CS 319 - Object-Oriented Software Engineering

Analysis Report

Jungle Pursuit

Fatma Begüm İlhan

Syed Sarjeel Yusuf

Sinan Öndül

Table of Contents

Table of Figures

1. Introduction

Jungle Pursuit is a replica of the board game Snakes and Ladders. We are planning to develop Jungle Pursuit with different features, images and with a slightly different gameplay that we remember from Snakes and Ladders. The game is meant to revolve around a jungle-like theme and hence the images that we choose will be appropriate for such a theme.

Our game is played by 1 or 2 players, rolling a dice in each turn to move their character. The path to victory consists of 100 squares and the goal is to reach 100th square. There will be similarities with Snakes and Ladders like Vine Branches as Ladders to move up, Snakes that eat the character and move the character backwards, some bonuses that move the character onwards or backwards randomly and shields that protect the character getting bit by a snake.

The game will be a mobile application based on Android Development. That is controlled by touchscreen commands to roll the dice. The player won't be interacting with the game other than touching the dice button to roll. Also, the player can shake his/her Android device to roll. The character will pursue its path according to the dice.

2. Requirement Analysis

2.1 Overview

The game involves players rolling a dice which determines the number of blocks they move. The players start on block 0 and their goal is to move up the board to block 100. The blocks are ordered in rows of 10. Moreover, the board is littered with snakes and ladders. Snakes are objects where if the user lands on the head of a snake on the board, he is transported down a certain number of block rows. Similarly if the user lands on the start of a ladder, he is transported up a certain number of block rows.

Our game is intended to be different from the traditional game and themes around a jungle environment. Hence the game will includes snakes and vines instead of snakes and ladder animations. Also it is going to include lucky bananas on the board and items. If the user lands on the lucky banana he will get a random number between -6 to 6 which will decide whether he slides forward or backwards. The items are meant to add more substance to the game. There will be two kinds of items, antivenom, and oil. The antivenom is meant to prevent you from sliding down if you are bit by a snake. The oil prevents you from climbing the vine as your hands are now oily. Furthermore, the game, unlike the traditional game, will allow players to 'beat' their opponent back three blocks if they land on the same block as their opponent. This is meant to make the game more competitive and hence more fun.

Other features of the game will include name panels, which will also display if the player has an item associated with him. The dice can be rolled by shaking the phone or clicking on a roll button. Moreover, our game allows a maximum of 2 players playing together, where the 2 players can be a human vs human or human vs the computer. Thus you can play against your friend or you can play with the computer.

We have to keep in mind that the game is turn based, where each player has his turn to roll the dice and move his character. Hence we must ensure that the game is implemented such where a player is not allowed to roll twice and if the phone is being passed from one user's hand to the other, it should not be detected as a shake to roll dice.

2.1.1 List of Dices

Figure 1: Dice 1

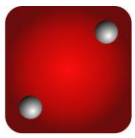

Figure 2: Dice 2

Figure 3: Dice 3

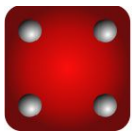

Figure 4: Dice 4

Figure 5: Dice 5

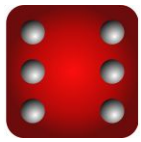

Figure 6: Dice 6

2.1.2 List of Collectables

Figure 7: Banana Skin

Figure 8: Oil

2.1.4 List of Snakes

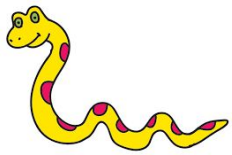

Figure 9: Snake 1

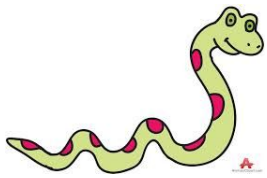

Figure 10: Snake 2

2.1.3 List of Characters

Figure 11: Character 1

Figure 12 : Character 2

2.1.5 List of Other Objects

Figure 13: Anti-Venom

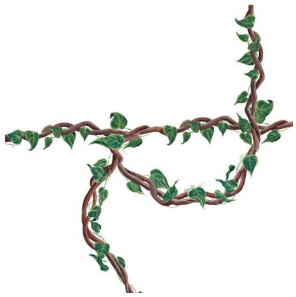

Figure 14: Vine

2.2 Functional Requirements

- Players will be moved automatically according to dice role and where they land and what item or object they land on.
- The player or players should be able to enter their name that will be displayed in a game panel
- The player should be able to collect items causing them to despawn on the board.
- User should be able to change character and choose:
    - Male Character
    - Female Character
- User will be able to access the help menu that consists of information about how to play the game and items.
- User can toggle music on and off.
- The user should be able to roll dice by shaking phone or clicking on role button.
- The users should play turn based.
- User should be allowed to play multiplayer or single player
- Players should be able to beat opponents back 3 squares

2.3 Non-functional Requirements

- Control mechanism of the game shall have short response time that allows the player to play with minimal delay.
- The snakes and vines should be placed appropriately on the board to ensure clear understanding of where the snake or vine starts and ends.

- The placement of items and snakes and vines should be static for each time the user starts a new game.
- The dice shall be a random number generated from 1 to 6.
- The lucky banana shall be a random number generated from -6 to 6, including 0.
- Dice roll time and animation should be appropriate for game aesthetics and response

2.4 Constraints

- The game will be implemented in native Android, Java programming.
- The game will be played in 30 FPS
- Drawings in the game will be smooth.
- The game will run on all devices that support Android OS, irrespective of other constraints that determine phone performance.

2.5 Scenarios

2.5.1. View Help

Use Case Name: Help

Primary Actor: Player

Entry Condition: Player selects “Help” option from Main Menu.

Exit Condition: Player selects “Back” option to return to Main Menu.

Event Flow:

-Player wants to know how to play the game.

-Bulleted list of instructions to play the game and list of items is displayed

Pre-conditions: Strings in Android Resources file contains the information to display.

Post-condition: -

Success Scenario Event Flow:

1. System displays instructions on how to play the game.

Alternative Flows:

-If player desires to return to Main Menu at any time:

1) User selects “Back” in Action Bar of android to return to Main Menu.

2) System displays Main Menu.

2.5.2. View Credits

Use Case Name: View Credits

Primary Actor: Player

Entry Condition: Player selects “View Credits” from Main Menu.

Exit Condition: Player selects “Back” to return to Main Menu.

Event Flow:

- Player wants to learn contact information of the people who developed Jungle Pursuit

- System displays contact information of software developers with their names.

Pre-conditions: Player should be in the Main Menu activity.

Post-condition: Player should be able to select Back to return to Main Menu.

Success Scenario Event Flow:

1. System displays contact information and names of people who developed Jungle Pursuit.

Alternative Flows:

- If player desires to return main menu at any time:

    1) User selects “Back” in Action Bar of android to return to Main Menu.

    2) System displays Main Menu.

2.5.3. Play Game

Use Case Name: Play Game

Primary Actor: Player

Entry Condition: Player selects “Play Game” button from Main Menu.

Exit Condition:

Event Flow:

-Player is taken to “Choose character and name” activity

-System gets name of player and player’s choice of character.

Precondition: Player should be in Main Menu activity.

Post-condition:.

Success Scenario Event Flow:

1. Choose Character and Name activity started.

Alternative Flows: none

2.5.4. Choose Character and Name

Use Case Name: Character and Name

Primary Actor: Player

Entry Condition: Player selects a game type option button from Game Type activity.

Exit Condition: Player selects “Back” to return to Main Menu.

Event Flow:

-Player chooses his character type and enters his name of choice into an EditText View

-Systemstores the player's name and character of choice.

Precondition: Default character is selected with default name.

Post-condition: Player's character and name updated.

Success Scenario Event Flow:

1. Player selects "Start Game" button to confirm his name and character selection.

2. The Start Game activity is started.

3.Player's character appears on board and name appears in bottom panel of Game activity.

Alternative Flows:

- If player desires to return main menu at any time:

    1) User selects "Back" in Action Bar of android to return to Main Menu.

    2) System displays Main Menu.

2.5.5. Start Game

Use Case Name: Start Game

Primary Actor: Player

Entry Condition: Player selects "Start Game" from Choose Character and Name activity.

Exit Condition: Player selects "Exit" to return to Main Menu.

Event Flow:

- Player names are in game panel and player characters are on board

- Game starts and player 1 get turn to roll the dice

- Character moves number of blocks according to number on dice and additional actions may occur depending on where the player's character lands

- Turn of second player starts

-Turns of the two players alternate until someone wins the game or if "Exit" option is pressed.

Pre-conditions: Game type and player characters and names are decided, along with game type.

Post-condition:

Success Scenario Event Flow:

1. One of the players reaches block 100 and the game ends with a Dialog box declaring the winner

2. The Main Menu activity is started again.

Alternative Flows:

- If player desires to return main menu at any time:

    1) User selects "Back" in Action Bar of android to return to Main Menu.

    2) System displays Main Menu.

2.5.6 Select game Type

Use Case Name: Game Type

Primary Actor: Player

Entry Condition: Player selects "Play" button from Main Menu activity.

Exit Condition: Player selects "Back" to return to Main Menu.

Event Flow: Player is given options to choose single player or multiplayer.

Pre-conditions: Default game type selected which is single player

Post-condition: Game type selection stored as input when selected using radio button

Success Scenario Event Flow:

1. Player selects game type
2. The selection is recorded and Select Character and Name activity started

Alternative Flows:

- If player desires to return to Main Menu at any time:

    1) Player selects “Return to Main Menu” button to return to Main Menu.

    2) System displays Main Menu.

2.6 Use-Case Models

This section provides information about the main use case model of Jungle Pursuit game, detailed use case explanations are included below.

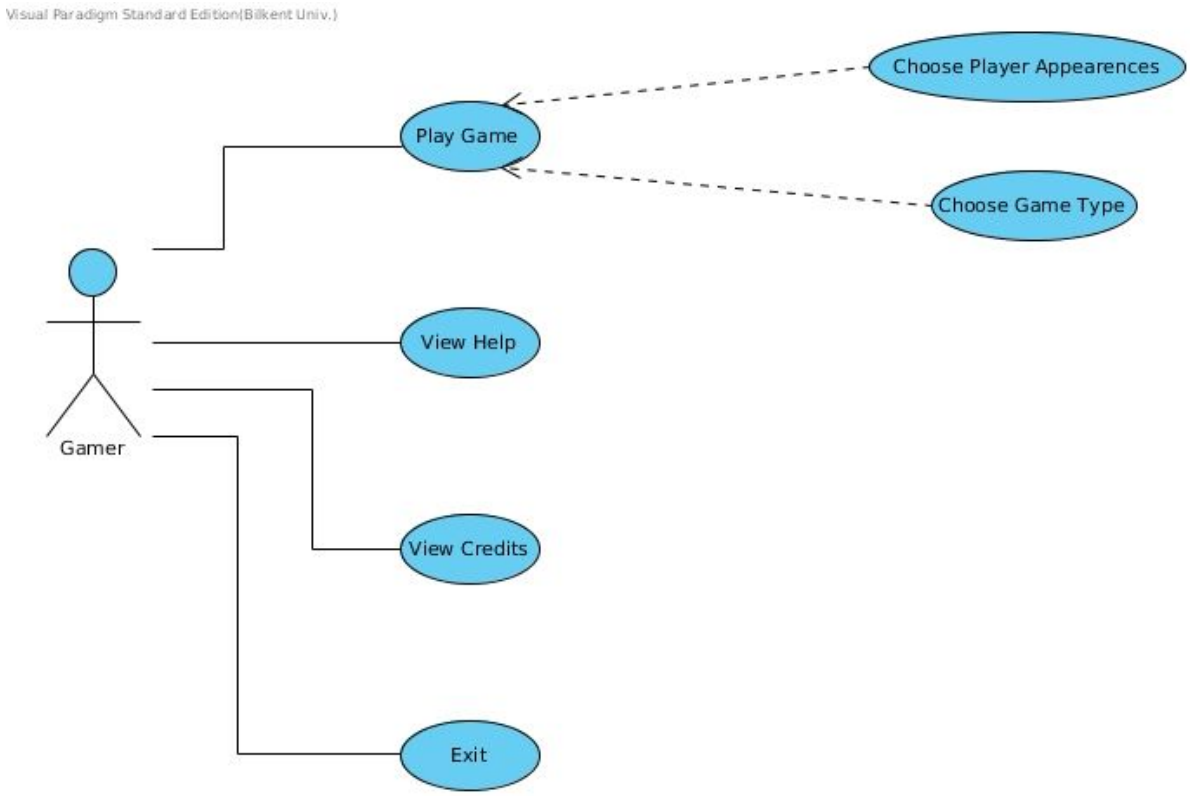

Figure 15: Use Case Model

2.7 User Interface

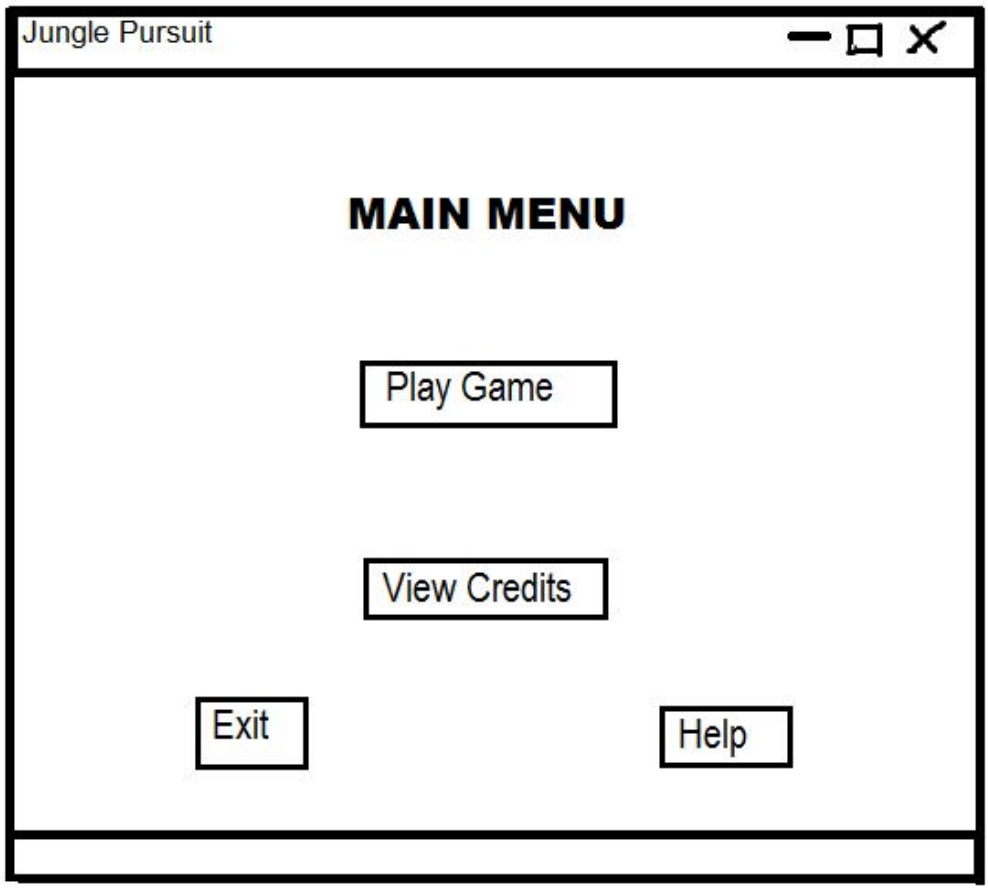

Figure 16: Main menu activity

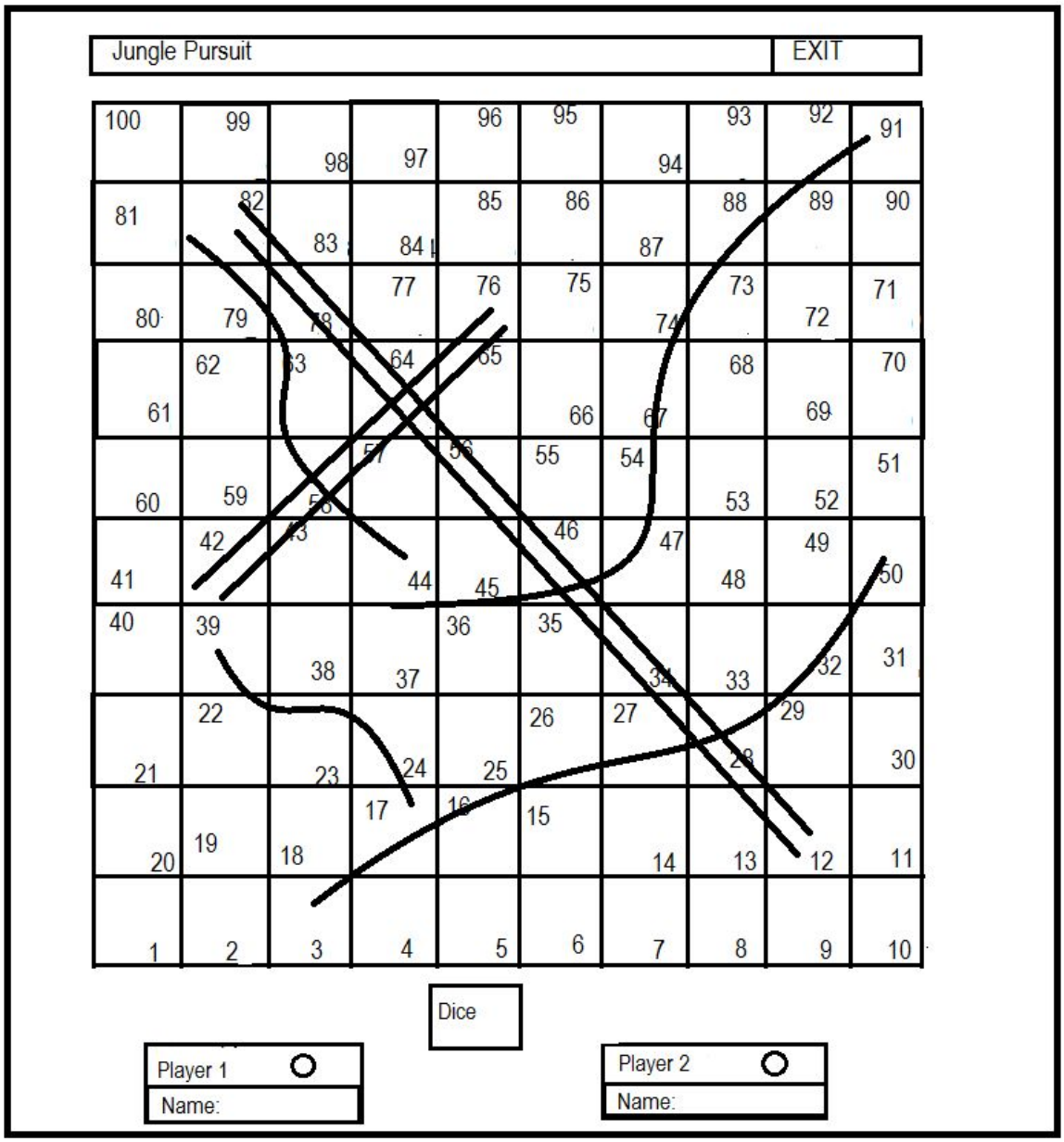

Figure 17: In game window

(That can be accepted as a draft version.)

The circle next to Player number describes the item the player currently holds

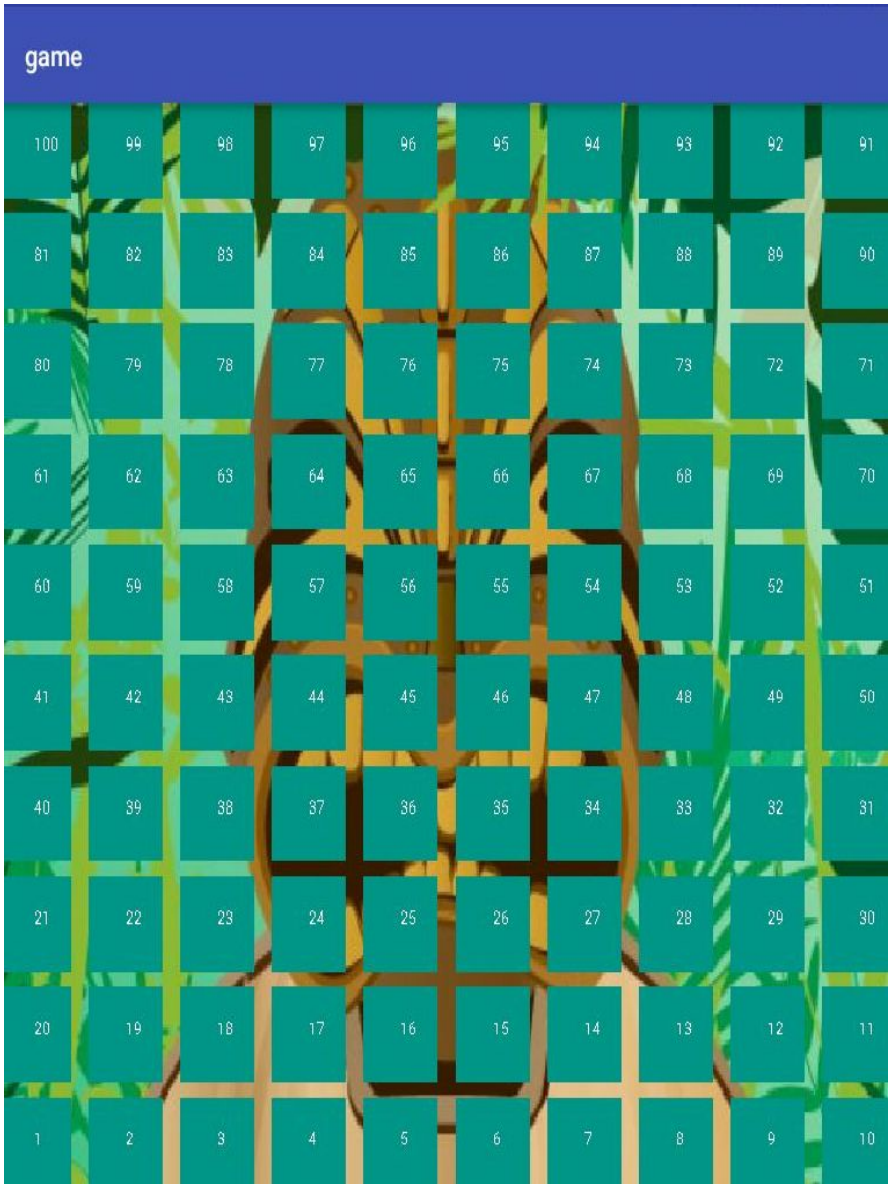

Figure 18: This is the intended version of game window that we are planning to be seen above.

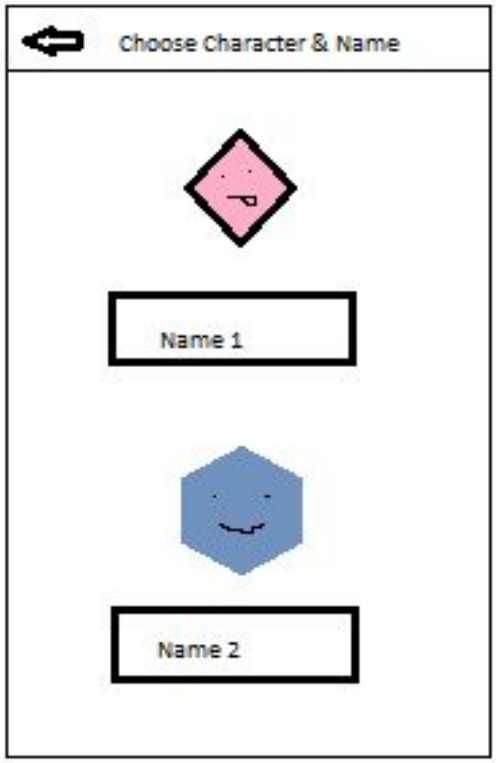

Figure 19: Choose Character and Name window

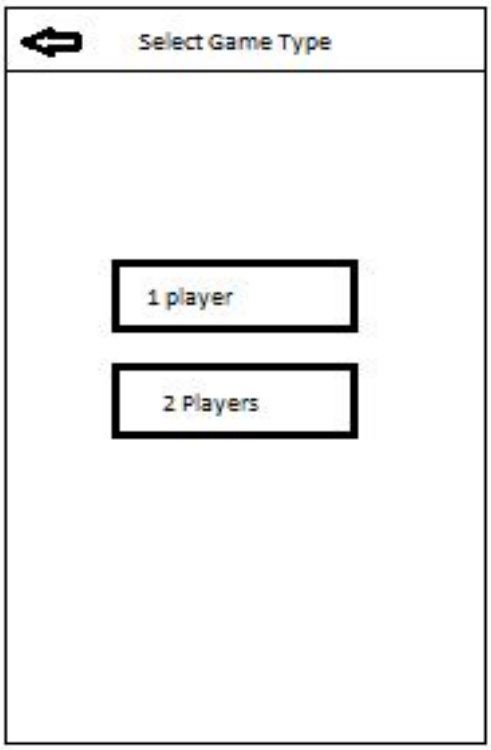

Figure 20: Select Game Type window

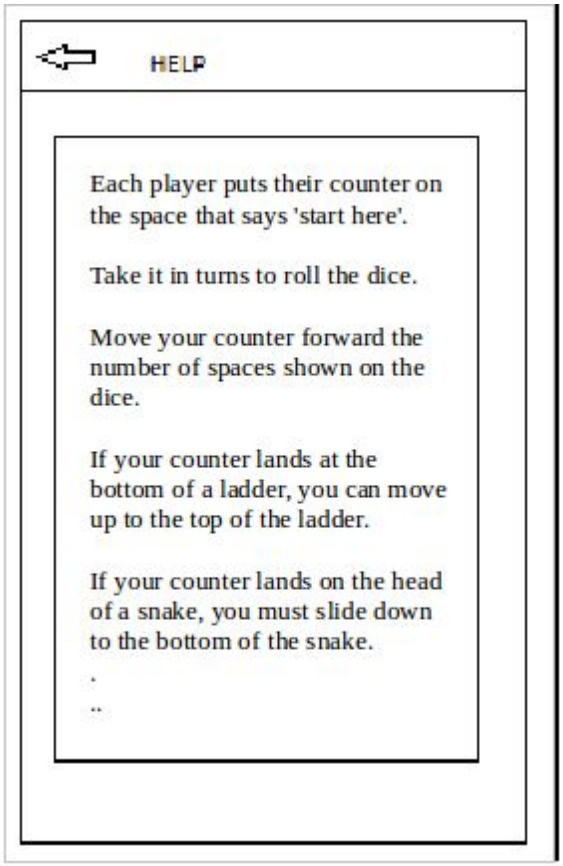

Figure 21: Help Window

Figure 22: Credits Window

2.7.1 Navigational Path

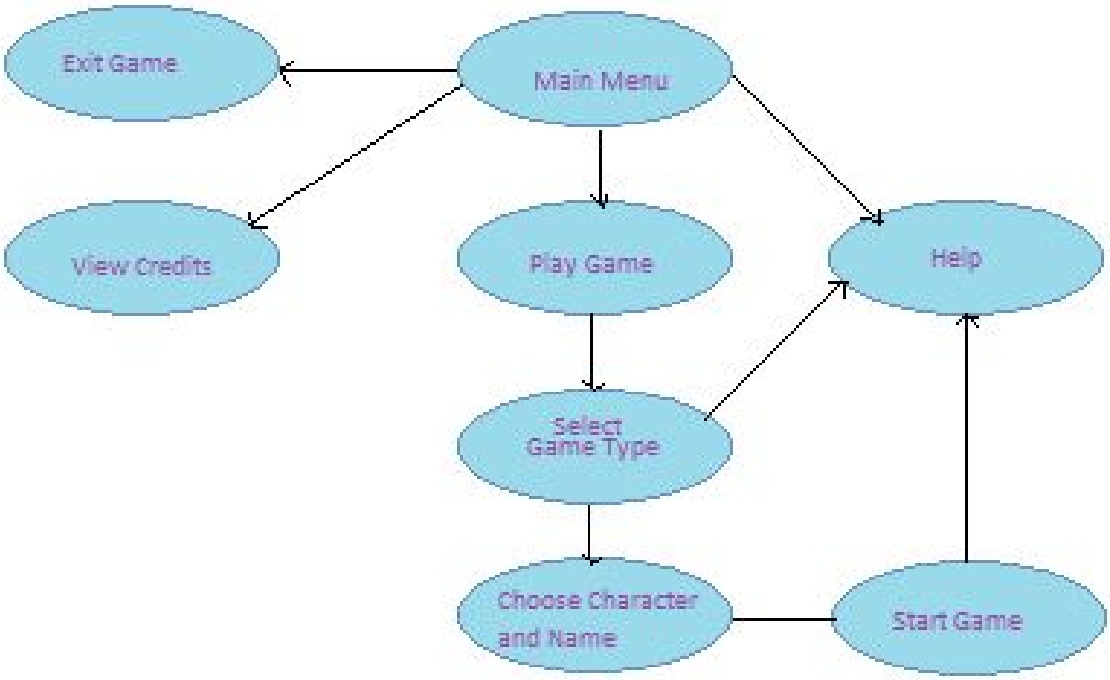

Figure 23: Navigational Path

3. Analysis

3.1 Object Model

Class Diagrams:

Jungle Pursuit consists of 10 classes that are going to be develop in terms of the requirements of the project.

Board class is the class where the main management of the game is handled. It is responsible for drawing the blocks on the screen, according to the screen size and then responsible for drawing and updating the board objects, items and players on the board. Hence without it, the other elements of the game would not exist.

Block class is an individual block that holds the item, and on which the player is on. Each block has a number attribute that tells the board which block it is. Also, the block has 'Item' attribute which when null means there is no item on the block.

Game Manager class is responsible for getting the user inputs of name, game type and character selection. It then ensures that the board is set up accordingly.

Board Object class includes the static objects that will be drawn on the board initially. It includes the snake and the vines. These objects have attributes that will inform the Board class of which block they should begin from on the board and which block they should end on in the board. Hence when the player lands on these objects he will move from start block to end 'Block'

MainMenu class handles all options of the game by contacting with the related classes. It also represents the features of our user interface.

Item class represents the items such as the oil, anti-venom and lucky banana. They have an attribute spawn point which allocates where they will spawn.

Player class is responsible for the player information and also stores the player's name and character of choice from the Game Manager class

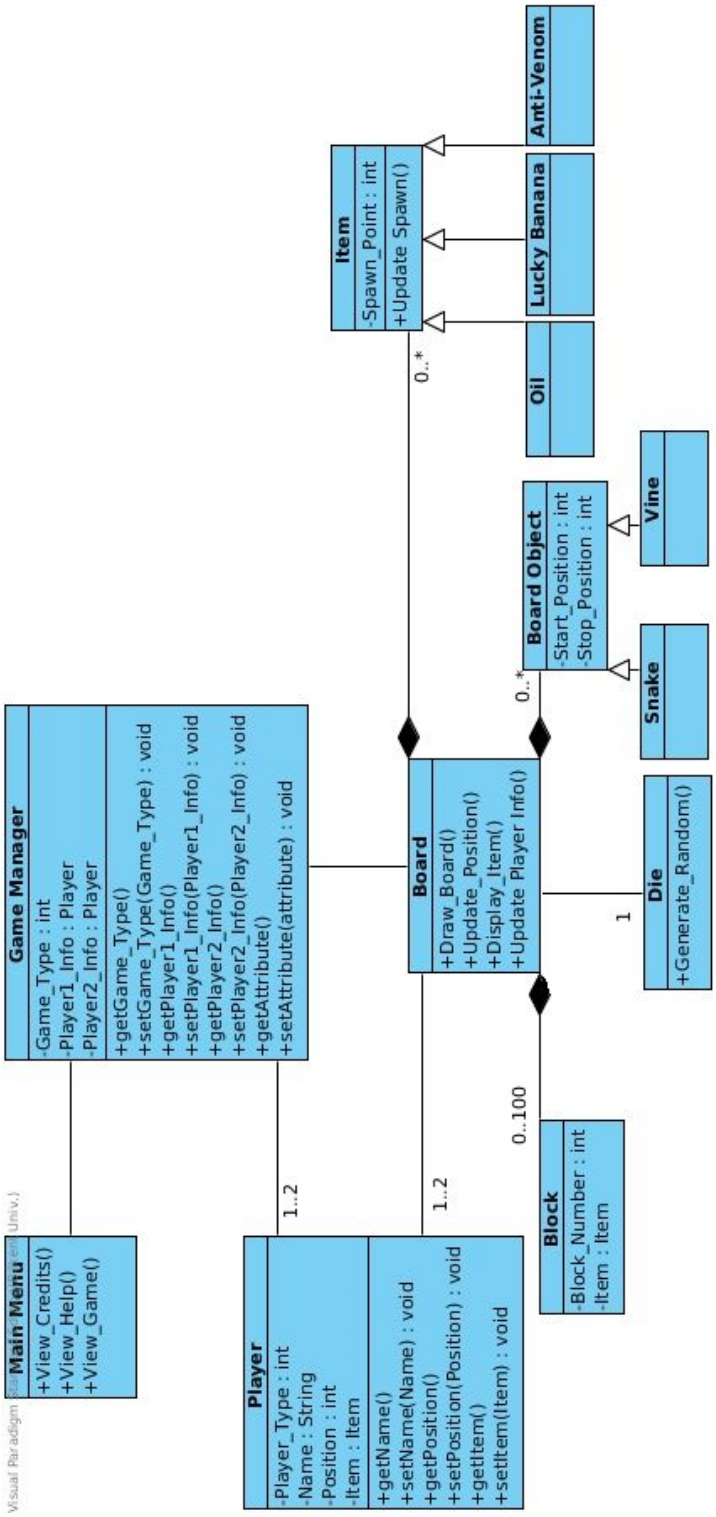

Figure 24: Class Diagram of Jungle Pursuit

3.2 Dynamic Model

3.2.1 Activity Diagram

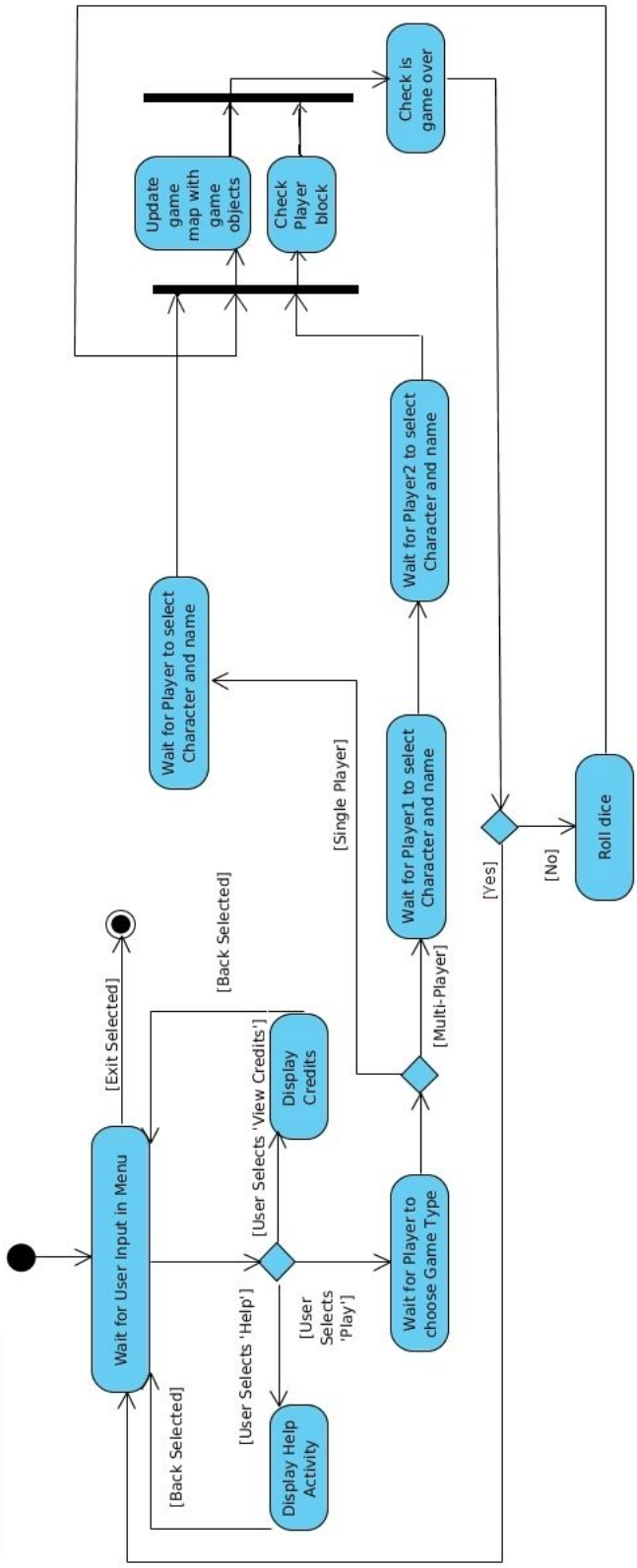

Figure 25: Activity Diagram of Jungle Pursuit

From Figure 25, it can be seen that the system is initiated and the first activity is the player selecting an option from the main menu. If he selects ‘Help’ or ‘View Credits’ he is taken to an activity from where the next activity leads him/her back to the main menu. If the player selects ‘Play’ he is taken to another activity where he is meant to select the game type. The player has the option to play single player or multiplayer. Depending on the game type selection the player inputs his name and chooses his character in the next activity, or Player 1 chooses his preferences followed by Player 2 if ‘Multiplayer’ is selected. Upon completion of game preferences the next activity is where the game board is set up according to these selected preferences. Furthermore, the position of the players on the board is also checked where the system checks if the player is on an item, or snake or vine. This occurs simultaneously as the board is being updated and any effects due to objects on the board is seen in the the next update of the board/map. The system then checks if any player has won and if the game is over or not. If it is then the system goes back to the initial activity, displaying the main menu. If not then the system waits for the correct player to roll the dice and the following activity is the update of the game board again.

Moreover, it can be seen that the exit condition can be reached from the main menu. However, due to android systems, the Jungle Pursuit can always be closed in between the game by going back to the main screen and and closing the application itself. Also, due to ‘Back’ buttons in android, the board activity can always be excited to go back to choose player or type of game. These aspects are not shown in the activity diagram as they will not need to be specifically programmed in Android Studio. However, it must be noted that the system can be exited in other ways and the player is not only restricted to exiting the game from the main menu.

3.2.2 Sequence Diagrams

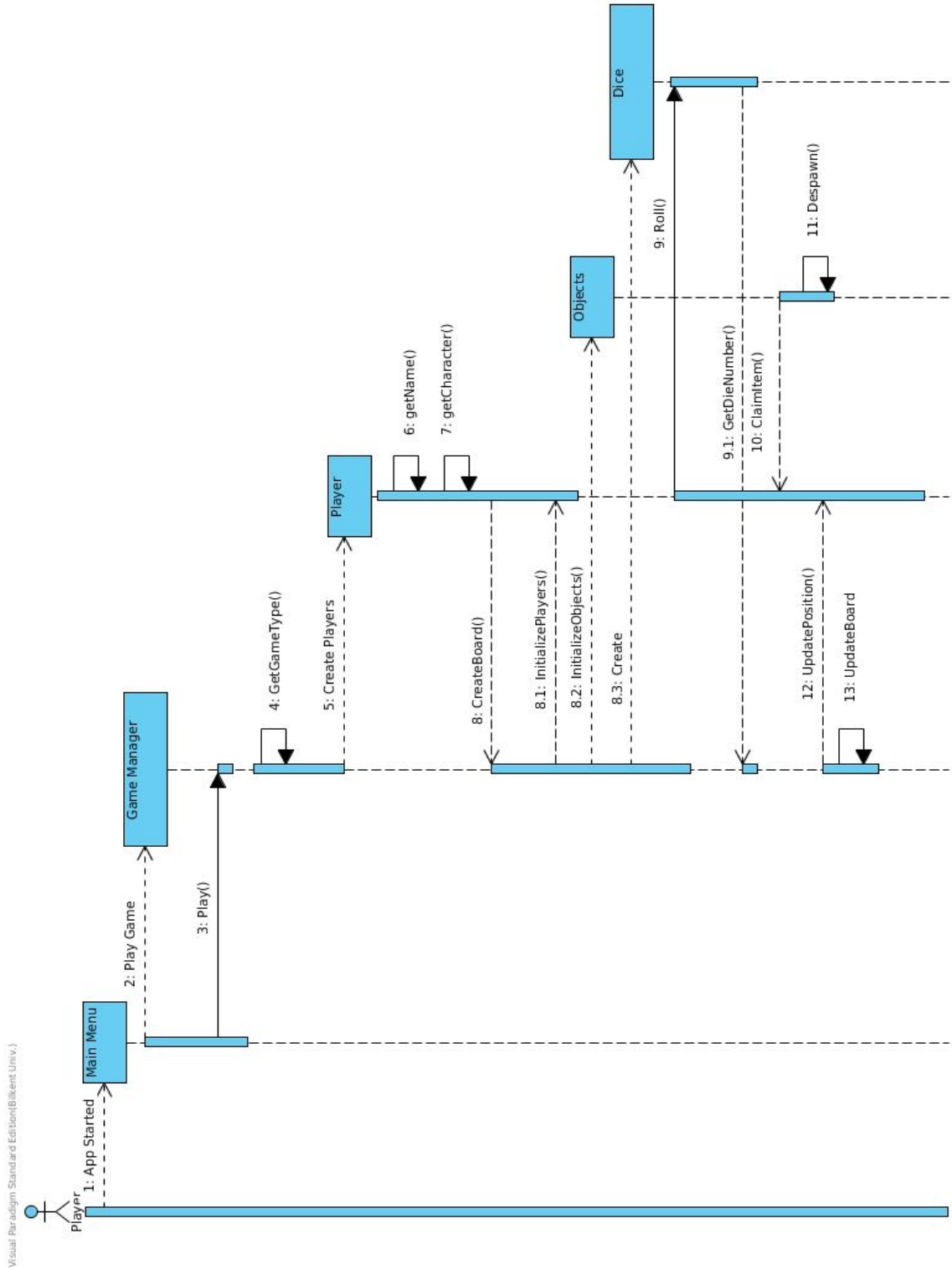

Figure 26: Sequence Diagram

The game is fairly simple when it comes to flow of events. From the main menu the player has three options, either to go forward to the help activity, the view credits or to start playing the game. Hence due to the simplicity of the game, the only sequence diagram needed is the one seen in Figure 26. No other sequence diagrams are necessary to describe the progression of events. This is because it is intuitive that when the user goes to the help activity of the android application, he/she will have no other option but to get back to the main menu. The same with the view credits activity. Therefore, to avoid unnecessary graphs with only two sequences.

From the diagram itself it can be seen that the entire sequence begins when the user clicks on the game icon in the phone's app list. The main menu sequence is achieved and after that the Game Manager class is created and the user then has to select the game type and then the player name and character. The board is then created with all the initial details and with the players at the start position, and the player positions are also updated.

Next, the dice is created and the user roll the dice and a random number is generated that is returned to the Game Manager. The game manager thus uses this number to update the position of the player on the board by also looking at the objects that have been initialised. The player position is also updated for the player in the specific class. Moreover, the item that is collected is despawns from the board and the value of that object becomes false.

4. Conclusion

In conclusion in this analysis report we aimed to describe the game we will create named as Jungle Pursuit. This analysis report has two main sections which are requirements specification and System Model.

In requirements specification, we examined almost all of the actions that a user can perform. We examined performances and indicated our functional and nonfunctional requirements regarding those performances.

In System Model Section consists of 4 parts which include;

1. Use Case Model
2. Dynamic Models *(Yet to be completed)*
3. Class Model
4. User Interface

We have decided our use case scenarios first. System Design was the second part which we have made our Sequence and Activity Diagrams. In sequence diagrams we have shown all the possible actions that a player can perform. Activity diagram demonstrates the gameplay. It shows our game components such as collectables, snakes & vines. Our class diagram also represents our implementation.

To conclude, Jungle Pursuit is a game derived from the game Snakes & Ladders which includes our own ideas and implementations. The game will be unique in the case of its attributes and theme.