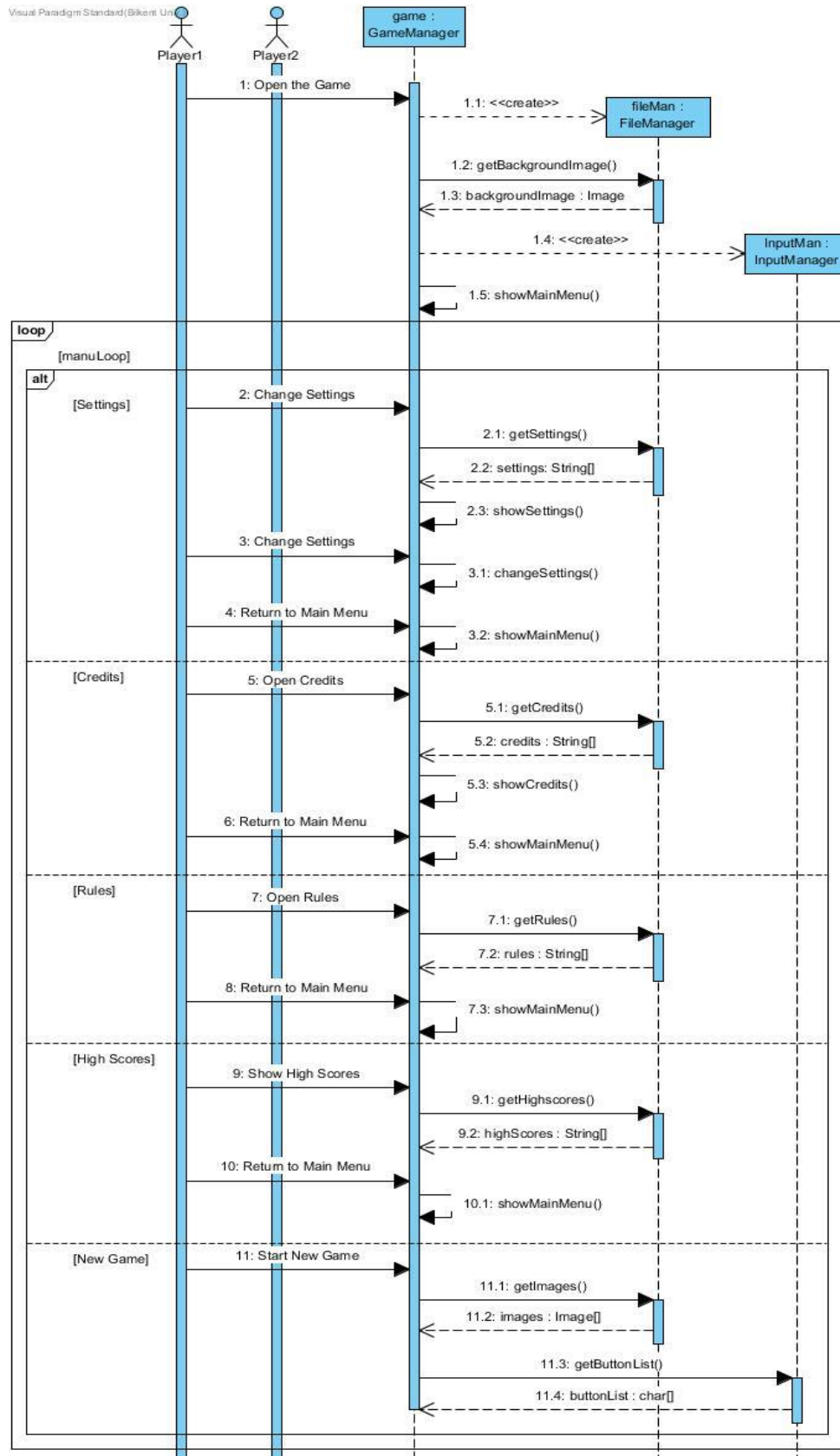
Figure 2 – Sequence Diagram of Main Menu Navigations

4.2.2 Play Game

Scenario: Assuming that new game started by performing steps on the previous sequence and pressed "Start Game Button", characters for both players with starting weapons, bubbles, and mirrors are created. Every time game loop is iterated, system updates positions of characters, bubbles, and bullets. If bubbles hit to walls or mirrors, they are reflected. Then system checks for collisions. If bullet-bubble collision happens, bubble popped and points of player hitting that bubble updated. If bubble-player collision happens, lives are updated accordingly. At the end of each loop, system checks if the round ended by checking remaining lives and bubbles. If round is not over, loop actions are performed again.
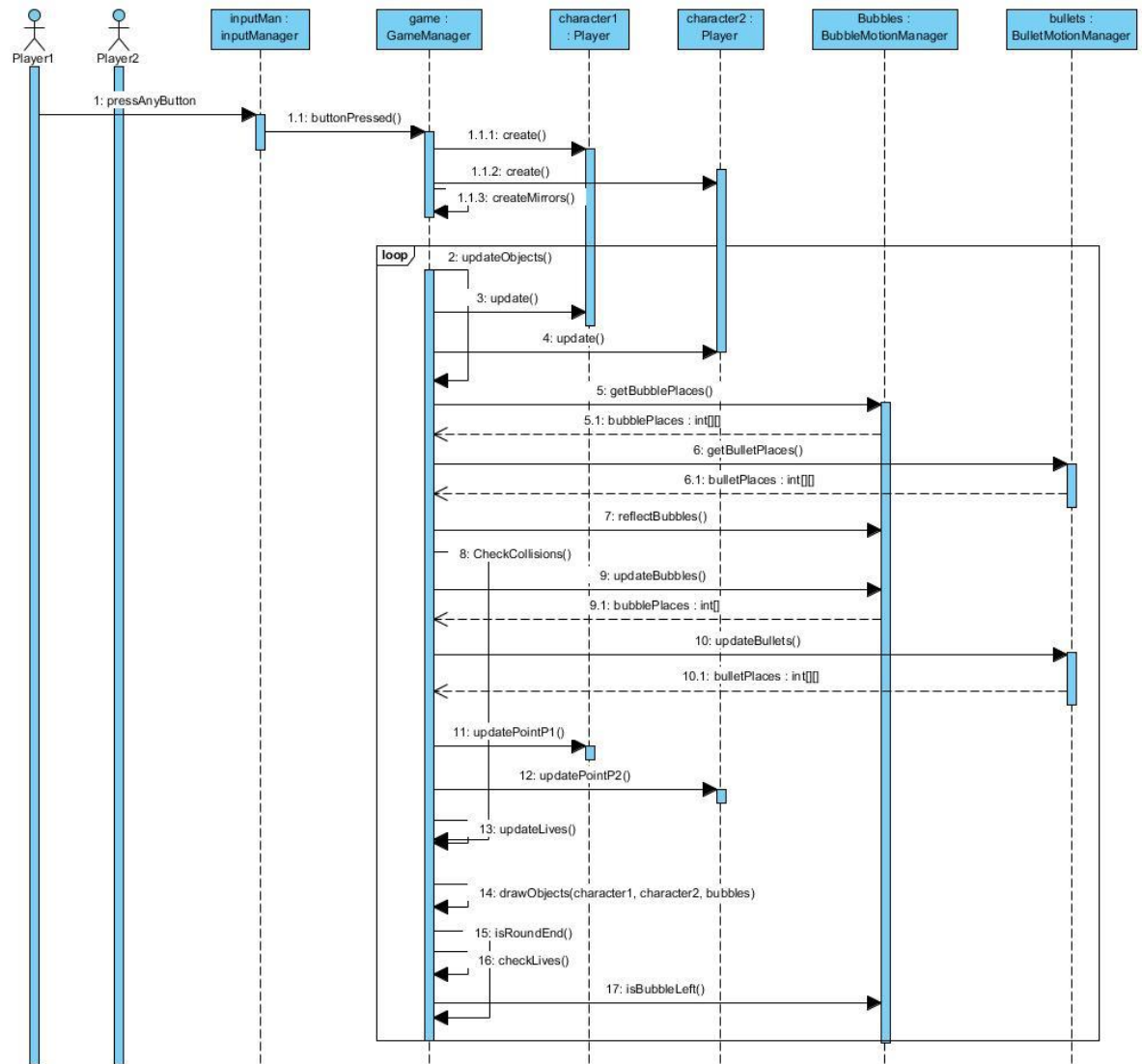
Figure 3 – Sequence Diagram of Playing Game

Bubble Popper – Analysis Report (CS319-2) – Group 2I – 10/7/2017

4.2.3 End of Rounds

Scenario: At the end of each round systems waits for player1 to press a button before starting the next round. When pressed to a button, system checks if it is the last round or not, if so total point is calculated. If total points are eligible to be at top 5 scores, player1 enters a team name, and score is saved to high scores. If it is not end of the game, weapon manager updates the weapons of the player according to their points and next round starts.
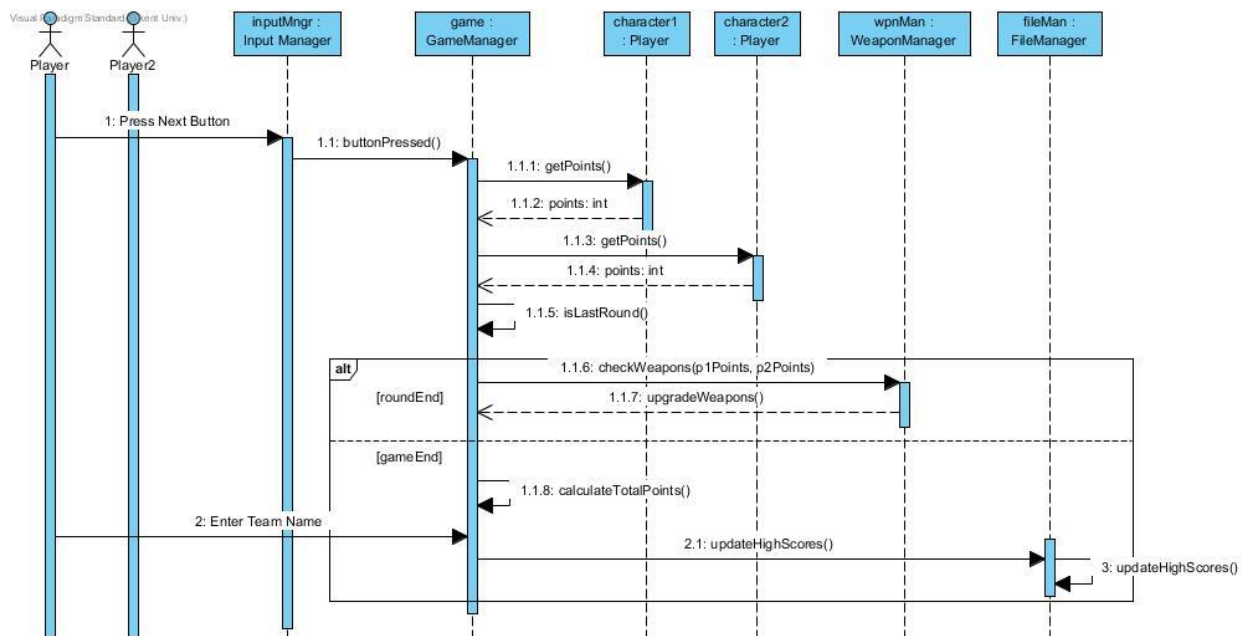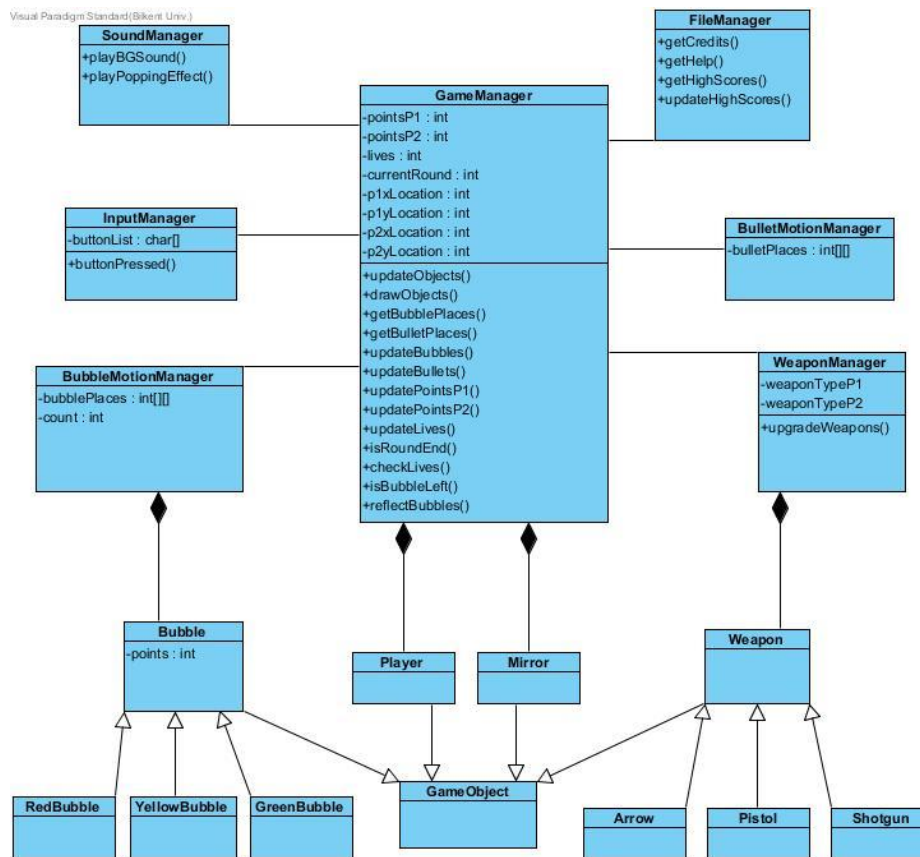

Figure 4 – Sequence of Actions at the End of Rounds

Bubble Popper – Analysis Report (CS319-2) – Group 2I – 10/7/2017

4.3 Object and class diagram

SoundManager
+playBGSound()
+playPoppingEffect()

FileManager
+getCredits()
+getHelp()
+getHighScores()
+updateHighScores()

GameManager
-pointsP1 : int
-pointsP2 : int
-lives : int
-currentRound : int
-p1xLocation : int
-p1yLocation : int
-p2xLocation : int
-p2yLocation : int
+updateObjects()
+drawObjects()
+getBubblePlaces()
+getBulletPlaces()
+updateBubbles()
+updateBullets()
+updatePointsP1()
+updatePointsP2()
+updateLives()
+isRoundEnd()
+checkLives()
+isBubbleLeft()
+reflectBubbles()

InputManager
-buttonList : char[]
+buttonPressed()

BulletMotionManager
-bulletPlaces : int[][]

WeaponManager
-weaponTypeP1
-weaponTypeP2
+upgradeWeapons()

BubbleMotionManager
-bubblePlaces : int[][]
-count : int

Bubble
-points : int

Player

Mirror

Weapon

RedBubble

YellowBubble

GreenBubble

GameObject

Arrow

Pistol

Shotgun

This diagram shows the object model of Bubble Popper. All classes and objects are shown in this diagram and their corresponding relationships. There are a total of 11 classes.

Game Manager: This class manages all other classes and is considered as the main backbone of the game. It contains all the methods required to function the basic actions of the game. It also contains some basic variables like lives and points for the users. It has the locations of the two players. The Game Manager is in charge of the graphics and the creation and deletion of the bubbles. It constantly checks for any changes that alter in the game such as losing a life, finishing a level, and even winning the game. A score manager is also incorporated in this class and updates the scores automatically upon popping bubbles.

Sound Manager: Controls the background sound of the game and FX volume of the bubbles popping and guns shooting.

File Manager: Manages the main menu of the game where methods such as updating the high scores, showing the credits or rules of the game are needed.

Input Manager: The controls of the game using mouse and keyboard are managed by this class.

Bubble Motion Manager: The paths that the bubbles takes will be handled by this class. In the case of a diversion in the path when colliding with a mirror, this class will manage it. It will also count the number of bubbles at a given time and their corresponding locations in x and y coordinates. It will control the collision of bubbles with bullets and players as well.

Bullet Motion Manager: Stores x and y coordinates of the bullets being shot by different players

Weapon Manager: Stores different types of weapons, for different players. Also has a method that upgrades the players weapons based off their points handled in the GameManager class at the end of each level.

Bubble: There are three different types of bubbles, and this class will store the different names, colours and their points. Also the frequency at which they appear will be controlled from this class.

Player: Stores the two player's names, guns and points.

Mirror: Different types of mirrors and their function and sizes are stored here.

Weapon: Each player has one of three weapons which is upgraded as they go on: arrow, pistol, and shotgun.

4.4 Activity Diagram
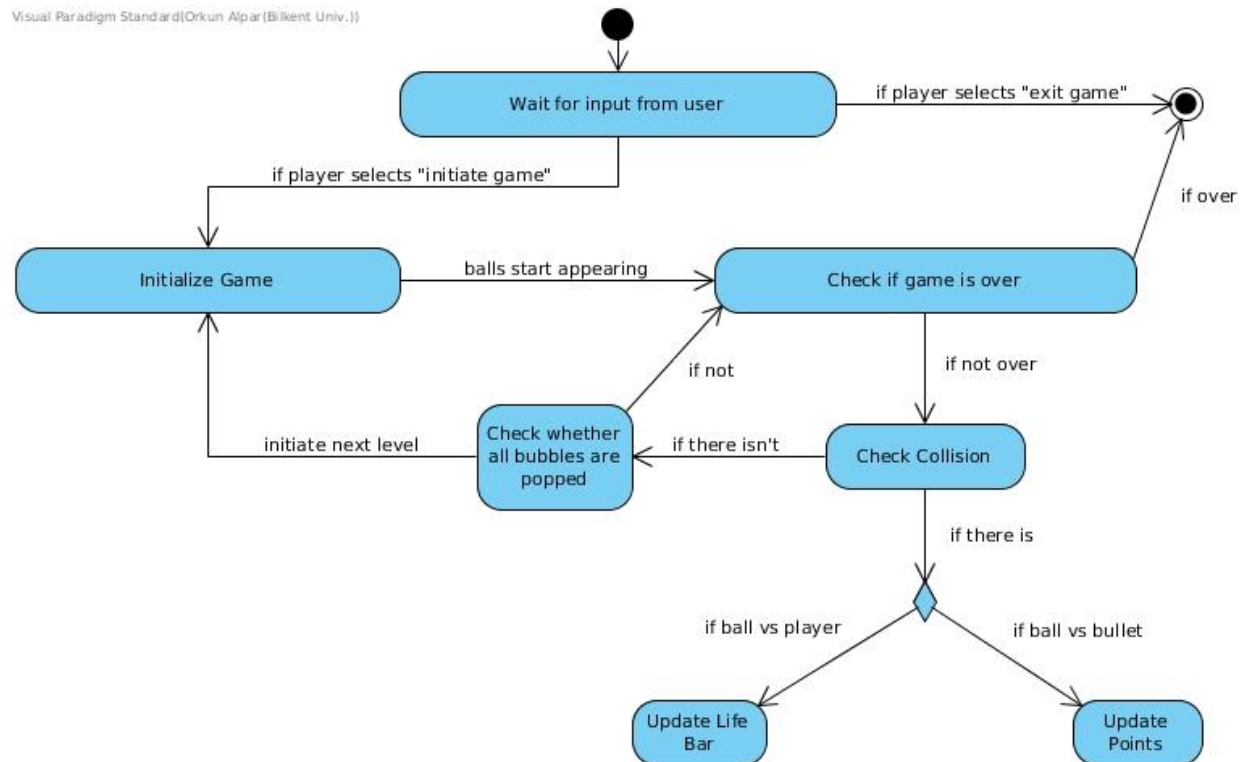
This figure shows the activity of the main events that can occur when the game is played. Initially the system will load the game and direct the user to the main menu from where it waits for the users input through the mouse. The user can select the high-scores, credits, help options and view the necessary information and return to the main-menu. Other than this the user can exit the game directly from main-menu. If a user clicks playgame the game will begin on the first level. The bubbles start appearing and the system checks if the game is over. In this section there are two cases, either the user loses all lives or the users are on the last level and complete it in which case the game is completed. In either case the user will be redirected to the main menu. If it's not over, the system checks for collisions, in this case the lives are updated or the points are updating depending on the type of collision. If there is no collision the system checks for any remaining bubbles, if there are remaining bubbles the cycle is repeated, if not then the next level begins.

5. User Interface

5.1 Navigational Path

Bubble Popper – Analysis Report (CS319-2) – Group 2I – 10/7/2017
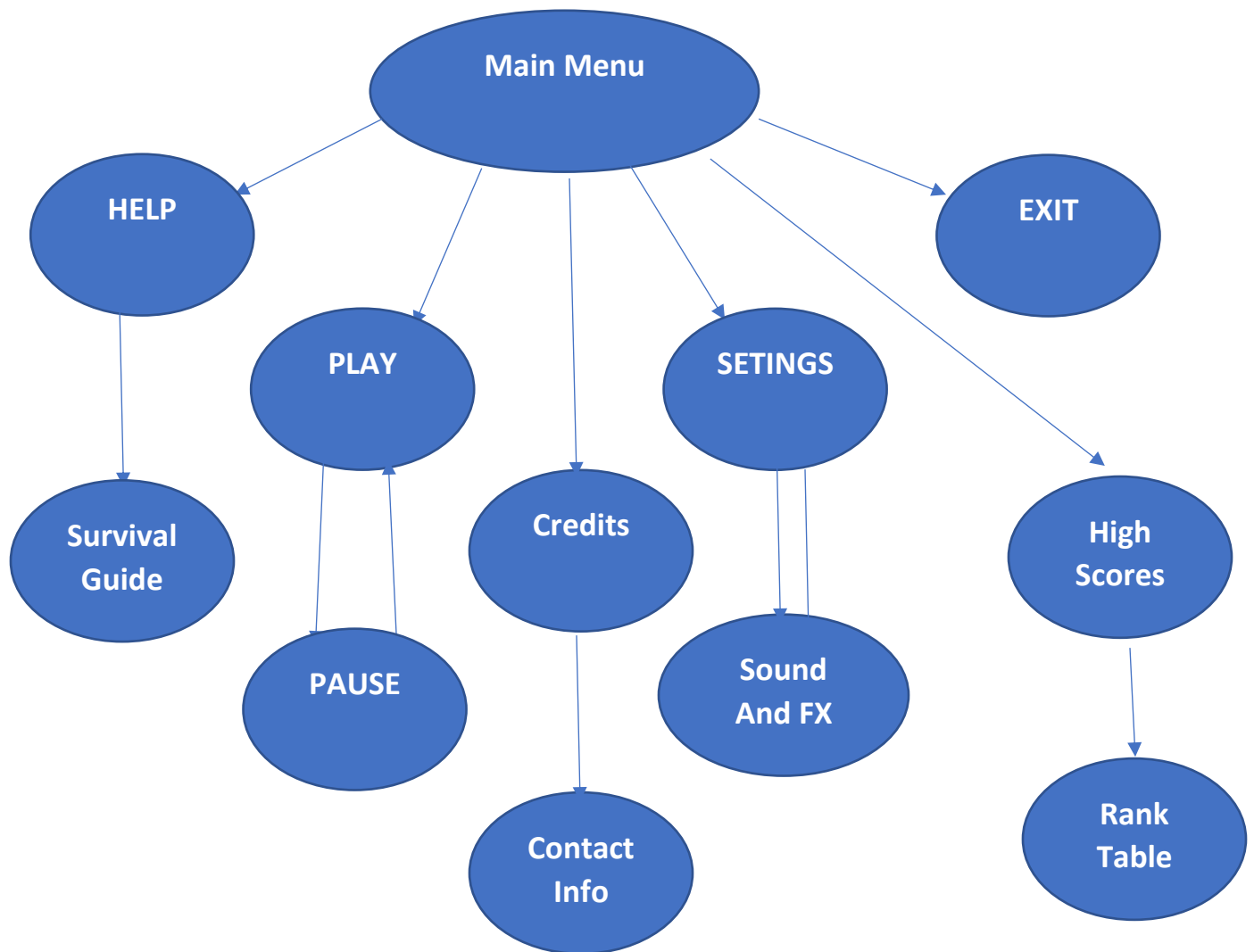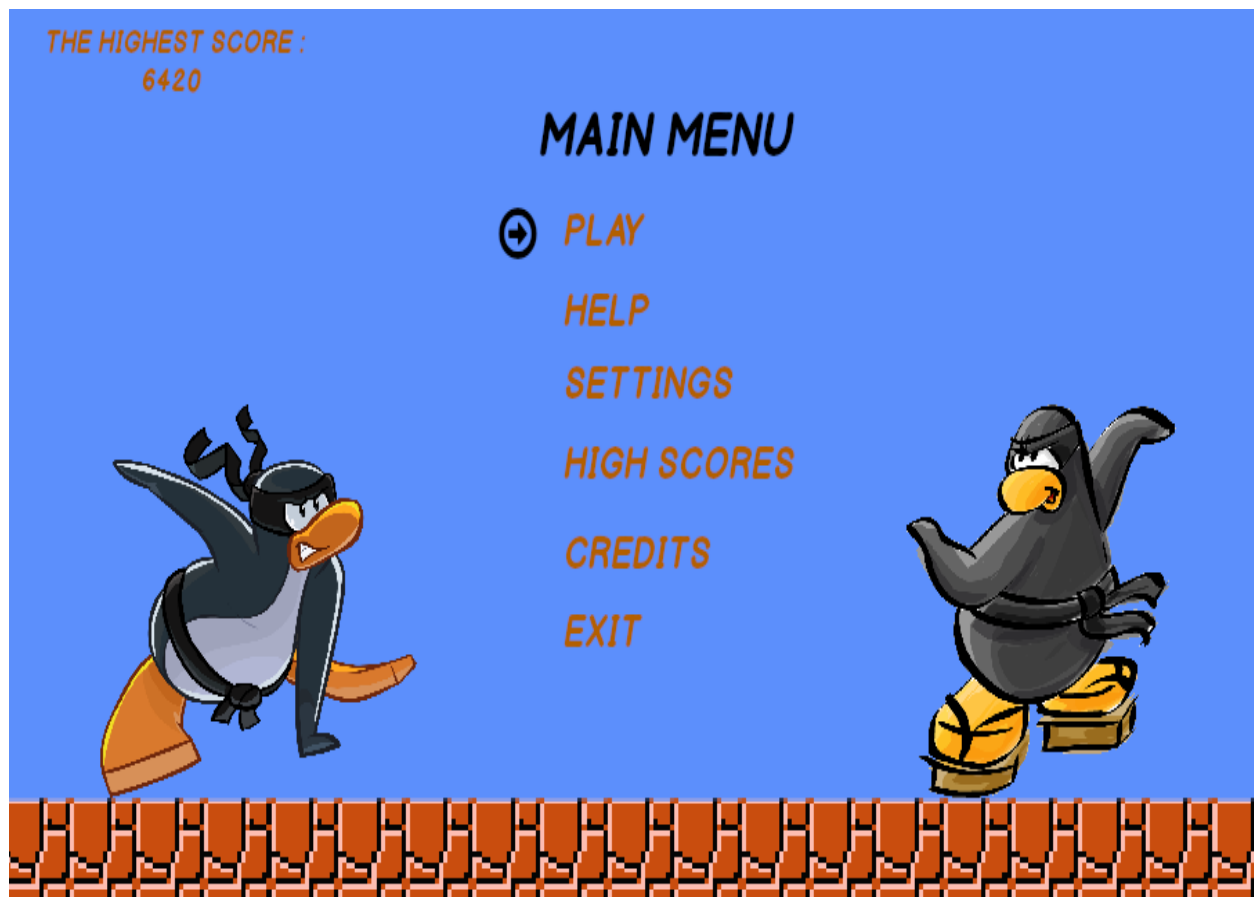
# 5.2 Screen Mock-Ups

5.2.1 Main Menu

Main menu is an interface with some features which are shown when the user opens Bubble Popper. This interface is designed to provide users with the ability to arrange their game settings and see how the game is played.

The options are listed below in the picture.

5.2.2 Settings

Settings button directs users to an interface which enable users to arrange their game settings according to what kind of a game experience they want. There are two sound types with three options which can be decided by users in this part. Whereas the first option asks users to determine volume of background music, second one asks about volume of effects. Users have three options for all parts which are mute, medium-level and high-level sound volume.
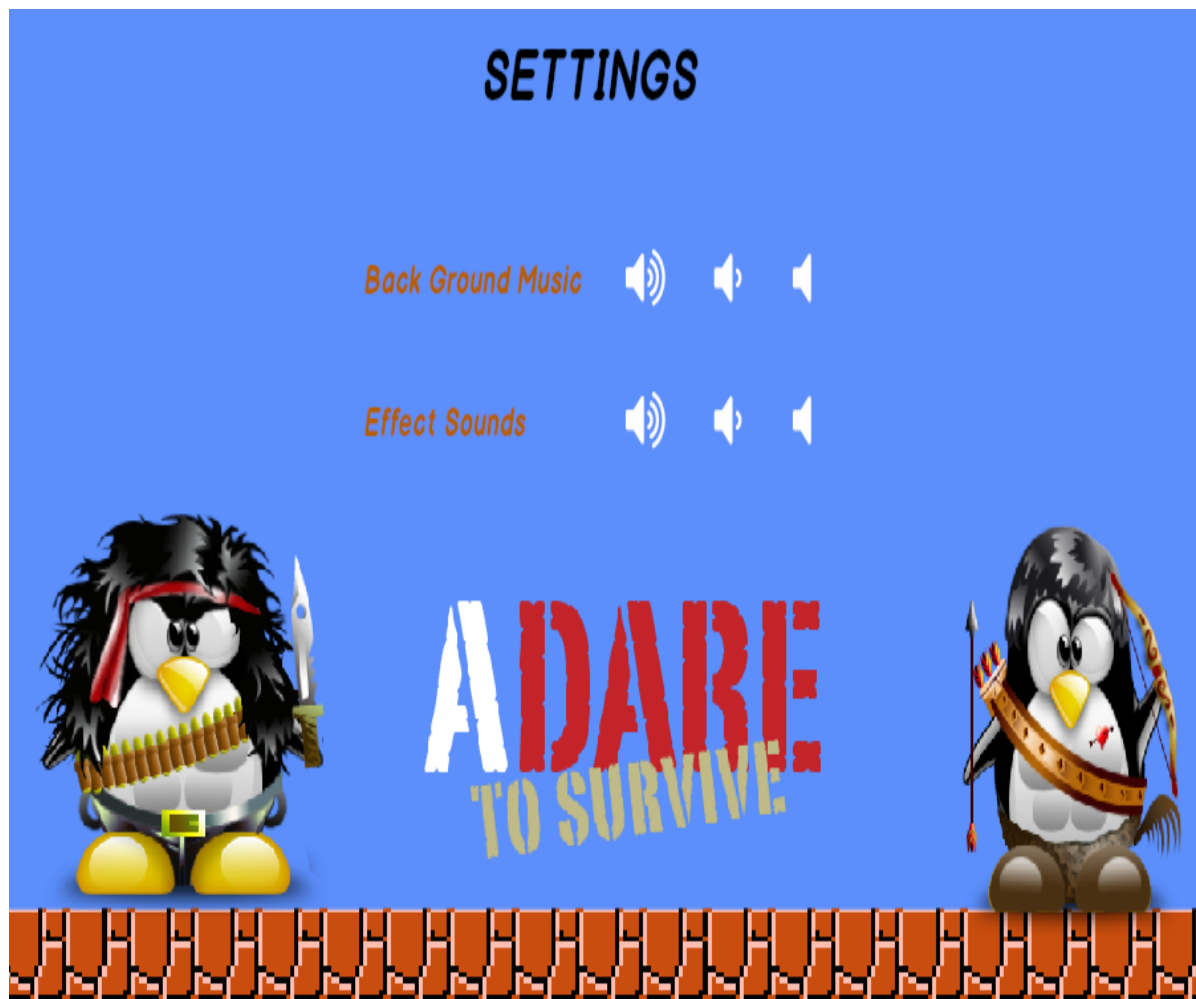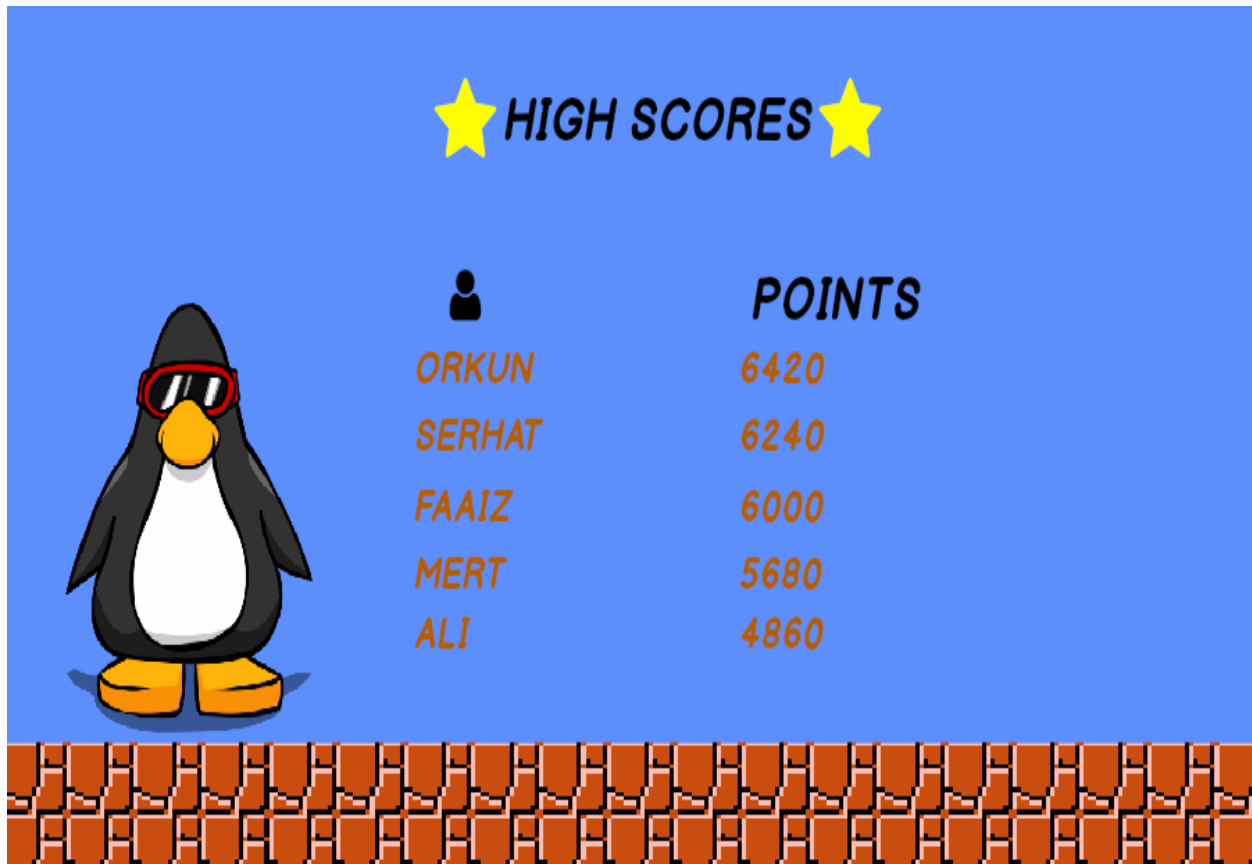
5.2.3 HELP

Help button directs users to an interface which provide users to be informed about how the game is played and which player (1 or 2) uses which control buttons in order to move the characters and pop the bubbles.
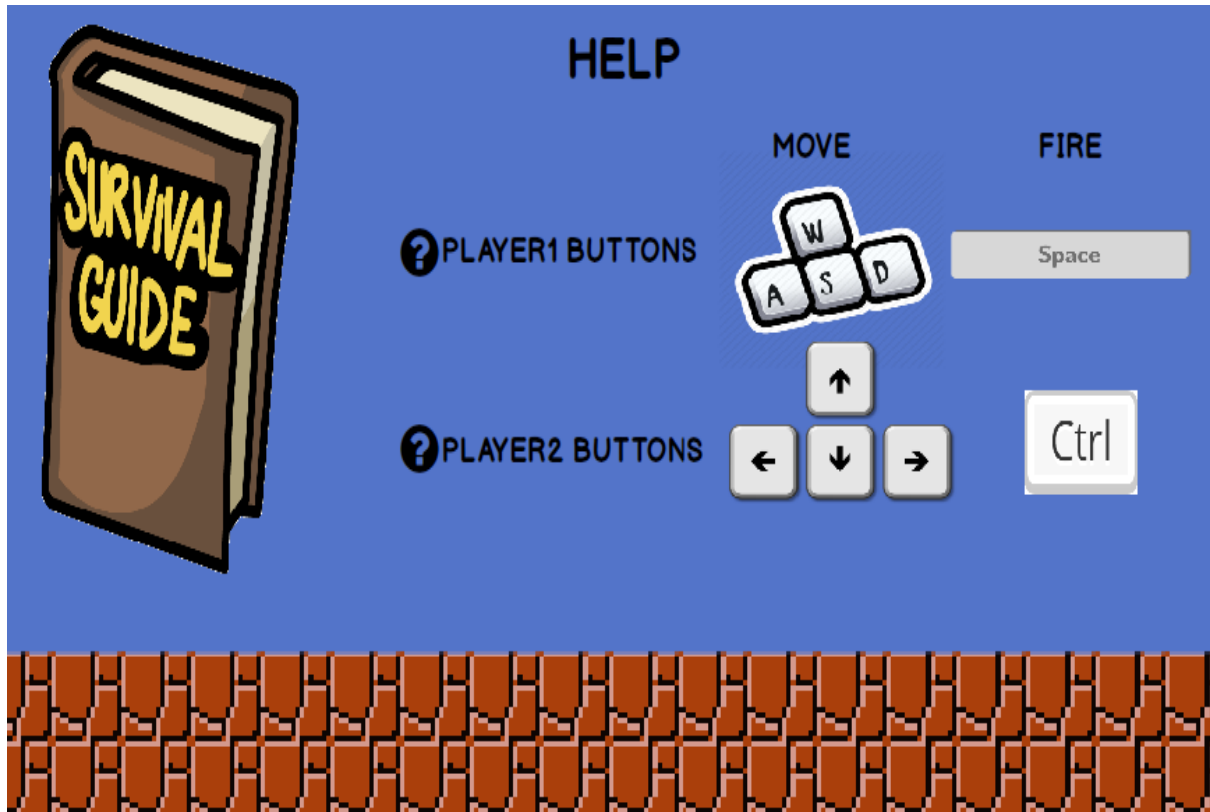
5.2.4 High Scores

High Scores button direct users to an interface which displays high scores achieved by players with their names.
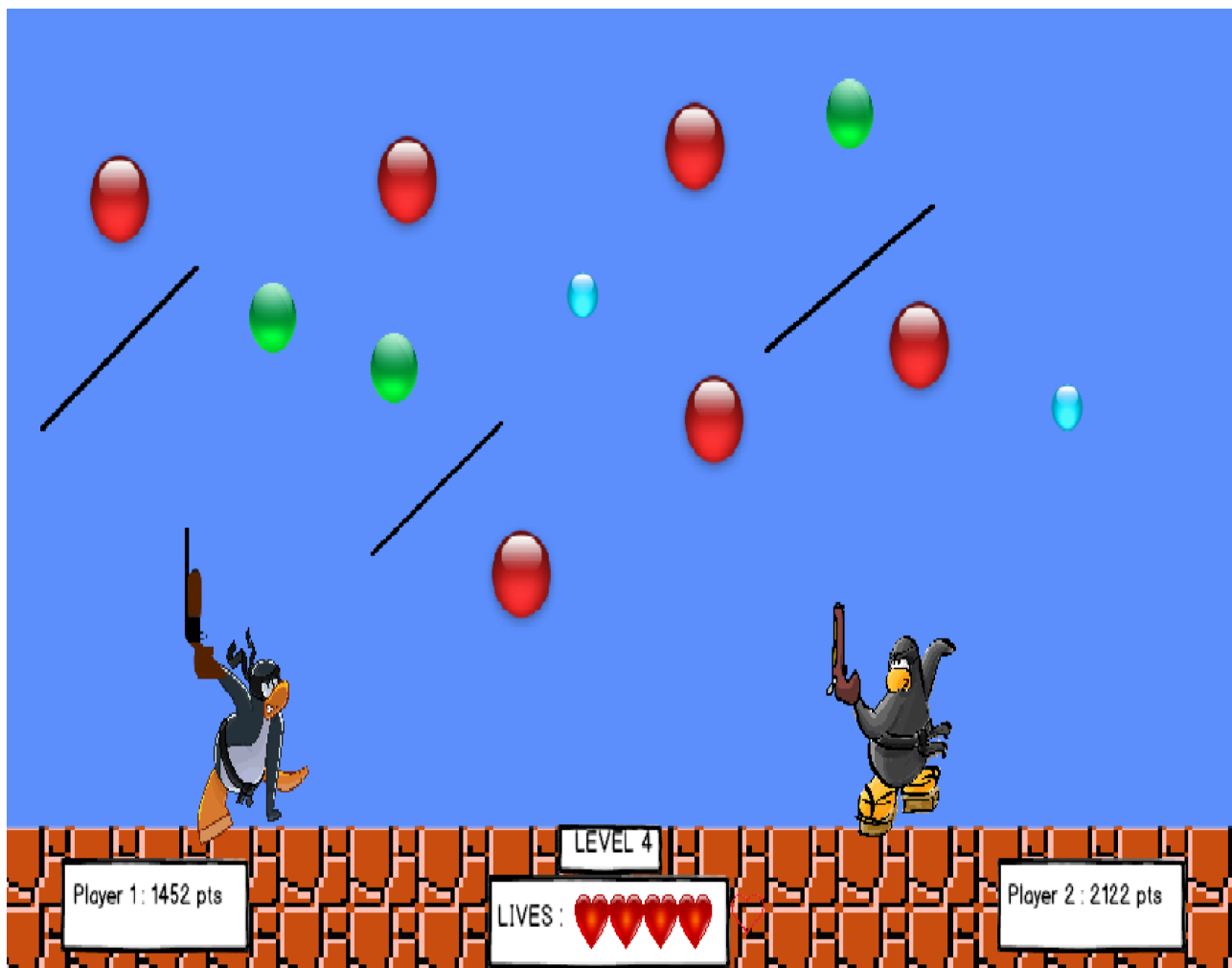
5.2.5 Credits

Credits is an interface which enables users to see the contact information of the developers in order for advisory, questions or suggestions about the game to improve overall performance, and adding future updates for newer versions.

5.2.6 PLAY

Play is a button which starts the game with current game settings, you can see a game mock up below.

5.2.7 PAUSE

Pause button will be available after game is started, it will be activated with pressing of button 'P' by player 1 at any time. Furthermore, when 'P' button is pressed again the game will continue from current position. A mock up is demonstrated below for paused game. From the pause button users can also exit the game or return to main menu
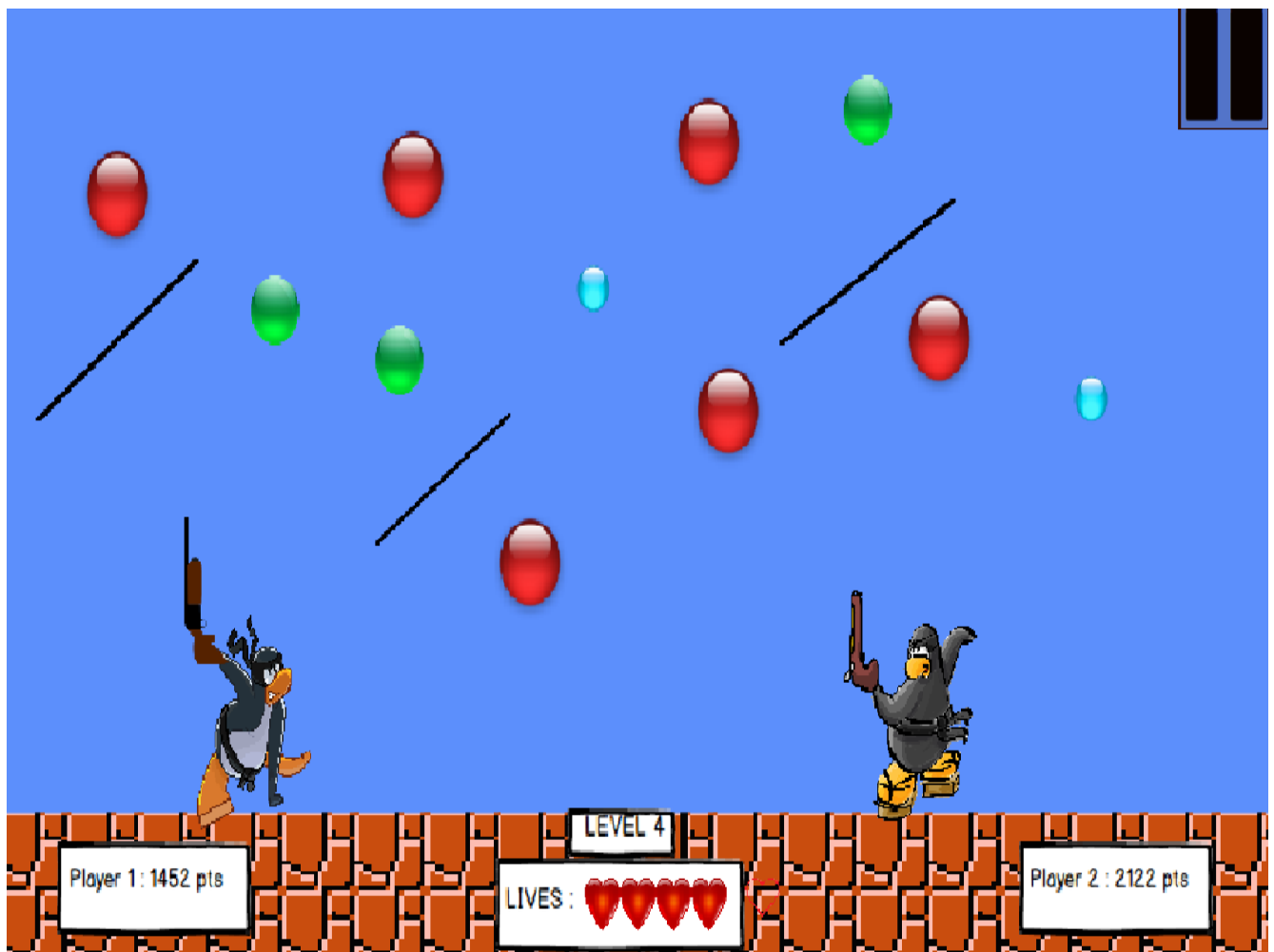
6. Conclusion

In this analysis report for our game called bubble shooter, we have aimed to provide extensive information regarding the implementation of the game. In section 1 and 2 we have introduced the game and provided an overview of the features and specifications that a player or future developer should know regarding the game. In section 3 the functional and non-functional requirements are listed and explained in detail. Section four provides diagrams related to the background and how the game is constructed and will play out by the system. These include sequence diagrams, activity diagrams, use case diagrams and class and object diagrams. In the final section the reader can see the navigational path and the possibilities players can take when operating the game. Screen mock ups are also provided for the reader to have an idea of how the game will look and turn out as a final product. This report additionally is a map for the creators of the app to look back and refer to as aid for the overall implementation. For these reasons the report is created with a lot of detail, to further ensure the success of the implementation of our project. The game idea is a unique one and derived from our teams own independent imagination.