Bilkent University

Department of Computer Engineering

CS 319 Project

Project short-name: Settlers of Anatolia

Analysis Report

Group members: Enes Merdane, Irmak Demir, Mehmet Alper Genç, İrem Kırmacı, Göksu Turan

Instructor: Uğur Doğrusöz
Teaching Assistant(s): Hasan Balcı, Barış Ardıç

Analysis Report
October 25, 2019

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object Oriented Software Engineering course CS319/1.

Contents

Introduction **3**

Overview **3**

Buildings 3

Tiles 4

Trading 4

Cards 4

Turns 5

Ending 5

Functional Requirements **5**

Main Menu 5

Player Specifications Menu 6

Play Game 6

How to Play 6

Settings 6

Non-Functional Requirements **6**

Usability 6

Time Constraints 7

Platform 7

System Modifications 7

Pseudo Requirements 7

System Models **8**

Use Case Model 8

Use Cases 8

Dynamic Models 11

Sequence Diagrams 11

Sequence Diagram for Building Settlements, Cities and Roads 11

Sequence Diagram for Gaining Resources 12

Activity Diagrams 13

Activity Diagram for Play Game 13

Activity Diagram for Play Turn 14

Object and Class Model 17

User Interface 19

Main Menu Screen 19

Settings 19

Game Options 20

In Game 20

End Game 21

Game Menu 21

Game Lost 22

Trade Request 22

References **23**

Analysis Report

Project short-name: Settlers of Anatolia

1. Introduction

The game we are implementing is called “Settlers of Anatolia,” which is inspired by Klaus Teuber’s famous board game Settlers of Catan, in which a number of players aim to settle an island by collecting resources, building roads and settlements with these resources and trading with each other. There is also a robber in the island who impedes the progress of the players that have a settlement near him, either by stealing from their resources or blocking resource collection. The game tracks the progress of the players using a victory point system, and the person who first reaches 10 victory points wins. Our group aims on implementing this game as a desktop platform game played with mouse and keyboard. The game will be played with bots. The programming language will be Java, and the JavaFX framework will be used to implement the graphical interface.

This report will document our progress in the analysis part of the project. The report will start by giving a brief explanation of how the game will be played. Then, the report will document the functional and non-functional requirements of the project, and afterwards will display the UML diagrams related to the analysis. Finally, the report will contain some early mockups of the game’s graphical interface.

2. Overview

2.1. Buildings

There are three types of buildings, two that can be built on vertices of the map, and one that can be built on the edges of the map.

First type of building is the settlement, which requires 1 brick, 1 lumber, 1 wool and 1 grain to build, and must be connected to at least one of the player’s roads. Another requirement for building a settlement is not to have another settlement in the neighbor vertices. In other words, a settlement cannot have neighbour settlements. Building a settlement gives one victory point.

The second type of building is the city, which requires 3 ore and 2 grain to build. A city is built on where the player already has a settlement, or in other words, a settlement is upgraded to a city. Upgrading a settlement to a city gives one extra point, or two in total.

The last type of building is the road, which is built on the edges of the map. The road requires 1 brick and 1 lumber to build, and must be connected to at least one of the player’s settlements, cities or roads. Building a road by itself awards no points, but making a chain of roads with length equal to 5 grants gives the “Longest Road” card which is worth 2 victory points. If another player builds a longer road than the

player currently holding the achievement for the longest road, then the card and the 2 victory points pass to the new holder.

2.2. Tiles

In our map, tiles are shaped like hexagons and there are six different types of tiles. The main types are forest, hills, fields, mountains and pasture which generate lumber, brick, grain, stone and wool respectively. There is also a desert tile, which does not produce any resources. At the start of the game, all tiles except the desert tile are given a number from 2 to 12 as such: Numbers 2 and 12 are only given to one tile each, number 7 is not given to any tile, and the rest of the numbers in that range are given to two tiles each. When the result of a dice roll is the number of a tile, if the robber is not on that tile, the tile will produce a resource to players that have buildings in its vertices. For each settlement, the tile produces one of its resource to the player that owns it. For each city, the tile produces two of its resource. When 7 is rolled, each player with 7 or more cards must discard the floor of half their number of cards. Afterwards, the player that rolled 7 chooses another tile for the robber to go to, and steals one random resource from a player that has a building on the vertices of the new tile.

2.3. Trading

The players can trade among themselves, or trade with the bank.

To trade with other players, a player must send out a trade request during their turn. This trade request is shown to every other player to accept or decline. The trade is made with the player that first accepts the trade.

To trade with the bank, one must provide 4 of a type of resource in exchange for one of another resource. This number can be changed for different resources by building settlements or cities on vertices adjacent to harbors.

2.4. Cards

There are 25 development cards placed randomly in a deck, which can be bought by players during their turns using resources, and can be used in their turns one turn after buying. These cards must stay hidden until their time of usage. The different types of cards are as follows:

Knight Cards: These cards allow the player to move the robber and steal one random resource from one player that has a building in the vertices of the new hexagon, similar to the case of rolling a 7. After using knight cards, they are revealed and placed in front of the player. Also, any player that reaches 3 knight cards gets a card named “Largest Army” which is worth 2 victory points. This card is passed to other players once they reach a greater number of knights than the current holder.

Victory Point Cards: These cards give the player one victory point each. They can only be revealed and used once the player has the potential to reach 10 points with them, or in other words, once the player can win.

Progress Cards: There are different types of progress cards that have different effects. They are as follows:

Road Building: These cards allow those who use them to construct 2 roads free of charge.

Year of Plenty: You can use these cards to draw 2 resources. These resources can be used to build in the turn that this card is used.

Monopoly: After using this card, you select one type of resource. All players must give all they have of this type of resource to you.

2.5. Turns

In the first turn, each player rolls the dice and the one who gets the highest result becomes the starting player and begins. The starting player places their settlement on a vertex and places a road adjacent to the settlement. Then, in the clockwise direction other players also set their settlements and roads.

In the second turn, the last person who placed his/her settlement places his/her settlement and road. Then other players place their settlements and roads in the counterclockwise direction. In the second turn, players do not have to place their roads or settlements connected to their other roads or settlements. In the second turn, players, right after placing their settlements, they gain their first resources for each hexagon terrain adjacent to the second settlement they have placed.

After these turns, starting player starts turn by rolling dice. After that, according to the total of dice values, players who has a settlement adjacent to the hexagon with the same number, gains one resource from that hexagon, and the player who has city gains two resources. Then, current player can trade with players or bank, buy an upgrade card, build settlements, upgrade settlements to cities, play upgrade cards or declare victory.

2.6. Ending

The first player to achieve 10 victory points during their turn is declared the winner and the game ends. If it is another player's turn, then they have to wait to win the game.

3. Functional Requirements

3.1. Main Menu

This is the main menu screen which will be displayed when the game icon is clicked on the desktop to open the game. This main menu will contain five buttons which are "Play Game", "Continue Game", "How to Play", "Settings" and "Exit".

With the exit button the player can quit the main menu screen and return to the desktop.

3.2. Player Specifications Menu

This is the Player Specifications Menu screen which will be displayed when the player clicks the “Play Game” button on the “Main Menu”. On that screen, the number of players will be entered by one player, usernames will be entered and colours will be selected by each player. Optionally, each player can change the avatars assigned to them by default.

3.3. Play Game

“Settlers of Anatolia” is a strategy game inspired from “The Settlers Of Catan”, which offers only one game mode as it is in the original board game. After selecting the number of players and their icons, the game starts with each player rolling the dice. During the game, players can build roads or cities according to their resources by left clicking the corresponding symbol and then selecting the area in which the settlement will be built. Also, players can send a trade request to another player or to the bank. Player who receives the trade request can accept or decline it by clicking the buttons on the window which will be opened. To win the game, the player has to reach 10 points. These points can be collected by building cities, settlements and roads, and gathering knights and collecting victory cards.

3.4. How to Play

This is the How to Play screen which will be displayed when the player clicks the “How to Play” button on the “Main Menu”. On that screen, information about the game and rules of the it will be displayed.

3.5. Settings

This is the Settings screen which will be displayed when the player clicks the “Settings” button on the “Main Menu”. On that Settings screen, background of the game can be changed by clicking “Change Background” button. Also, one of the players can turn on/off the game voice on that screen.

4. Non-Functional Requirements

4.1. Usability

Having a simple UI design is a very important issue for the project. Any player who knows the board game version already should not have problems related to the UI design while they are opening the game, changing settings, starting a game, and playing the game. To make this happen, the learning process should be easy for users at the first execution of the game. The usage should be easy to remember on the other executions as well. They should be able to find the necessary operations on the

screen without any help from someone or any written documentation. Thus, we will create our UI considering this issue.

4.2. Time Constraints

The game should not run at a slow speed so that the players have a smooth gameplay experience and not lose interest while waiting for the code to finish executing. Furthermore, the reaction time of the GUI to clicks and button presses must be half a second at most. Since Java is a language that spends a lot of time because it spends a good amount of time garbage cleaning, we are planning to write our code as efficient as possible to minimize off-time.

4.3. Platform

The game will be playable in both Windows and Mac systems.

4.4. System Modifications

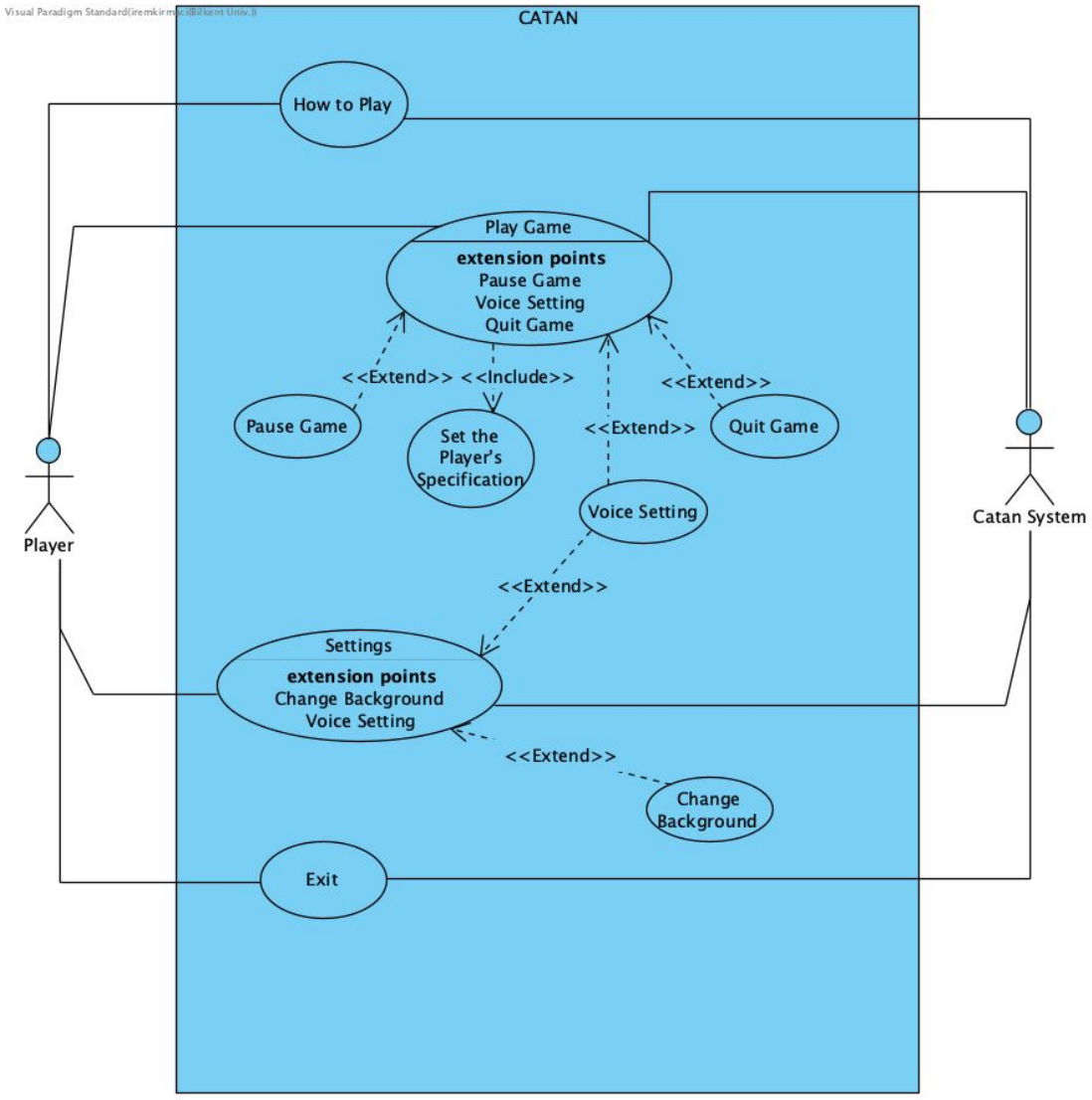
Extensibility and reusability are the important parts of the software. Therefore, this game will be implemented in a way that it can conform with new updates. It can be updated and extended according to the required changes which could be received by the user feedback or detected by a developer for a better service.

4.5. Pseudo Requirements

Java will be used in the implementation of the game. For the graphics JavaFX library will be used and due to the fact that game will be offline there is no need for another program usage to login and play.

5. System Models

5.1. Use Case Model



5.1.1. Use Cases

Use Case #1

Use Case name: Play Game

Participating actors: Player

Entry Condition: Player opens the game and clicks “New Game” button.

Exit Condition:

- One of the players reaches 10 points.
- Players want to exit game, one of the players clicks “Main Menu” button and then players return to main menu.

Main Flow:

- Player opens the game.
- Player clicks “New Game” button on the “Main Menu”.

- Configuration screen opens.
- Player provides the number of players and enter their username and select their color.
- Player clicks “Start Game” button.
- Game is prepared according to the number of players.
- Game screen opens.
- Player or one of bots reach 10 points.
- The system displays “Congratulations” message with the winner’s username or “Lost” message and asks the player if they want to play again.

Alternative Flow:

- Player clicks the “Continue Game” button.
- Player continue playing where they left off in the game.
- One of the players reaches 10 points.
- The system displays congratulations message with the winner’s username and returns game menu.

Use Case #2

Use Case name: How To Play
Participating actors: Player
Entry Condition: The player is in the “Main Menu”.
Exit Condition: The player clicks the back button.
Main Flow:

- The player opens the “how to play” menu.
- A frame containing the rules and tutorial of the game opens.

Use Case #3

Use Case name: Settings
Participating actors: Player
Entry Condition: The player opens the game and then clicks “Settings” button.
Exit Conditions:

- If the player didn’t make any changes, they exit by clicking the “Main Menu” button.
- If the player made settings, they exit by confirming the changes by first clicking Confirm and then clicking “Yes” on the frame that pops up.

Main Flow:

1.Player opens the Main menu.
2.Player clicks the Settings button.
3.The settings screen opens.
4.Player clicks Change Background button to change the background.
5.Player chooses from different background options.

Use Case #4

Use Case name: Quit Game
Participating actors: Player
Entry condition: Player is in the game.

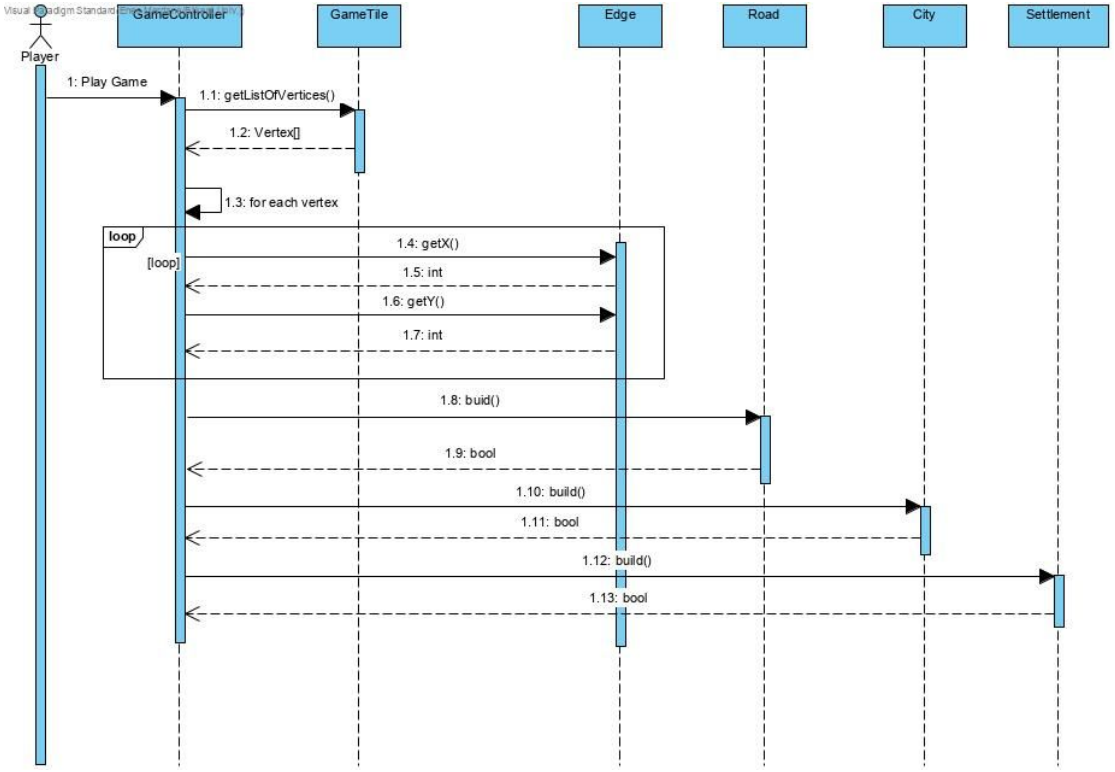
Exit Conditions: Player clicks Continue Game on the Main Menu.
1-Player wants to quit the game and go back to the main menu.
Main Flow:
1-Player clicks the Menu button while in game.
2-Player clicks “Back to Main Menu” button.
3-The system asks the player whether they are sure.
4-If player wants to go back to Main Menu, they click “Yes”.
Alternative Flow:
1-Player clicks “No” button on the fourth step and doesn’t leave the game.

Use Case #5
Use Case name: Exit to Desktop
Participating actors: Player
Entry condition: The player is in the main menu.
Exit Conditions:
1-Player wants to exit to desktop, player clicks the button Exit.
Main Flow:
1-Player opens the Main Menu.
2-Player clicks the Exit button.
3-The system asks the player if they are sure.
4-If player wants to exit the game and return to the desktop, they click “Yes”.
Alternative Flow:
1-Player clicks “No” on the fourth step and stays on the Main Menu.

5.2. Dynamic Models

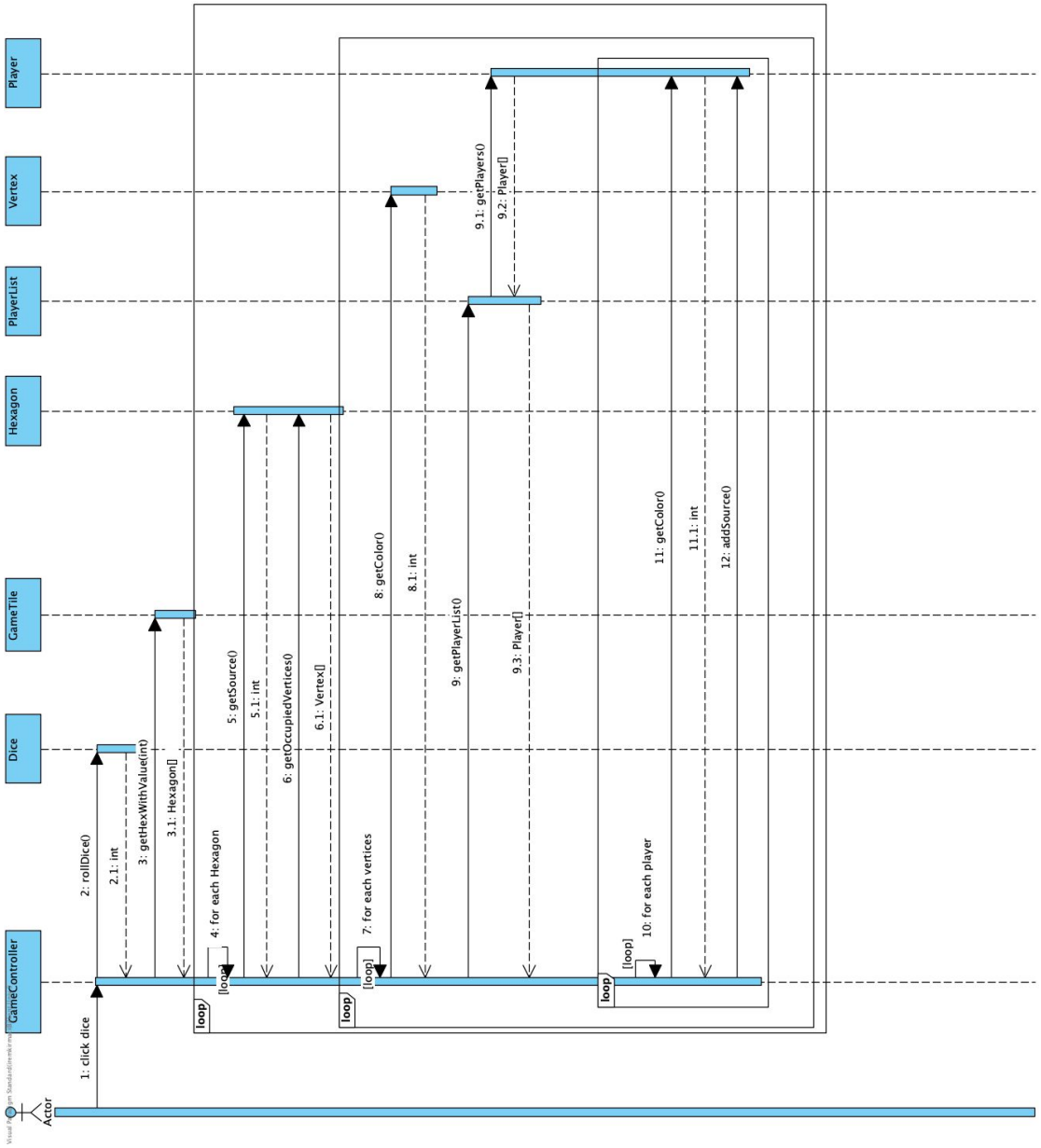
5.2.1. Sequence Diagrams

Sequence Diagram for Building Settlements, Cities and Roads



Scenario: In this scenario, the game has already started and user wants to build a road, city or settlement. After clicking on a point on the game board, the game controller receives the x and y coordinates of the vertices and edges to determine whether the player clicked on a vertex or an edge. After determining where the user clicked, the game builds a road, city or settlement.

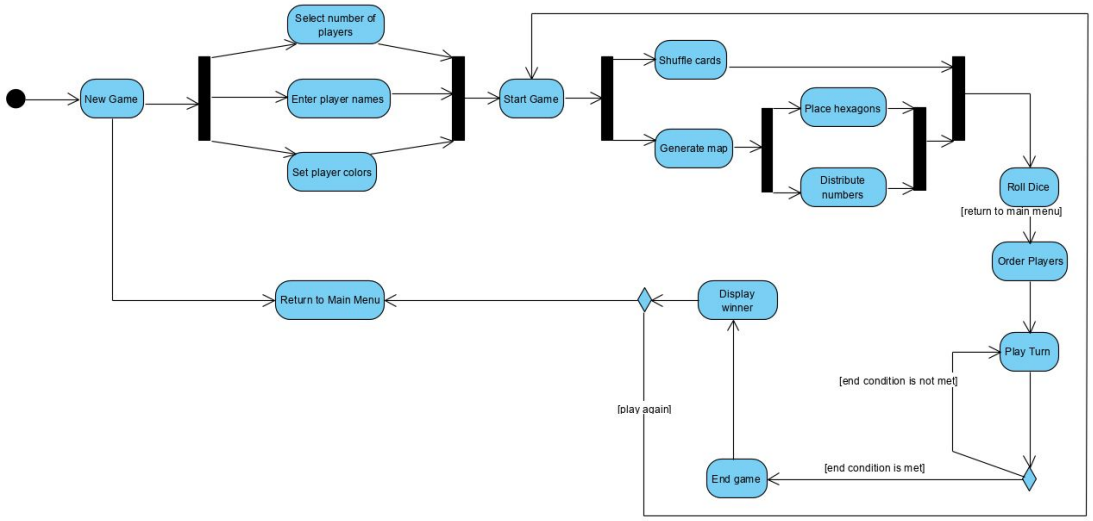
Sequence Diagram for Gaining Resources



Scenario: The game has already started and the initial buildings have been built. Then the scenario starts with the rolling of dice. After the dice is rolled and the value is obtained. List of hexagons corresponding to that value is gotten. For each hexagon in that list, its source type and the surrounding vertices and edges is returned. Then, the buildings on that vertices and edges and the owners are identified. One of that particular resource is added to each owner.

5.2.2. Activity Diagrams

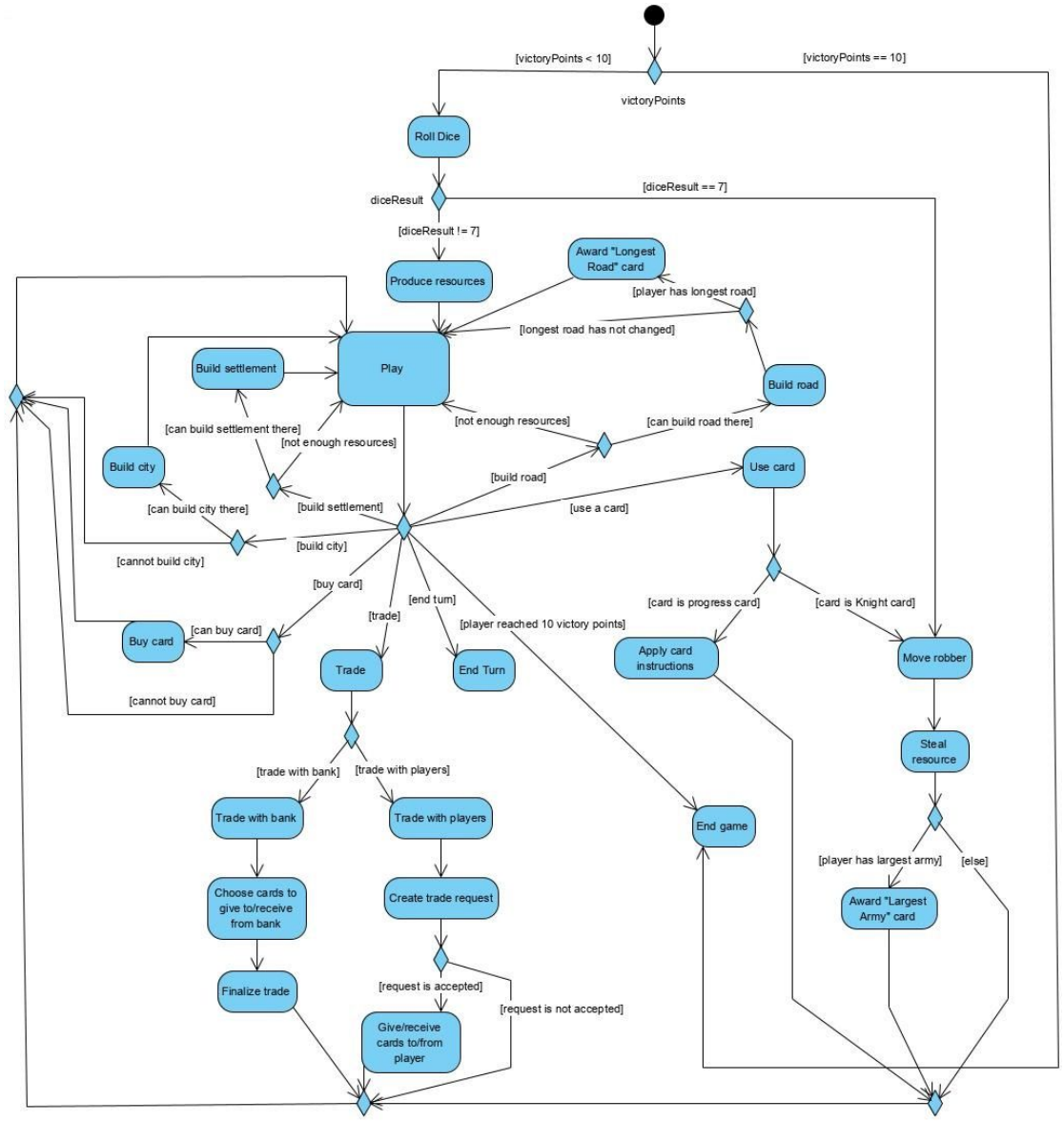
Activity Diagram for Play Game



Initially the player clicks “New Game” button on the “Main Menu”. After that, the player has to set player colors, enter their names, and select the number of players. Then, they can click “Start Game” button. This button divides into two lines. In one, the progress cards are shuffled, and in the other one, the game will generate the map in two actions, placing hexagons and distributing numbers to tiles.

When these operations are completed, the players will be able to roll the dice. After rolling the dice, the player and bots will start playing turns until the win condition is satisfied. When it is satisfied, the players will be asked whether they want to play again or not. Based on their answers, they will be directed to either the “Return to Main Menu” step or the “Start Game” step.

Activity Diagram for Play Turn



This activity diagram shows the range of activities that can be performed by a player while the turn is theirs. Initially, the game checks whether or not the victory points of that player add up to 10. If so, then the game ends and that player is declared the winner. If not, the game continues.

At the start of each turn, the players must roll the dice. No other action can be performed before rolling the dice. If the result of the dice roll is 7, then the player must select a new tile for the robber to go to, and steal resources from that hex. If not, the tiles with number corresponding to the result of the dice roll will produce

resources. Afterwards, the player can perform a range of actions. The different choices of actions are as follows:

Build Settlement

If a player chooses to build a settlement, then there are a number of conditions that must be checked:

1. The selected vertex should not contain a settlement or city.
2. The selected vertex must be connected to at least one of the current player's roads.
3. The player must have enough resources to build a settlement.
4. The selected vertex must not have a neighbor vertex with a settlement.

These conditions are grouped under the "can build settlement there" condition in the activity diagram for simplicity. If one or more of these conditions are not satisfied, the player cannot build a settlement.

Build city

If a player chooses to build a city, than this choice must satisfy these conditions:

1. The selected vertex must contain a settlement built by the current player.
2. The player must have enough resources to build a city.

As before, these conditions are grouped under the "can build city there" condition in the diagram.

Buy card

If a player chooses to buy a development card, they must have enough resources to buy a development card. If they do, then the cost will be taken from the player and the card will be added to the player's hand for future use.

Trade

There are two different ways of trading in this game: with the bank, and with other players.

Trading with bank requires that the player discards 4 of the same resource for one other resource. This number can be changed for different resources if a player has a settlement or a city adjacent to a harbor. When the player finalizes their choice, the trade is made and the player can continue with their turn.

Trading with the players involves sending a trade request containing the offer and what the player wants in return. This trade request is then sent to all other players in the game. The trade is done with the first person to accept this trade. If no player accepts this trade request, the turn continues.

Use card

Players can use the development cards they have one turn after buying them.

If the card to be used is a Knight card, then the player will move the robber and steal resources, much like rolling a 7 with the dice. Then the game checks whether or not the current player has reached 3 knights. If so, the “Largest Army” card is awarded to the player, which grants 2 victory points to its current holder. The card changes ownership once another player has more knights than the current holder. Afterwards, the turn continues.

If the card to be used is a Progress card, the instructions of this card are applied and the turn continues.

Build road

Just like building a settlement or a city, building a road also has a number of conditions:

1. The selected edge must be connected to at least one of the current player’s buildings.
2. The selected edge must not contain another road.
3. The player must have enough resources to build a road.

Again, these conditions are grouped under “can build road there” in the diagram. If all of these conditions are met, the road is built. After each time a road is built, the game checks if the current player has a longer chain of roads than the current longest, or 5 if there is no “Longest Road” award given. If the player has a longer road, they receive the “Longest Road” award.

End turn

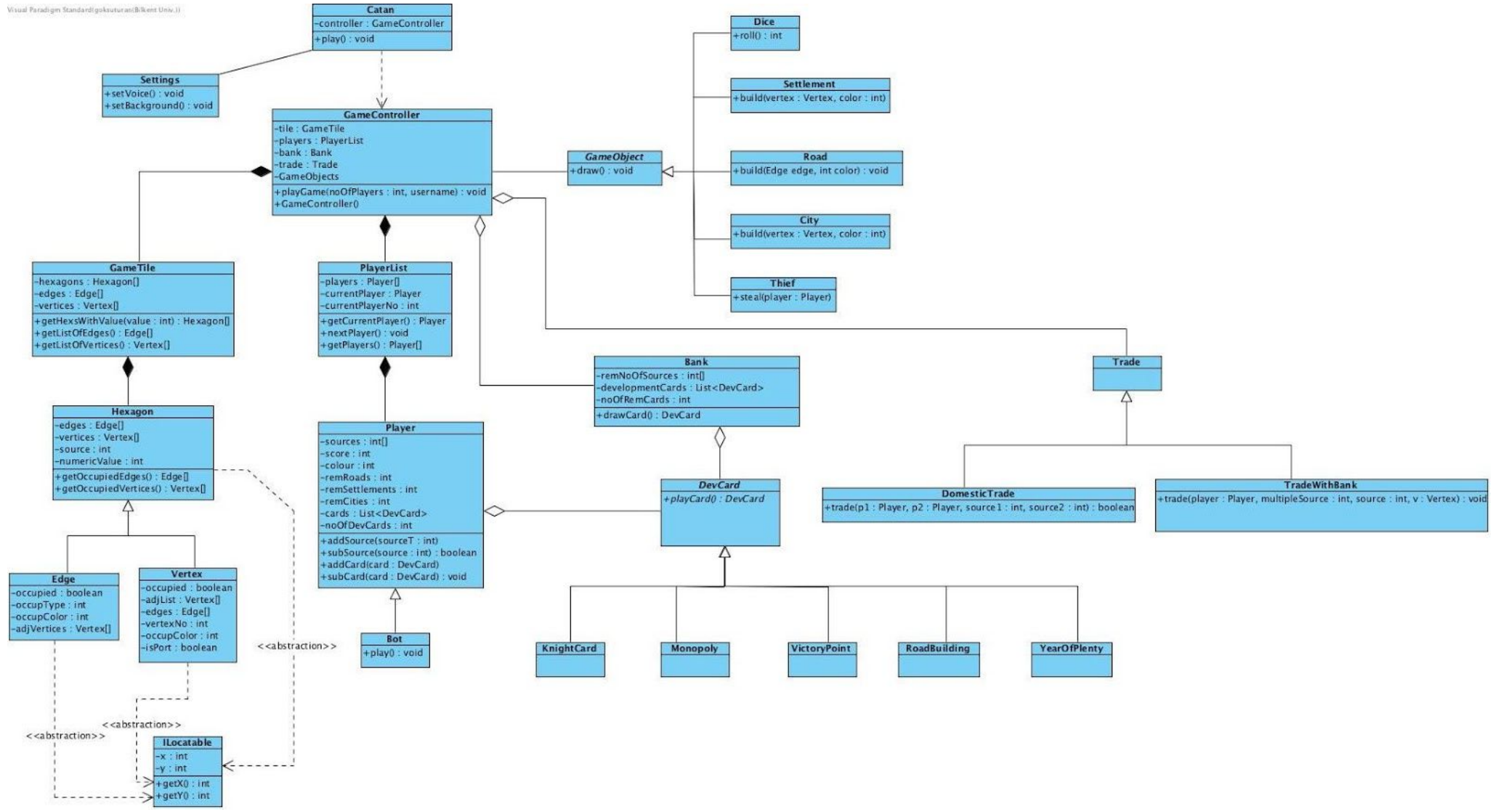
Finally, players can choose to end their turn, in which case the next player’s turn starts.

End game

If the player reaches 10 victory points in their turn, then the game immediately ends and the current player is declared as the winner.

5.3. Object and Class Model

In the UML class diagram given in the next page, the getter and setter functions for the private properties will normally be implemented but not included in the UML class diagram. The UML object class diagram was created regarding theSingle Responsibility Principle of classes.


Visual Paradigm Standard(goksuturan(Bilkent Univ.))
Catan
-controller : GameController
+play() : void
Settings
+setVoice() : void
+setBackground() : void
GameController
-tile : GameTile
-players : PlayerList
-bank : Bank
-trade : Trade
-GameObjects
+playGame(noOfPlayers : int, username) : void
+GameController()
GameObject
+draw() : void
Dice
+roll() : int
Settlement
+build(vertex : Vertex, color : int)
Road
+build(Edge edge, int color) : void
City
+build(vertex : Vertex, color : int)
Thief
+steal(player : Player)
GameTile
-hexagons : Hexagon[]
-edges : Edge[]
-vertices : Vertex[]
+getHexsWithValue(value : int) : Hexagon[]
+getListOfEdges() : Edge[]
+getListOfVertices() : Vertex[]
PlayerList
-players : Player[]
-currentPlayer : Player
-currentPlayerNo : int
+getCurrentPlayer() : Player
+nextPlayer() : void
+getPlayers() : Player[]
Bank
-remNoOfSources : int[]
-developmentCards : List<DevCard>
-noOfRemCards : int
+drawCard() : DevCard
Trade
Hexagon
-edges : Edge[]
-vertices : Vertex[]
-source : int
-numericValue : int
+getOccupiedEdges() : Edge[]
+getOccupiedVertices() : Vertex[]
Player
-sources : int[]
-score : int
-colour : int
-remRoads : int
-remSettlements : int
-remCities : int
-cards : List<DevCard>
-noOfDevCards : int
+addSource(sourceT : int)
+subSource(source : int) : boolean
+addCard(card : DevCard)
+subCard(card : DevCard) : void
DevCard
+playCard() : DevCard
DomesticTrade
+trade(p1 : Player, p2 : Player, source1 : int, source2 : int) : boolean
TradeWithBank
+trade(player : Player, multipleSource : int, source : int, v : Vertex) : void
Edge
-occupied : boolean
-occupType : int
-occupColor : int
-adjVertices : Vertex[]
Vertex
-occupied : boolean
-adjList : Vertex[]
-edges : Edge[]
-vertexNo : int
-occupColor : int
-isPort : boolean
Bot
+play() : void
KnightCard
Monopoly
VictoryPoint
RoadBuilding
YearOfPlenty
<<abstraction>>
<<abstraction>>
<<abstraction>>
ILocatable
-x : int
-y : int
+getX() : int
+getY() : int

5.4. User Interface

5.4.1. Main Menu Screen

This is the screen which the user sees after running the game.

5.4.2. Settings

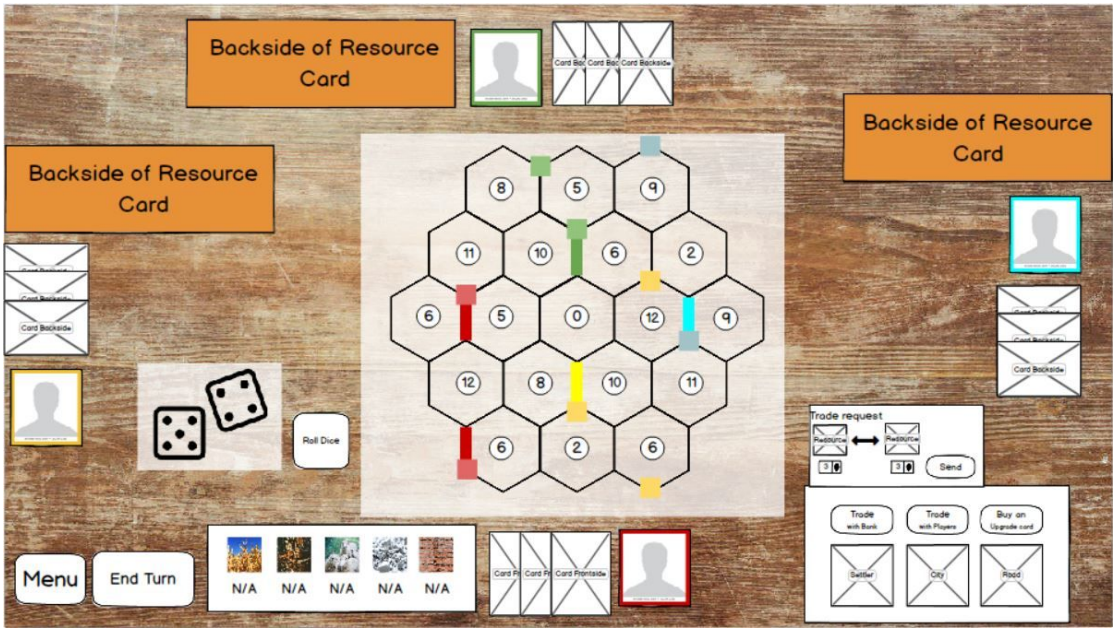
This menu is for users to adjust settings. They are able to turn the game music and the game sounds on or off, and change the background of the game from given collection.

5.4.3. Game Options

Game Options menu is the screen which opens after clicking the “Play Game” button from Main Menu. This screen is for players to set the colors and the numbers of the players of the game. After setting the options, they can start the game by clicking Start Game.

5.4.4. In Game

This screen is the game screen. It includes the game board, player avatars, the dice, resources and building costs.



5.4.5. End Game

This screen appears at the end of the game.

5.4.6. Game Menu

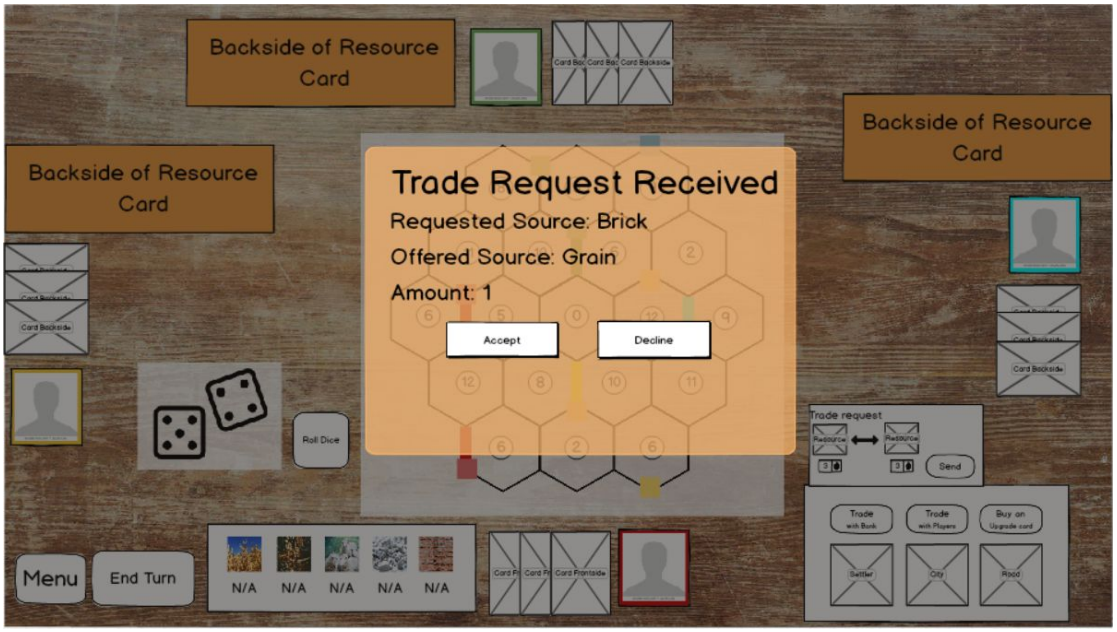
Game menu is the screen which opens when the player clicks Menu button while in game. On this screen, players will be able to set the game music and sounds on or off, go back to the Main Menu, exit game, or continue playing.

5.4.7. Game Lost

This screen appears when the player loses the game.

5.4.8. Trade Request

This is the screen that appears on a player's view when another player sends a trade request.



6. References

Inspired by the game Settlers of Catan:
https://www.catan.com/en/download/?SoC_rv_Rules_091907.pdf