CS 319 - Object-Oriented Software Engineering

Analysis Report Draft

Color Shooter : The Spectrum Adventurer

Group 3-F

Hasan Selim Yağcı

İrem Ural

Erdem Adaçal

Alper Mehmet Özdemir

Table of Contents

1.Introduction 4

2. Overview 4

2.1. Gameplay 5

2.2. Level Design 5

2.3. Tiles 5

2.4. Enemy Types 6

2.5. Power-ups 6

3. Requirement Specificaiton 7

3.1 Functional Requirements 7

3.1.1. Play Game 7

3.1.2. Change Settings 7

3.1.3. Pause Game 8

3.1.4. View Help 8

3.1.5. View Credits 8

3.2Non-Functional Requirements 8

3.2.1. Game Performance 8

3.2.2. User-Friendly Interface 9

3.2.3. Supportability 9

3.2.4. Extendibility 9

4.System Model 10

4.1. Use Case Model 10

4.1.1. ViewCredits 11

4.1.2. Play Game 12

4.1.3. Change Settings 14

4.1.4. View Help 15

4.1.5. Pause Game 16

4.1.6. New Game (doneIrem) 16

4.1.7. Continue Game

4.1.8. Exit Game

4.2. Dynamic Models 18

4.2.1. Sequence Diagrams(selim..working on first sequence) 18

4.2.1.1. Start Game (Scenarios done) 18

4.2.1.2. Power up Management (Scenario Done) 20

4.2.1.3. Pause Game 21

4.2.1.4. Death of Player 23

4.2.1.5. Kill an Enemy 24

4.2.2. Activity Diagram 25

4.2.2.1. GamePlay Activity Diagram 25

4.2.2.2. Game Class Diagram (TODO) 27

5. User Interface 29

5.1. Navigational Path (done) 29

5.2. Screen Mock-ups 30

5.2.1. Main Menu 30

5.2.2. Pause Menu(TODO) 34

5.2.3. Power Ups(TODO)(Sayfa 7 de var zaten) 35

6. Important Decisions in overall Analysis(TODO) 35

7. Conclusion(TODO) 35

8. References 37

1. Introduction

Color Shooter: The Spectrum Adventurer is a 2D platform-shooter single player game. The player will jump from platform to platform, destroy various enemies and overcome several obstacles during its journey to reach a goal. There will be a colour system in the game. It will restrict the player to be able todamage only the enemies which has the same color as the bullets the player has fired. Also there will be some obstacles such as walls and the player should change his/her color to the color of the wall to be able to pass through them. There will be different power-ups on the platform as well. Additionally, there will be two difficulty levels, in hard mode there will be a special jump system that will be handled by the game manager, therefore the player will not deal with jumping. Every frame the game manager will check if the player is not on platform and if it was on a platform in the previous frame. If that is the case the Game Manager will make the player jump automatically. The player will have full control of his movements in the air, meaning he/she can change his/her direction. Although it seems to ease the players side, sometimes it creates difficulties. The color system and jumping feature are inspired from the following games respectively:

- Call Me Young Kaleido (http://armorgames.com/call-me-young-kaleido-game/18162?tag-referral=platform)
- Jumphobia (http://armorgames.com/jumphobia-game/18009?via-search=1)

The Spectrum Adventurer will have different levels and each level will be designed to teach different features of the game. When the player reaches the goal specified in each level, next level will be unlocked.

2. Overview

Spectrum Adventurer is a platform-shooter game with a unique jumping and color system . It is quite easy to learn to play but also becomes challenging in later levels. The scaling difficulty will make the game quite enjoyable and possibly addictive. Basicly you are a trying to navigate through enemies and obstacles to reach the goal in a crystal themed world. You will use your gaming skills to try to understand the correct route to follow to get to the goal in each level and use your reaction skills/speed to overcome the various enemies and obstacles

without dying. Reaching the goal in every level will unlock the next level allowing you to play a more challenging and fun level. The game will keep track of how many times you died ("restart level"s included) and tell you how many lives it took for the player to finish the game at the end.

2.1. Gameplay

The player will use the keyboard to play the game and use the mouse to navigate and use the menu. The player will move left and right with the arrowkeys, shoot with the "Spacebar" and change colors with the "Z" (blue), "X" (red) and "C" (green) keys. The player can pause the game and bring up the pause menu with the "Enter" or "P" keys. The player can select option from the pause menu with the mouse and the left mouse button. The player can resume the game by either pressing"P" key again or clicking the resume button on the pause menu. The player can restart the current level by pressing the "R" key

2.2. Level Design

There are 10 different levels in the first version of Spectrum Adventurer. The first 4 levels will be organized to teach the player the core mechanics of the game and adapt to its differences from similar games. The following 6 levels will be in difficulty order and the player will have to overcome more challenging enemies and will have less time to react to the dangers of the game environment. Especially in the last 2 levels the player will have to push the limits of his/her abilities as the level will have very little lenience towards wrong actions from the player.

2.3. Tiles

There will be different platform types in its game (as influenced by "Jumphobia") and each of them will be affected by the color system (similar to "Call Me Young Kaleido"). There will be 4 different colors of platforms: black, blue, red, green. Black will be the default platform color and will always behave as an obstacle to the player. Blue, red and green will behave as

obstacles when the player is a different color then the platform, otherwise the player will be able to pass through the platform or a wall.The types of tiles are as the following:

- **Standard Tiles:** Black tiles that the player will perform normal collision with.
- **Colored Tiles:** Red, green or blue tiles that will allow the player to pass through them if the player shares the same color as the tile.
- **Fading Tiles:** Tiles that will start to fade and get destroyed after the player touches them. Can be black or colored. Will not fade if the player is the same color as the tile.
- **Moving Tiles:** Tiles that will move left to right or up and down between two sets of predetermined coordinates. Can be standard or colored tiles.
- **Spikes:** Pointy obstacles that will cause the player to die on collision regardless of remaining health value (instant death). Can be standard or colored tiles.

2.4. Enemy Types

The player will encounter enemies during the game. There will be several types of enemy which have different features in terms of their abilities. All enemy types may carry weapons and shoot the player.

The enemy types are:

- ***Walker w/o weapon:*** This type of enemy will move without stopping changing direction when it encounters a wall/obstacle.
- ***Patroller w/o weapon:*** This type of enemy checks whether it reaches the ledge, and change its direction accordingly.
- ***Jumper w/o weapon:*** This type of enemy checks whether it is on the floor and jumps.
- ***Floater w/o weapon:*** This type of enemy floats above the floor following a predetermined path.

2.5. Power-ups

In some cases the power-ups in this game will allow the player to pass the level with more ease while in others it will be required to pass the level. Power-ups will appear in predetermined locations when the level is initialized. These power-ups will augment the players own capabilities and attributes and will not effect enemies directly. The player will be able to activate these powerups by moving through (colliding with) them. The power-ups and their icons that will be used in this game are the following.

- **Increase Rate of Fire:** Increases the frequency in which the weapon can shoot bullets.
- **Full Spectrum Mode:** Allows bullets to damage enemies regardless of color.
- **Increased Move Speed:** Increases the players horizontal move speed.
- **Temporary Invulnerabilty:** Protects the player from hazards. Instant death hazards not included.
- **Fill Health:** Refills the players current health bar.

3. Requirement Specificaiton

3.1. Functional Requirements

3.1.1. Play Game

Color Shooter: The Spectrum Adventurer (CSSA) is a Mario-like 2D platform game which is enriched by power-ups and various arcade features to enhance playability. The goal of the game is to complete all levels successfully. In the beginning of the game, the Player starts with full health. Player has to reach the goal at the end of the level without spending all of his/her health to pass through to the next level. The game will be over, if the player falls into any spiked pit. Player has to jump successfully over all pits to continue the game. Depending on difficulty choice, jump mechanism changes as manual or automatic. In addition, the player can take damage from enemies. If player gets damaged, his health will be reduced and the game will end when he spends all of his health. Player can also harm enemies, but he must

use bullets of the same color as the enemy's color. Some power ups will appear at points determined by the system during the game.

3.1.2. Change Settings

CSSA allows the player to change the default settings defined by the system. This section contains two scrollbars. One of them is to adjust the volume of background music and the other sets the volume of the sound effects(SFXs). Player can change the settings during gameplay by pausing the game or before starting game from ‘Change Settings’ on the Main Menu.

3.1.3. Pause Game

The game can be paused by the player during the game. Player can continue the game from where he paused. He can also make changes in Pause mode, when the game stops. These changes are restricted with changing volume of the background music and the sound effect. From this mode, player can return to the Main Menu as well.

3.1.4. View Help

Player can get all instructions about the game. Instructions are listed below:

- Instructions to control Player
- Color Mechanism
- Jump Mechanism
- Effects of Power-ups

This section aims to maximize entertainment by giving information about the gameplay.

3.1.5. View Credits

Player can receive contact information of the game developers. Also, player can access the information of the game developers' GitHub page so that he can browse their other projects.

3.2. Non-Functional Requirements

3.2.1. Game Performance

It is a priority that the features we will add will not negatively affect playability. That's why we will try to make CSSA work with highest performance. Moreover, we will set the system requirements of the game at the lowest level. Thus, even an under average computer will be able to run it.

3.2.2. User-Friendly Interface

CSSA will be a game that is easy to understand. To do this, we plan to give priority to how fluid the UX(User Experience) of CSSA is. We will design the game with smooth graphics and avoid the images that cause to eye strain in order to build a user-friendly interface.

3.2.3. Portability

Supportability is an expected element in every game today. Since CSSA will be developed using Java infrastructure, it will work on all major platforms by taking advantage of Java's portability.

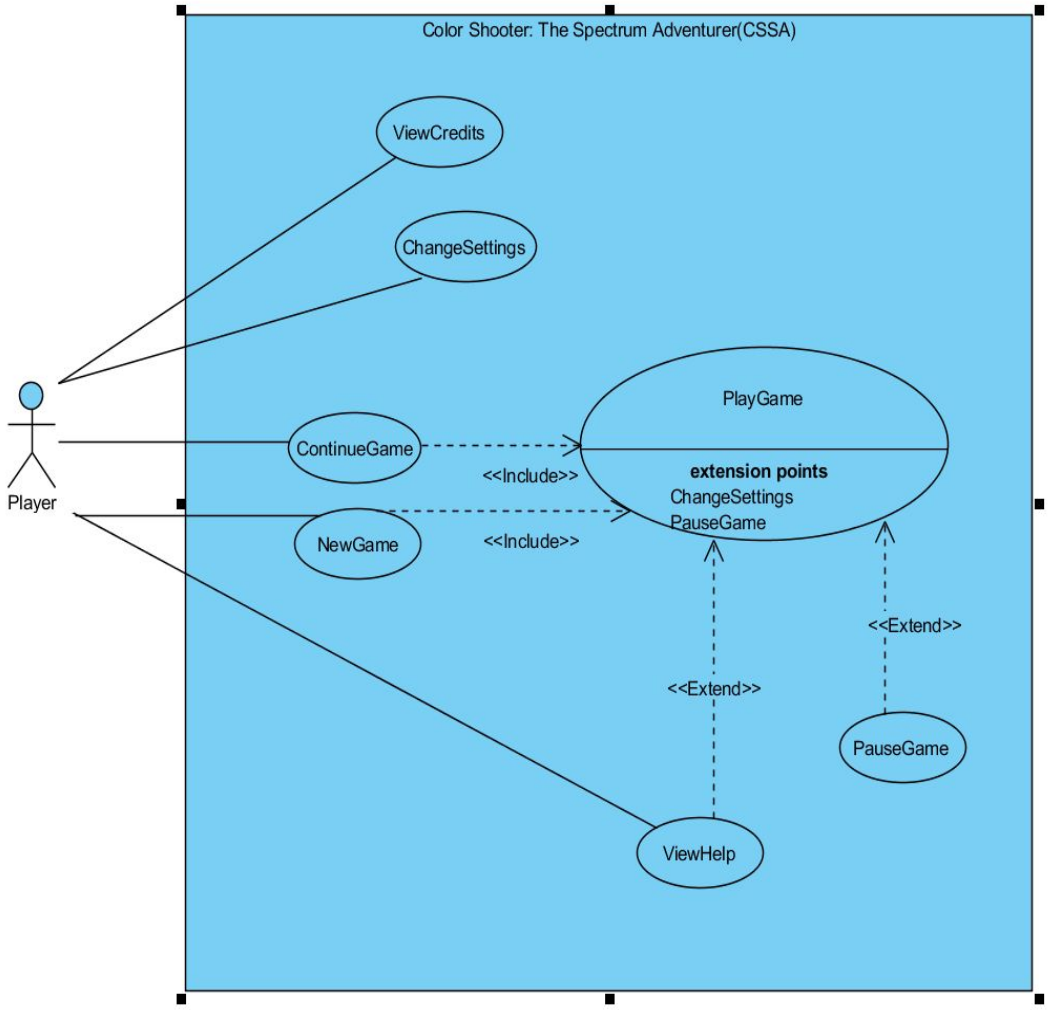
3.2.4. Extendibility

Reusability and extendibility are two important concepts in software engineering. We will make CSSA following this principle so that it will be suitable to be extended and reused in future projects. Later on, the project can have a login system and a score system, and they can be stored in database. The high score table can be displayed in the main menu as well, however these features will be left as a future work. Hence in order to extend our project in the future, we will design the Color Shooter: The Spectrum Adventurer in a way that the programmer can easily add new concepts to the game.

4. System Model

4.1. Use Case Model

The figure 4.1 illustrates the Use Case diagram of the Spectrum Adventurer. Textual description of each use case will be explained as well.



4.1.1. ViewCredits

Use case name: ViewCredits

Participating Actor: Player

Pre Condition: Player should open Main Menu.

Post Condition: -

Entry Condition: Player should select the button named as“View Credits” on the Main Menu.

Exit Condition: Player should select the “Return Main Menu” button on the “View Credits” screen in order to return Main Menu.

Flow of Events:

1. Player selects the “ViewCredits” button on the Main Menu.
2. CSSA displays the view, which includes information about the developers.
3. Player selects the Back button on the screen to return Main Menu.
4. CSSA displays the Main Menu.

4.1.2. Play Game

Use Case Name: Play Game

Primary Actor: Player

Stakeholders and Interests:

-Player aims to complete all levels and with the least amount of deaths.

-System keeps track of how many times the player has died.

Pre-condition: For first running, game settings are set as default. If Player changes game settings, adjusted settings (s) will be saved and used by the Game Manager.

Post-condition: The game will tell the player how many times he has died at the end of the game.

Entry Condition: Player selects “Play Game” button from Main Menu.
Exit Condition: Player selects “Return to Main Menu” from Pause Menu.

Success Scenario Event Flow:

1.Game is started by Game Manager.
2.Player starts playing from first level.
3.Player plays the level until he reaches the level’s goal.
4.System grants access to next level.
5.Player starts playing next level.
Player repeats the steps 3 – 5 until all levels are completed.
6.Game Manager records the Player’s total number of deaths and gives him a rank based on what interval his number of deaths are in.
7.Game Manager returns to Main Menu.

Player repeats the steps 1 – 7 if he wants to play the game again and attempt tog et a better rank.

Alternative Flows:

3A. Player dies by falling on spikes:

3A.1. Player starts playing the level.
3A.2. Player jumps or falls.
3A.3. Player falls on spikes.
3A.4. Player dies, death count is incremented.
3A.5. Game Manager reinitiliazes the current level.
3A.6. Player reattempts to pass level.

- *Player follows the anyone of the flows for Play Game again*

3B. Player collects the power ups during game:

3B.1. .Player collides with a power-up.

3B.2. Game Manager removes the insatnce of the collided power-up.

3B.3. The Game Manager will apply necessary changes to game objects based on the type of the power-up collected.

3B.4. Player tries collects power up(s) by using his paddle.

- *Whenever a power up is collected during the game, steps 3B1-3B4 are applied.*

A. If player requests to pause the game at any time during the game:

A.1. Player presses proper key from keyboard to pause the game.

A.2. The Game Manager pauses the game.

A.3. The Game Manager displays the pause menu.

A.3.1. If the Player selects resume game from pause menu, Game Manager resumes the game.

A.3.2. If the Player should select “Return to Main Menu” the Game Manager will save the progress up to that point and then returns to the Main Menu.

A.3.3.If the Player uses the scrollbar for music and SFX volume the Game Manager will follow steps 2-3 of the change settings use case.

A.3.4. If the Player selects “View Help”, View Help use case is applied.

4.1.3. Change Settings

Use case Name: ChangeSettings

Participating Actors: Player

Pre-condition: CSSA sets music and sound settings to max as default.

Post-condition: Sound and music settings will be changed according to the preferance of the Player.

Entry Condition: Player clicks to the "Change Settings" button in the Main Menu.
Exit Condition: Player clicks to "Return Main Menu" button.

Flow of Events:

1. Player clicks to "Change Settings" on the Main Menu.
2. Player can change Sound and Music Settings of the game in this menu.
3. CSSA changes the settings according to Players choice.

4.1.4. View Help

Use case name: ViewHelp
Participating Actor: Player

Pre Condition: Player should be on the Main Menu or Pause Menu.
Post Condition: -

Entry Condition: Player should select 'View Help' button on the screen in the Main Menu or while on the Pause Menu.
Exit Condition: Player should click the "Back" button to return to Main Menu or the Pause Menu.

Flow of Events:

1. Player clicks on 'View Help'button on the Pause Menu.
2. CSSA opens the View Help screen.

Alternative Flow of Events:

1. Player clicks to 'View Help'button on the Main Menu.
2. CSSA displays a screen which displays specific informations about the game, how to play the game, explains some special features of the game.
3. Player returns to the Main Menu by clicking 'Return to Main Menu' button.

4.1.5. Pause Game

Use case name: PauseGame

Participating Actor: Player

Pre Condition: Player should be playing the game.

Post Condition: Game should continue from exactly same place when player pauses the game.

Entry Condition: Player should select 'Pause' button on the screen or press 'P' on the keyboard.

Exit Condition: Player should select 'Resume' button to return the game.

Flow of Events:

1. Player clicks to 'Pause' button on the screen or presses 'P' on the keyboard.
2. CSSA stops the game and displays the pause screen which has options to
   - Resume the game
   - Quit to Main menu
   - View Help
   - Change music volume
   - Change sound effects (SFX) volume

4.1.6. New Game

Use case name: NewGame

Participating Actor: Player

Pre Condition: -

Post Condition: Game should start from the first level.

Entry Condition: Player should select New Game button on the Main Menu.

Flow of Events:

1. Player clicks to 'New Game' button on the Main Menu.

2. Player selects difficulty.
3. Game manager reinitializes save file.
4. Game Manager starts level 1.

4.1.7. Continue Game

Use case name: ContinueGame

Participating Actor: Player

Pre Condition: Player shoould be on the Main Menu screen.

Post Condition: Game should continue from the same level where the Player last left off.

Entry Condition: Player should click 'Continue Game'button on Main Menu.

Flow of Events:

1. Player clicks to 'Continue Game' button on Main Menu.
2. Game Manager retrieves the last unlocked level from the save file (1 if the game wasn't played yet)
3. Game Manager opens the level retrieved from the save file.

4.1.8. Exit Game

Use case name: ExitGame

Participating Actor: Player

Pre Condition: Player should be on the Main Menu in order to exit the game.

Post Condition: Game will be closed.

Entry Condition: Player should click 'Exit Game' button on the Main Menu.

Flow of Events:

1. Player clicks to 'Exit Game' button on the Main Menu.

2. CSSA stops the gameplay and closes the game immediately.

Alternative Flow:

1.While playing the game, if Player wants to exit, he/she can pause the game.

2. On the Pause Menu, by clicking on the 'Return Main Menu' button, Player can return to Main Menu and exit from the game.

4.2. Dynamic Models

This section provides detailed information about the Crazy Ball game by illustrating some crucial scenarios of the game in sequence diagrams. Since the ball and the paddle objects are two crucial objects of the game, states of these particular objects are also described in state diagrams. Besides, activities of the system during the gameplay are also stated in activity diagram in this section.

4.2.1. Sequence Diagrams

4.2.1.1. Start Game

Following sequence Diagram illustrates the scenario explained below:

Scenario 1: New Game

Player Mehmet requests to start the game by clicking on the "New Game" button from the main menu. After that the Game Manager resets the save file and initializes the first level. Then according to the corresponding file the Game Manager gets the position of the all the elements (player, tiles, enemies, powerups, goal) and places them on the game screen. Finally, the Game Manager enters the game loop, continuously updating the game.

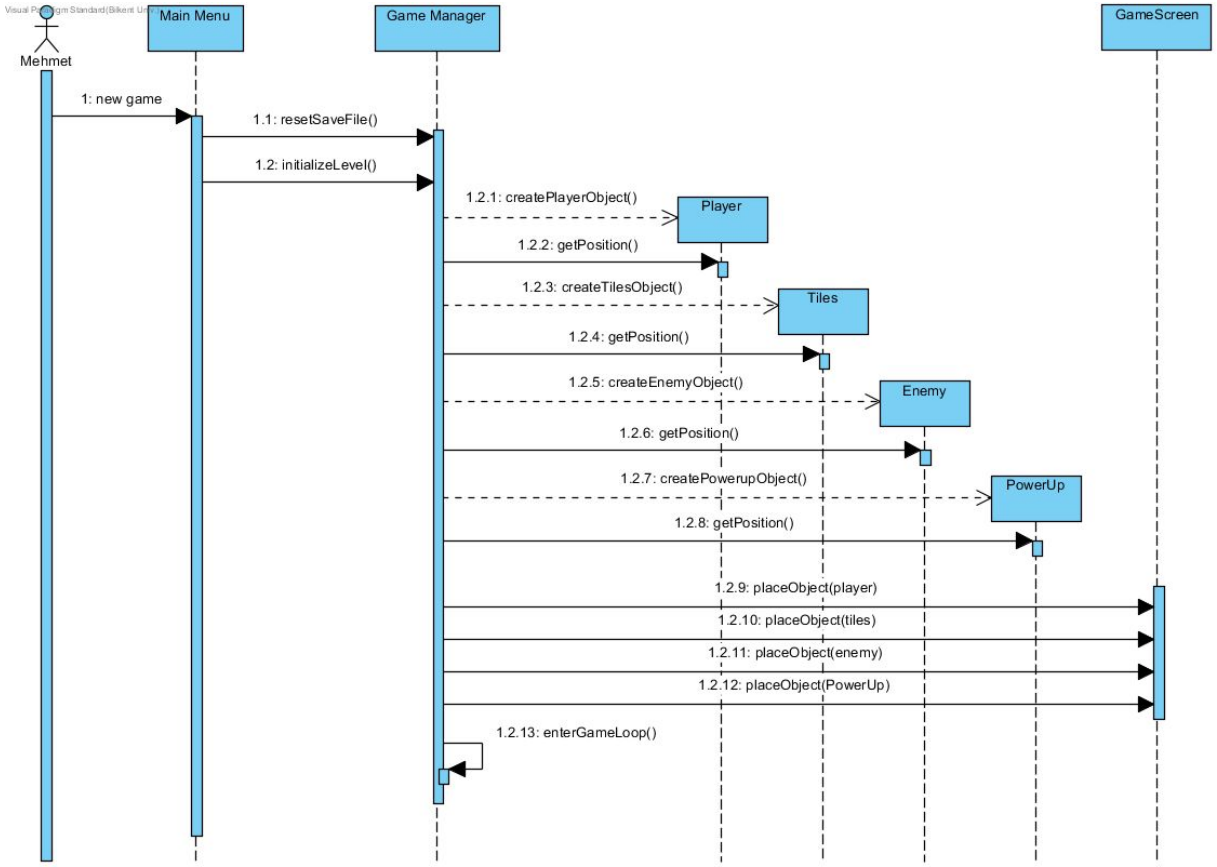


Figure 4.2.1.1.a sequence diagram which explains new game scenario

Scenario 2: Continue Game

Player Mehmet requests to start the game by clicking on the “Continue” button from the Main Menu. After that the Game Manager loads the save file and initializes the last level that the player had unlocked. Then according to the corresponding file the Game Manager gets the position of the all the elements (player, tiles, enemies, powerups, goal) and places them on the game screen. Finally, the Game Manager enters the game loop, continuously updating the game.

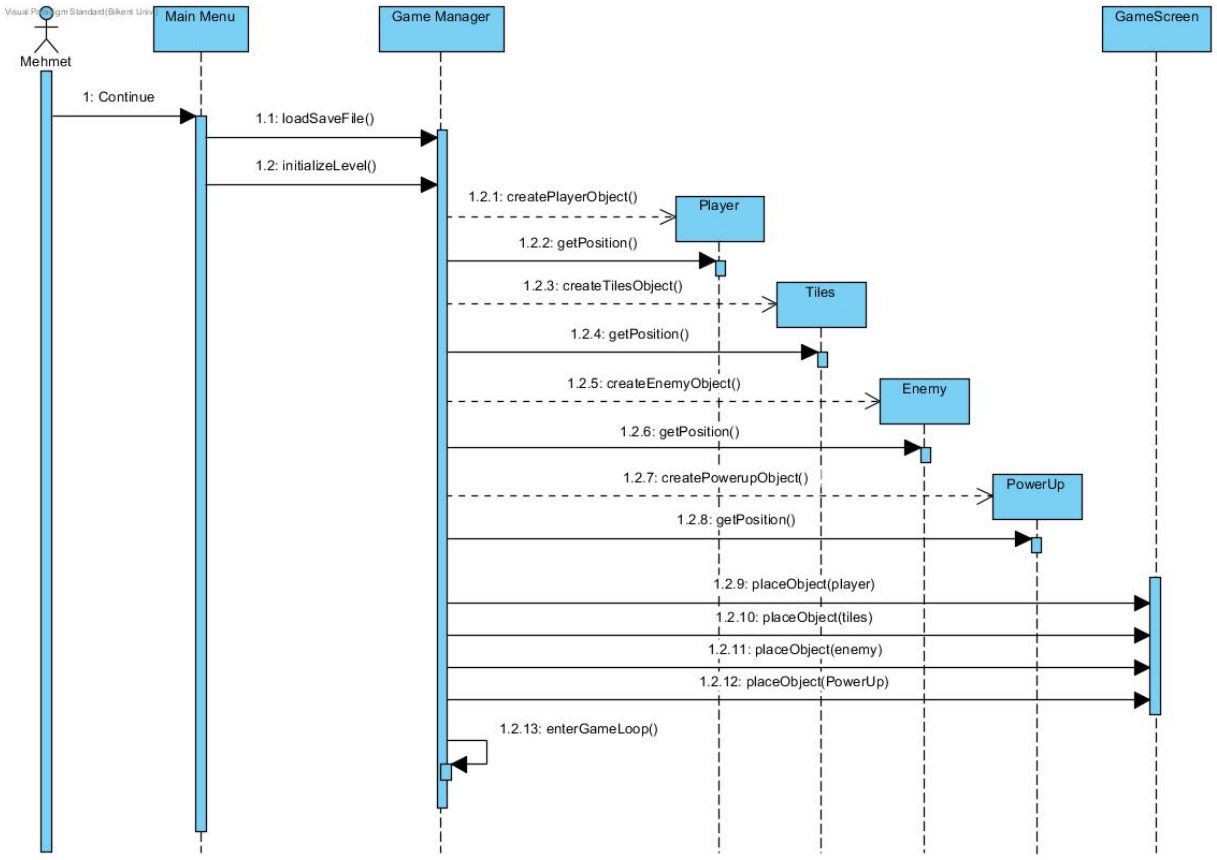


Figure 4.2.1.1.b sequence diagram that explains continue game scenario

4.2.1.2. Power up Management

Following sequence Diagram illustrates the scenario explained below:

Scenario: Player İrem requests to start game by pressing "New Game" or "Continue" from the Main Menu. Assuming that the steps from the previous sequence diagram are done, the Game Manager enters a game loop, managing all the game's dynamics. When the player object collides with a power-up, the collision handler will destroy that particular power-up instance. It will then perform the effects of the power-up on the player. In this instance we will assume it is the health up power-up. In this case the Collision Handler will assign the player's health attribute to its maximum value.

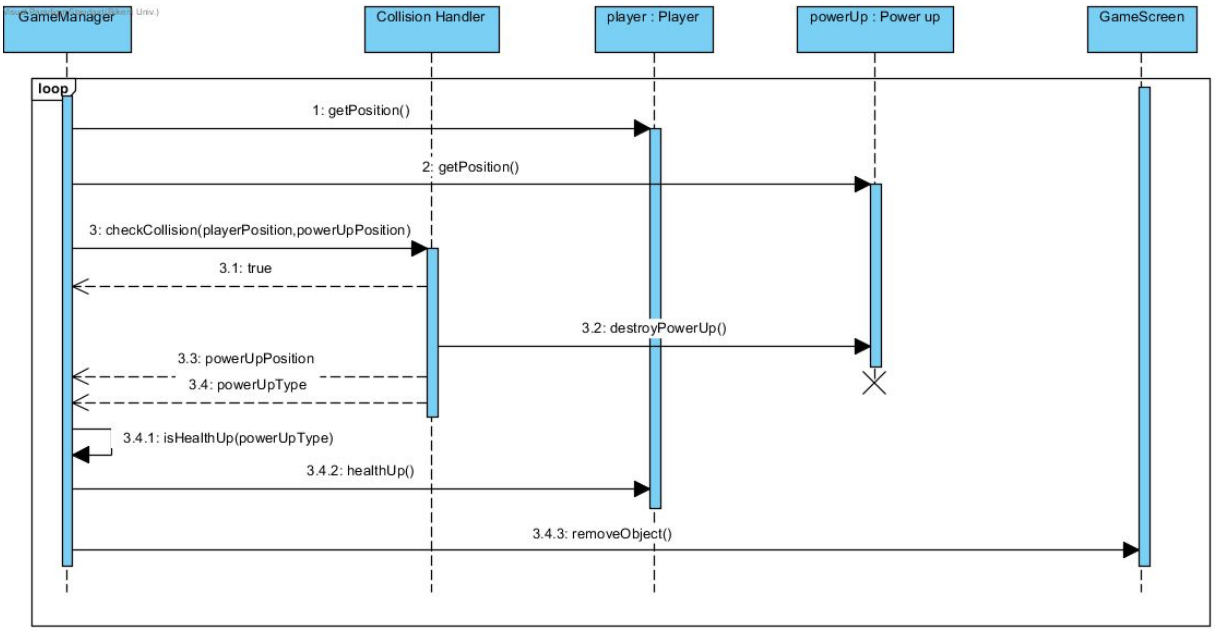


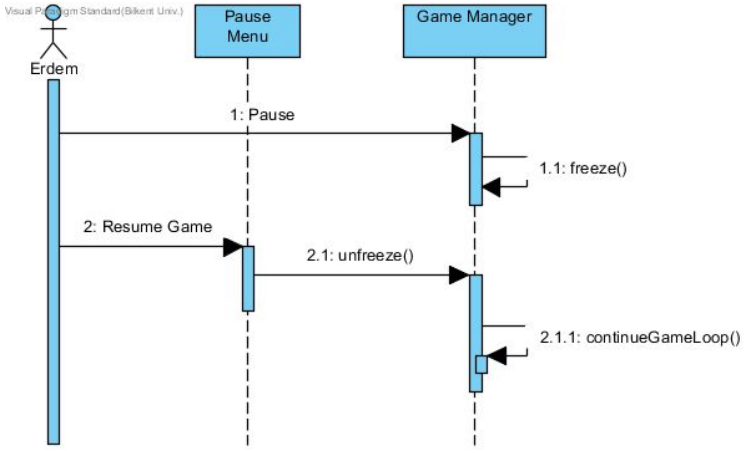
Figure-4.2.1.2 sequence diagram of power up management

4.2.1.3. Pause Game

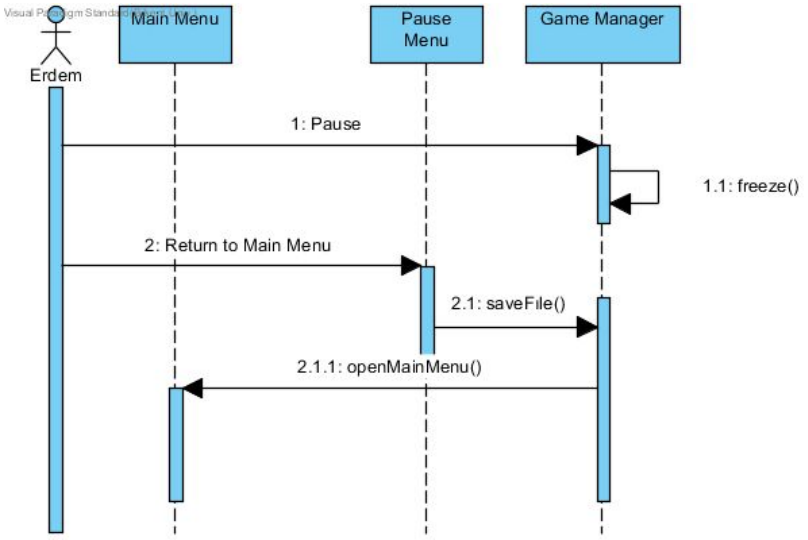
Following sequence diagram illustrates the scenario explained below:

Scenario 1: Erdem wants to pause the game. He presses the "P" key on the keyboard while playing a level. The Game Manager freezes the game.

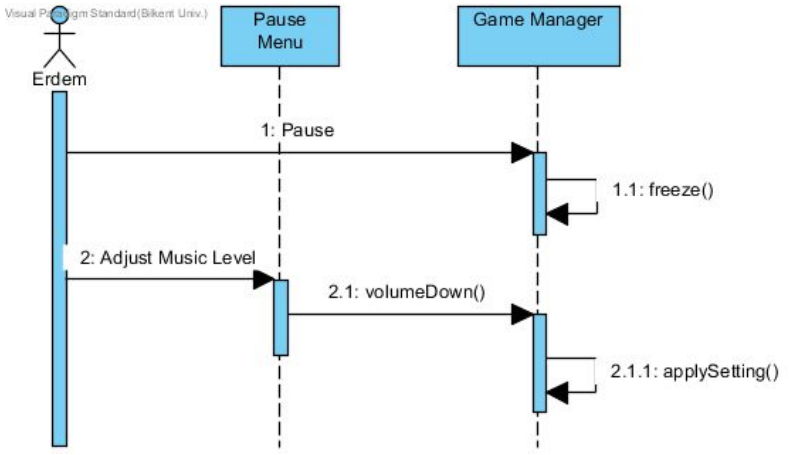
Scenario 1.1: Erdem wants to continue the game. He clicks on the "Resume Game" button on the pause menu with the left mouse button. The Game Manager unfreezes the game. Erdem continues to play the game.



Scenario 1.2: Erdem wants to return to the "Main Menu" screen. He clicks on the "Return To Main Menu" button on the pause menu with the left mouse button. The Game Manager saves the player's progress on the corresponding save file. Then the Game Manager opens the "Main Menu Screen".



Scenario 1.3: Erdem wants to change the music level of the game. He presses down the left mouse button on a scroll bar under "Music"(informative text) and drags the controlling component of the scroll bar (left (%0) to right (%100)) to adjust the music level to a percentage he desires.



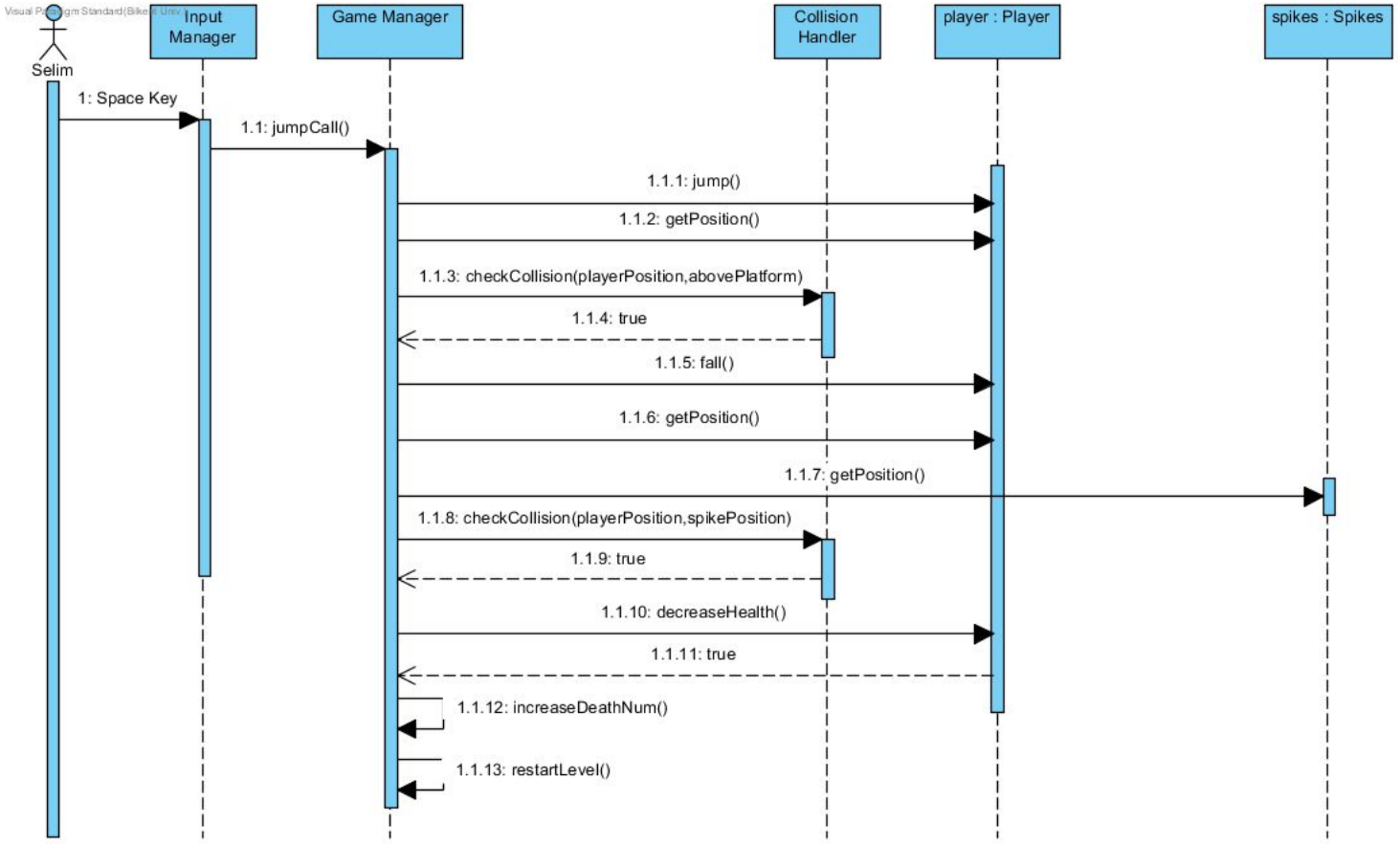
Scenario 1.4: Erdem wants to change the sound effects level of the game. He presses down the left mouse button on a scroll bar under "SFX" (informative text) and drags the controlling component of the scroll bar (left (%0) to right (%100)) to adjust the sound effects level to a percentage he desires.

4.2.1.4. Death of Player

Basic scenarios for the death of player is considered in this part and the sequence diagrams of these scenarios are given. More complex ones, which player encounters a power up, or came across with different enemies will be left as to do for next iteration.

Scenario: Player Selim already starts playing the game in easy mode . He uses keyboard commands while playing the game.

Scenario 1.1: He came across with a ledge and he presses space to jump through the ledge. However he collides with the above platform and he falls into the pit with thorns and this causes an immediate death. Number of deaths on the game screen is increased by one.

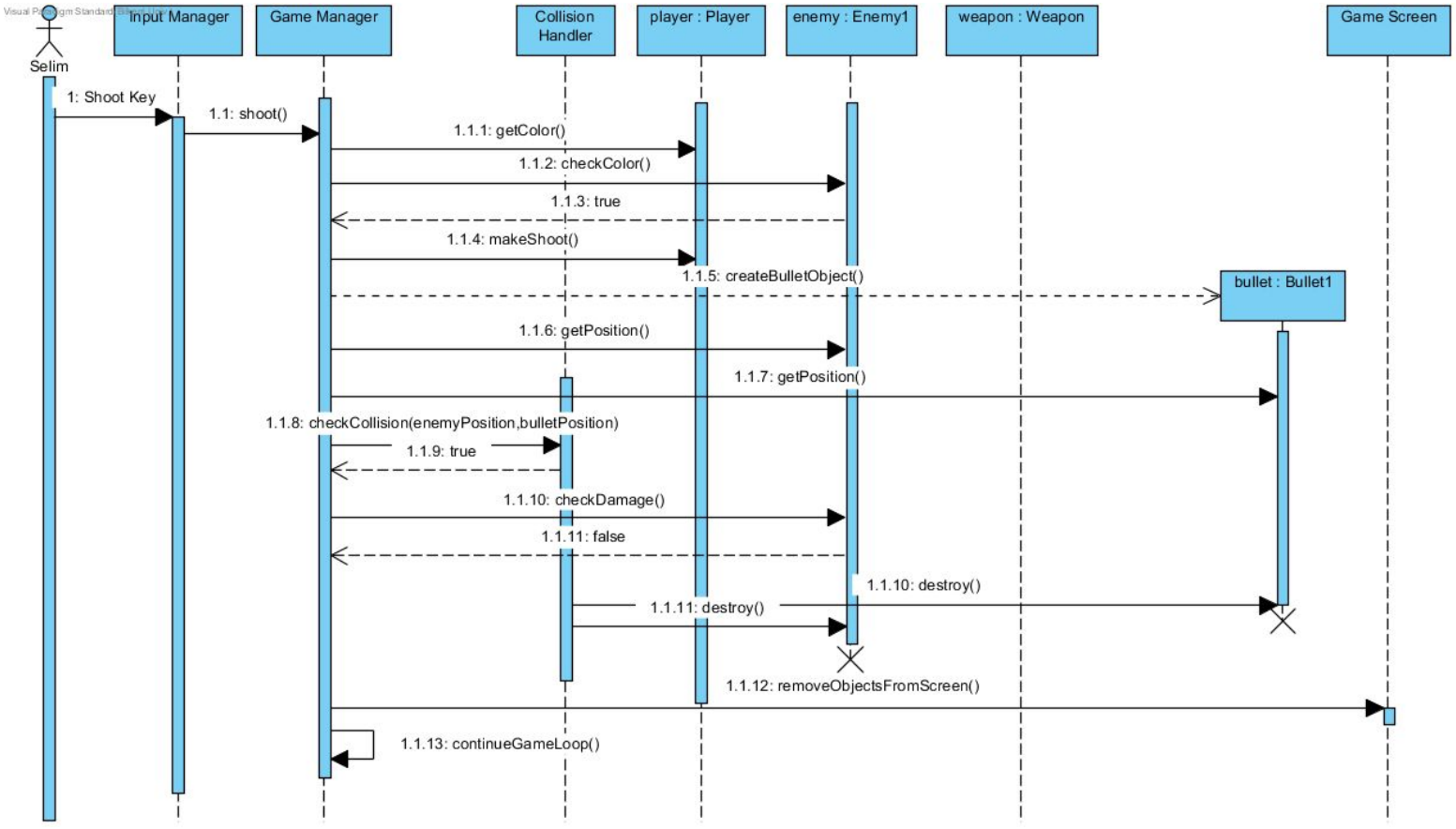


Scenario 1.2: Selim encounters an enemy with red color on the above platform, and he must go through that platform, which is in black color, to finish the level. He changes his color from blue to red to kill the enemy. In order to kill the enemy Selim jumps to the above platform. However when he jumps, he touches(collides) with the enemy and it caused an instant death of the player. Number of deaths on the game screen is increased by one.

Scenario 1.3: Selim plays a level with several enemies with weapons. While defeating the enemies, he changes his color to shoot enemies. Unfortunately he gets several damages from bullets while changing his color to shoot. When these damages cluster, health bar on the screen becomes empty and the player dies. Number of deaths on the game screen is increased by one again.

4.2.1.5. Kill an Enemy

Scenario: Player Mehmet has been playing the playing the game. When he encounters an enemy, he changes his color according to the color of the enemy and he starts shooting to the enemy. If the bullets send by Mehmet, reaches to the enemy(collides with the enemy), the health bar of the enemy will decrease. Finally with enough damage, by shooting the enemy for some time Mehmet can kill the enemy.



4.2.2. Activity Diagram

4.2.2.1. GamePlay Activity Diagram

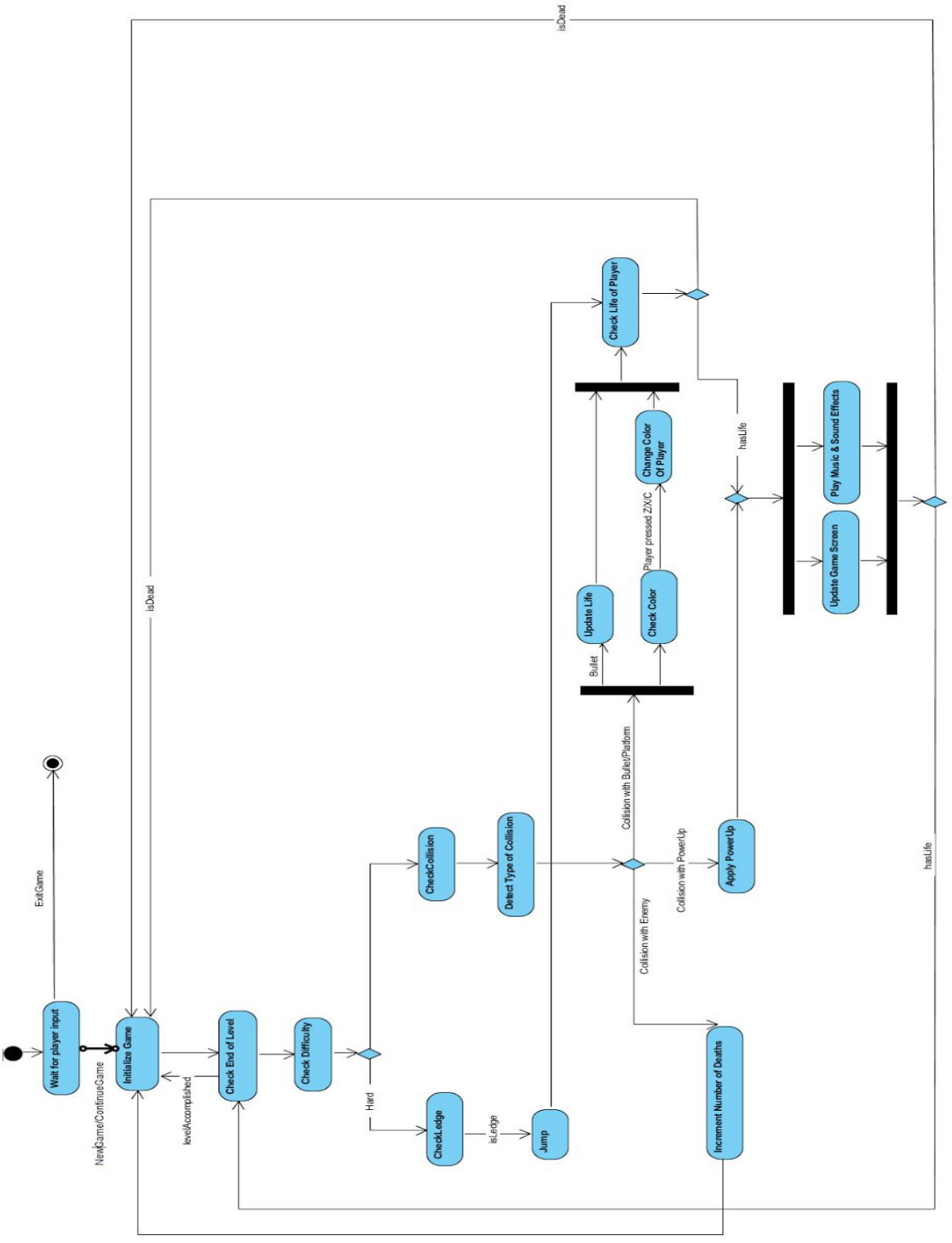
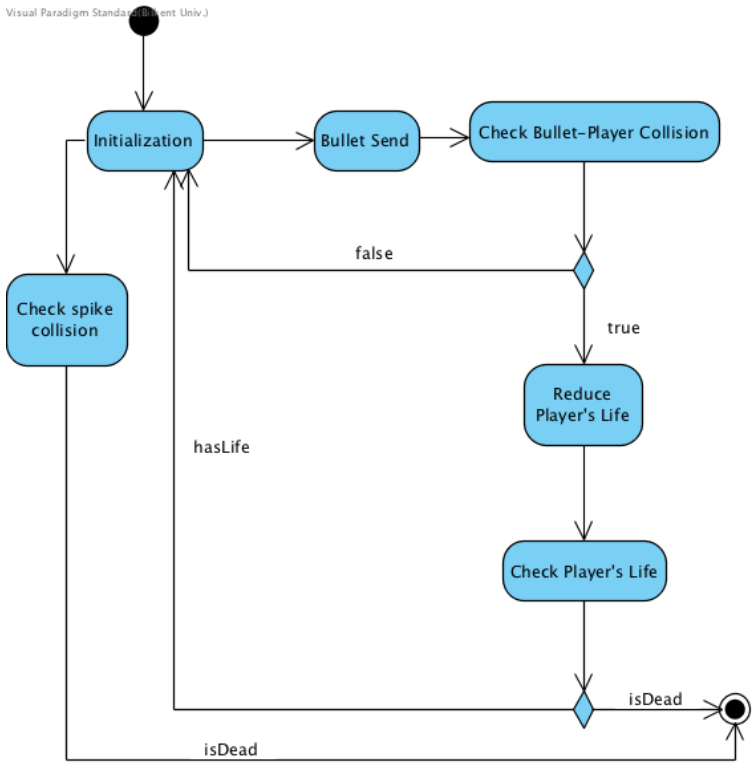


Figure 4.2.2 GamePlay Activity Diagram

In Figure 4.2.2 Activity diagram describes the behaviour of the systems during the gameplay. Main activities are included in the diagram. Firstly, the game starts if player clicks to NewGame or ContinueGame. The level finishes if player overcomes the obstacles and reach to the door in the game view. Hence there is an activity which checks if player reaches to the goal or not. There are two difficulty levels in the game, easy and hard. If player selects the hard mode, the system will jump automatically if walks over ledges. After jump activity, there is a check for life of player as player can die if he/she could not control his movements(direction of the jump left/right) during the jump. Furthermore, the collision handler will be used continuously in the game. It will detect a collision between the player and the game objects such as the platform, powerups, enemies and the bullets released from some type of enemies. If there is a power up, then it will apply the power up to the Player and then update the UI accordingly. However if Player hits an enemy, he/she will die immediately and the number of deaths will be increased. Lastly, if Player touches a bullet, life of the Player is decremented, and if Player hits a platform, color of the player and the platform will be compared. If Player hits a platform, Players keyboard input should be taken as well, or the system should check if there is an input. The reason behind this activity is that, in order to pass the platform, Player should arrange his/her colour. The last two possibilities should be synchronized as there is a user input, there might be certain time delay. After that, there is a life check. If Player dies, game starts at the beginning of the level. On the other hand, at the end, UI should be updated according to the changes and there should be music and sound effects at the background, these two activities should be synchronized. Then it returns to the first activity, it checks whether Player finished the level or not.

4.2.2.2. The Damage Taken By Player Activity Diagram



4.3. Game Class Diagram

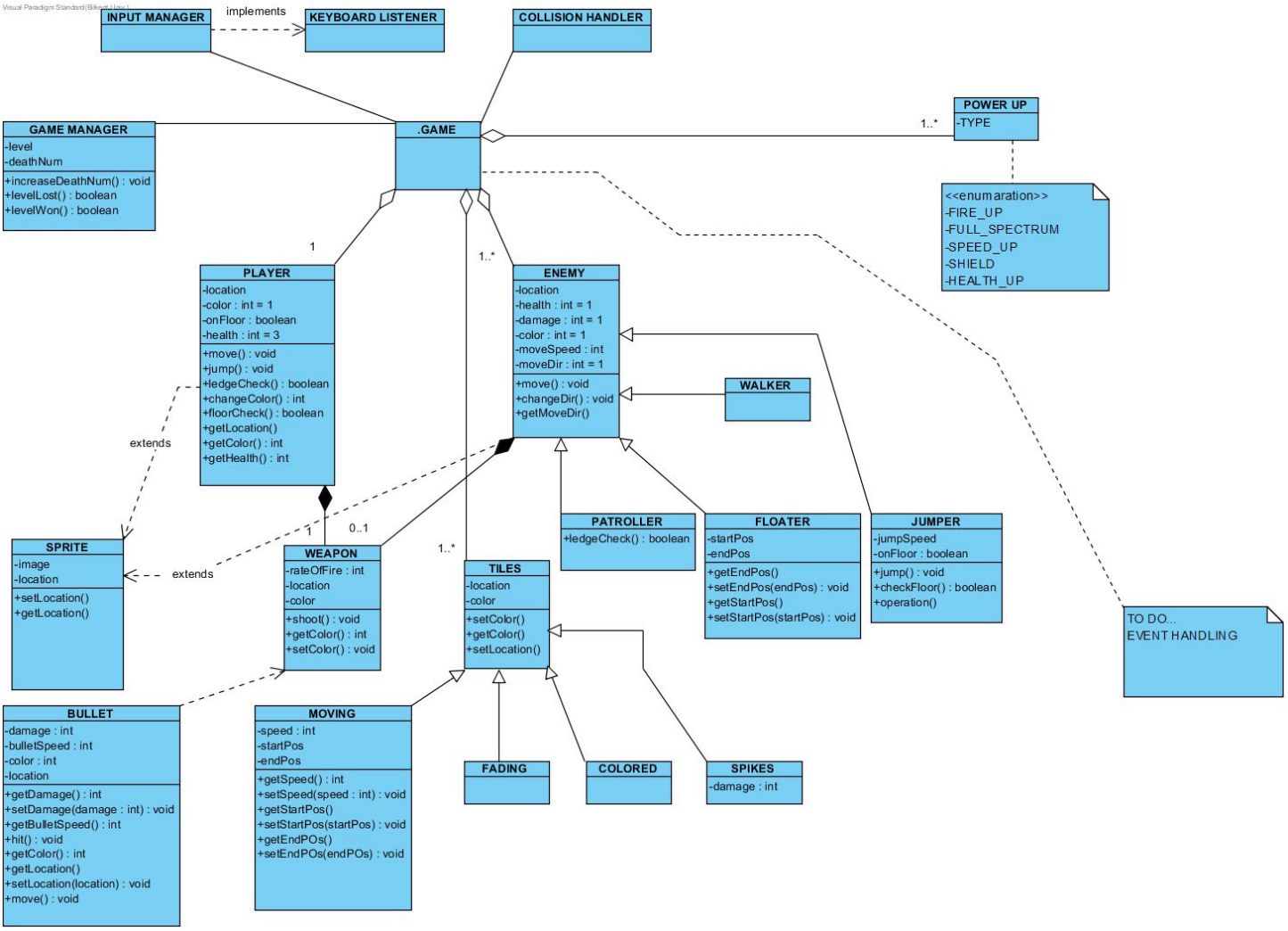


Figure-4.3 Class diagram

5. User Interface

5.1. Navigational Path

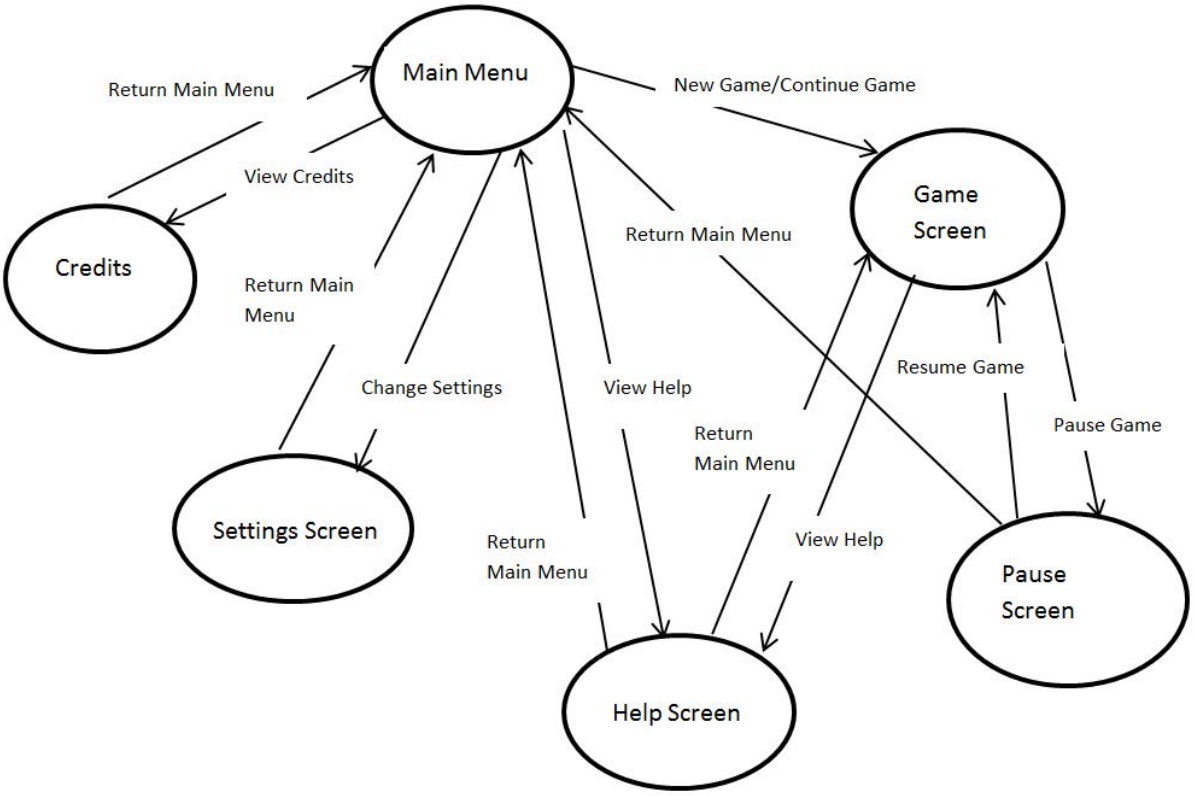


Figure 5.1 Navigational Path of Color Shooter: The Spectrum Adventurer

5.2. Screen Mock-ups

5.2.1. Main Menu

When the Player runs the executable the Main Menu screen (Shown in figure 5.2.1.1) will be the first screen the Player will see. The Main Menu screen will have 5 options which are, new game, continue game, change settings, view help and credits.

Figure 5.2.1 Screen shot of Main Menu

-New Game: If the Player selects New Game, the game will reset the save file and start the game on level 1.

Figure 5.2.1.1 Screen shot of Level 1 (current design)

-Continue Game: If the Player selects New Game, the game will load the save file and start the game the level that player last unlocked.

-Change Settings: When the change settings button is clicked a settings menu with two scroll bars will show. These two scroll bars will manage the games music and SFX levels and can be changed by dragging it to the left to decrease volume and by dragging to the right to increase volume.(Figure 5.2.1.2)

Figure 5.2.1.2 Screen shot of Change Settings

-View Help: If the player clicks on the view help button , a new screen appears explaining the jumping mechanism, color mechanisms and powerups. Player can return to previous menu by clicking the back button (Figure 5.2.1.3)

Figure 5.2.1.3 Screen shot of View Help

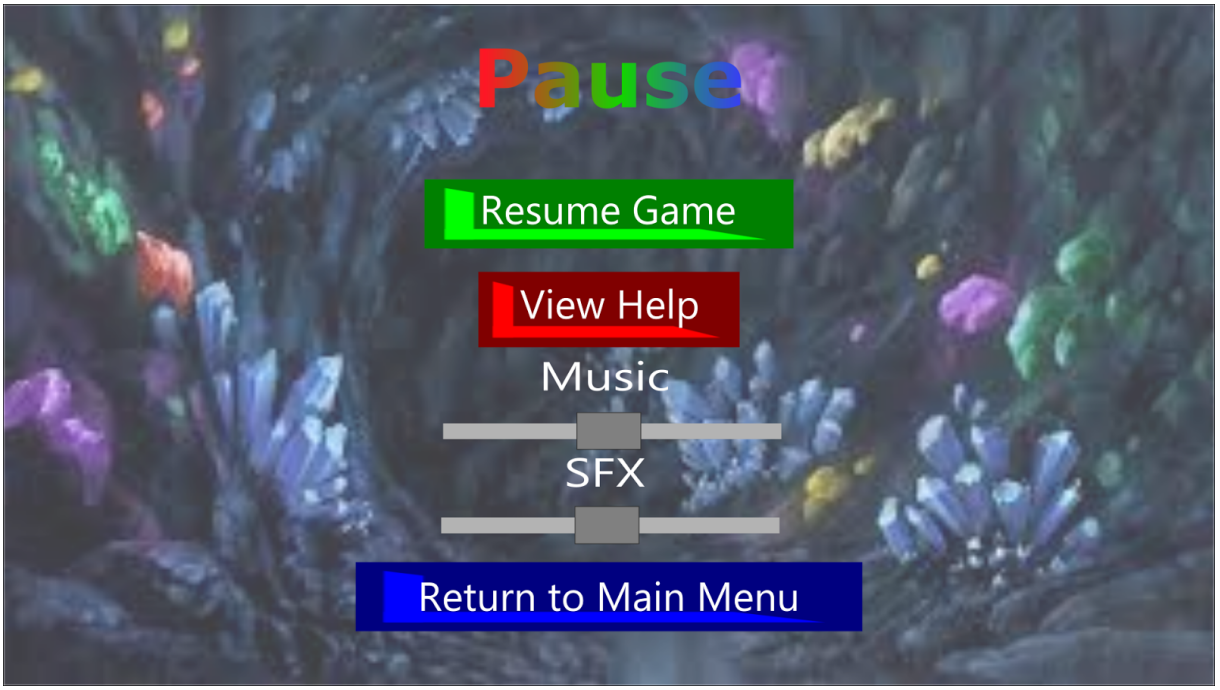
-Credits: When Player selects Credits, contact information and names of game developers are shown on screen. Player can return to previous menu by clicking “Back”. (Figure5.2.1.4)

Figure 5.2.1.4 Screen shot of Credits

-Exit Game: When player selects exit game, application stops running and game window is closed.

5.2.2. Pause Menu

When Player pauses game by pressing “P” on keyboard, pause menu will be shown on screen. Content of Pause menu is similar with Main Menu. However, exit game option is not in pause menu. Therefore; if player wants to exit game, he should select “Return to Main Menu” then he can exit game. (Figure 5.2.2)



6. References

- Call Me Young Kaleido (http://armorgames.com/call-me-young-kaleido-game/18162?tag-referral=platform)
- Jumphobia (http://armorgames.com/jumphobia-game/18009?via-search=1)