Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Group Project

CS 319 Project: Monopoly Bidding War

Analysis Report

Deniz Kasap 21702460, Irmak Çeliker 21702502, Batuhan Budak 21704212,

Kübra Okumuş 21600980, Ömer Faruk Kayar 21602452

Instructor: Eray Tüzün

Teaching Assistant(s): Emre Sülün, Barış Ardıç and Elgun Jabrayilzade

Progress/Analysis Report Iteration 1

Oct 28, 2020

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.

Contents

Analysis Report

CS 319 Project: Monopoly Bid Wars

1 Introduction

Monopoly Bidding War is a turn based multiplayer strategy game. Each player will have a distinct pawn which will have the ability to move around the map by rolling dice. According to the squares they land on, they will purchase assets by auction, collect rent from the assets, draw cards, lose, or gain money throughout the game.

The main goal of the game is to purchase as many assets as possible while building houses or hotels on properties and to collect rent from the other players to bankrupt them. Besides owning assets, players can collect money from the chance or community cards, by landing on free parking or by selling their assets. Also, players may lose money in many different ways along with paying rent. Chance or community cards can make players pay money as well as pay tax squares or landing in jail. There is no limit of rounds, the game ends when there is a winner. The last player who does not end up bankrupt or surrendered will be the winner of the game.

2 Overview

2.1 Game Play and Control

The game will be online and will have two different player options: player and a host. The line-up between the players will be made by rolling dice. According to the line-up, each player will roll the dice on their turn to move on the map that contains 10 different types of squares: Go, Property, Utility, Chance, Community Chest, Pay Tax, Free Parking, Railroad, Go to Jail and Jail. By landing on these squares, players will be able to buy properties or utilities and keep on gaining or losing money throughout the game depending on the square they landed. The main goal is to lose a minimum amount of money while owning the maximum amount of properties to gain money and bankrupt all the other players. When there is only one player remaining who has not bankrupted or surrendered, he or she will be the winner of the game.

The main control is maintained by the mouse and the keyboard. Besides entering an amount for a property, the keyboard will not be used. Every other operation will be done by clicking.

2.2 Map

The map will be 11 x 11, containing 10 different types of squares: Go, Property, Utility, Chance, Community Chest, Pay Tax, Free Parking, Railroad, Go to Jail and Jail on the sides. In the middle, Chance and Community Chest cards will be settled, along with a space to pay tax and jail money on.

2.2.1 Go

Go square is the starting point of the game. After the first round, every player who passes or lands on this square will be rewarded with $200. Passing the Go square means starting a new round for the players.

2.2.2 Property

Properties are meant to be bought and collected by the players. There will be 22 Property squares consisting of 8 different colors and initial prices, the value and the rent of each property will become higher respectively from the start to the end.

If the player lands on an unowned property, an auction screen will pop up automatically, so that every player will have a chance to purchase that property.

If the player lands on an owned property, he or she will pay rent to the owner.

If the player lands on to his or her own property, he or she will have a chance to build a house or to do nothing.

2.2.3 Utility

There will be 2 Utility squares, same as properties they are meant to be owned and the price also increases from start to end. The same rules apply for landing on a Utility; however, players cannot build houses on Utility squares.

2.2.4 Chance or Community Chest

There will be 3 Chance and 3 Community Chest squares. When the player lands on one of these squares, by clicking on the pile found in the middle of the map they will be able

to draw a card. According to the content of the drawn card, the player may lose or gain money, be sent to jail, or have a chance to get out from jail without paying any money.

2.2.5 Pay Tax

There will be 3 Pay Tax squares, these squares will show the amount of money to be paid. The person who landed on any of these squares will have to put the specified amount of money on the middle of the map.

2.2.6 Free Parking

If a player lands on Free Parking square, he or she will collect the money found in the middle.

2.2.7 Go to Jail

Go to Jail square is located on the corner of the map and sends the players to Jail where it is located on the cross corner of the square.

2.2.8 Jail

When a player lands on the jail by rolling dice, they are not counted as in jail and keep on playing on the next round.

There are three ways to end up in jail: by rolling doubles successively for 3 rounds, by landing on Go to Jail square or by drawing the Go to Jail Chance or Community Chest card. People who land on this square will also have three options to get out of jail.

 After ending up in this square, the player can try rolling doubles on the next round to get out of jail, if it is a success they can keep on playing, if it is a failure they keep on waiting. However, a person can only try rolling doubles for 3 rounds, if it is still a failure in the third round, they need to pay $50 to the middle and can start playing in the next round.

As a second option they can pay $50 to the middle any of the next 3 rounds and keep on playing on that round without trying to roll doubles.

The last option is to use the Get Out of Jail Free card. The players can own this card by drawing on the previous rounds or by buying or trading from another player who owns this card. The player can return this card and keep on playing on that exact round.

2.2.8 Railroad

Railroad is also an asset which can be purchased by the players, the rent collecting will be the same as utility. There will be 4 of them and players will not be able to build houses on these assets.

2.3 Players

All the players will join the game online, there will be no computer players actively playing the game. The minimum limit of players will be 2 and the maximum will be 4.

2.3.1 Host Mode

If a person chooses this mode, they will start a new game.

2.3.2 Player Mode

If a person chooses this mode, he/she will be added to a suitable host's game.

2.3.3 Banker

The banker will not be from one of the players, this role will be played by the computer.

2.4 Items

2.4.1 Property Cards

Property cards will display the rent, house or hotel building and the mortgage value of the particular property. The rents will change according to the number of houses or hotels on that property and it will be displayed by these cards.

Besides the owner of the property, other people will not be able to see the content of the card. However, every player can see the property cards owned by the other players to be able to send trade or buy requests.

2.4.2 Chance or Community Chest Cards

They will be located at the center of the map. People who land on Chance or Community Chest squares will click here to draw a card. Every card will have a different content, such as get out of the jail free, go to jail, pay tax, collect money etc.

2.4.3 Pawns

There will be 8 different pawns. Each player needs to choose a pawn that has not been chosen by another player. Pawns will move across the map according to the output of the dice, so that players will be able to follow where they are located.

2.4.4 Money

Each player will start the game with $1500. People will be able to track each other's money amount. Collecting rent and purchasing will be done by it.

2.4.5 Utilities and Railroad Cards

Utilities cards will carry almost the same characteristics with property cards. There will be 2 different utilities and 4 different railroad cards, each of the cards will display the rent and the mortgage value. However different from the property cards, the more utility and railroad cards a player collects, the higher rent they will receive since there is no build house option.

Same as property cards, people will be able to see the belongings of other players without seeing the content.

2.4.6 Houses and Hotels

Houses and Hotels will be built by the players to increase the rent of a property. There will be 32 houses and 12 hotels, if all these houses and hotels have been purchased, people will not be able to buy any other.

They will be displayed on top of the property squares, so that every player will be able to follow the game.

2.4.7 Dice

There will be two dice and players will be able to move on the map by rolling them. If it is the player's turn, the roll dice screen will pop up and when the player clicks on it, the pawn will move according to the random output given by the dice.

2.5 Buying Properties

The main method for purchasing a property will be made by performing an auction. When a player will land on an unowned property or utility, the auction will start automatically. A request for entering the auction will pop up on players' screen. If a player decides to enter, a screen for bidding will be displayed so that players will be able to enter their offers. Players will not see each other's bids and after the auction is concluded, the player who has landed on the square initially will be notified if he/she won or lost the auction. A second chance will be given to the player to make another bid. Also, if there are two or more players who have entered the same amount of money and one of them is the player who landed on the square, he/she will be the automatic winner. Else, the auction will keep on going between the remaining players, until one of them wins. After the auction is concluded, the winner of it will be displayed on every players' screen and the winner will be able to purchase the property by paying the last bid he/she has entered to the bank.

The rules for bidding are simple, every player will offer a higher amount of money than the initial value of the properties and the minimum amount to increase a bid will be $1.

The same rules apply when a player wants to sell his or her properties, only with one exception: none of the players will have a priority or will be given a second chance.

Reserving: Players will be given the right to reserve a single and an unowned property or utility in every round. When the player passes the Go square the reserve right will get back up to one. However, the players will not be able to purchase any other property after they purchase their reserved ones. Also, players will be given the chance to give up their reserved square but their right to join the auction for that property will be taken away in that round.

2.6 Paying Rent

Players need to pay rent if they have landed on an owned square. If the player who will pay rent has not enough money, he/she cannot send a trade request or become a debtor. The player may try selling his belongings or putting his/her properties on mortgage and make the payment on that exact round.

2.6.1 For Properties

Properties' rents depend on the number of the houses. As the house number increases, the rent also increases.

2.6.2 For Utilities and Railroads

The rent of these are affected by two factors: the output of the dice and the number of utility or railroad cards the owner has. Five times the number of utility or railroad cards the owner has and the output of the dice will be multiplied in order to calculate the rent. For example, if the player rolled 9 and the owner has 2 utility cards, the rent will be 2 x 5 x 9, which equals $90.

2.7 Building Houses or Hotels

As mentioned before, there will be 32 houses and 12 hotels, and when all of them are being used in the game people will not be able to build houses anymore. There are three conditions to build a house on a property: a player needs to be located on that square, has to build only a house at one time and he/she needs to own all the properties with the same color. A player cannot build houses unevenly, which means after building a house he/ she needs to build the same number of houses on the other properties with the same color. For example, a player will not be able to build the second house on a property without having a house on the other properties of the same color.

5 houses equal 1 hotel, so if a player wants to build a hotel, he/ she needs to build 4 houses. When a player builds the 5th house, it automatically becomes a hotel, and the other 4 houses are returned to the bank. The maximum number for building a house on a single property is also 5.

2.8 Selling Properties and Trading

Players will have the chance to send each other requests to trade or buy the belongings of the other players if it is their turn. The player can offer money or his/her belongings in return of the desired properties or cards. After that the other player will receive the offer in his/her screen and can either accept or deny the request, however if the other player wants to negotiate, he/she can wait for his/her turn and make a new offer to the player.

Players can sell their properties only on their turns. If the player decides to sell a property, an auction will take place and the winner will be able to purchase the property. Detailed rules for the auction are explained in 2.5.

2.9 Bankruptcy

When a player loses all his/her money and owned properties, or has all his/her belongings on mortgage with no money, the player will be declared as bankrupt. A bankrupted player will not be able to play and loses the game, however he or she can keep on watching it.

2.10 Mortgage

If a player does not want to lose his/her property or utility and needs money, he/she can put his/her properties on the mortgage. The player will receive the specified mortgage amount of the property by the bank; however, he/she will not be able to collect rent until the property is unmortgaged.

If the player decides to unmortgage the property, he/she will need to pay the mortgage amount plus %10 interest.

2.11 Surrender

When the players want to quit playing, they can choose the surrender option and stop playing. After a player surrenders, his/her belongings will be returned to the bank, so the properties or utilities he/she has will be opened to purchase.

2.12 Winning

When all the other players have bankrupted or surrendered, the last person to remain will be declared as the winner.

3 Functional Requirements

3.1 Play

Players should click Play in order to start playing the game. When the player clicks this button, three new options will pop up: Join, Host and Back.

3.1.1 Join

Players who choose to Join will enter an id given by a host. If the id is not valid a warning message will be displayed until the player enters the right id.

3.1.2 Host

If a person chooses this option, he/she will be able to start a new game. Hosts will enter an id for the game and share it with other people he/she would like to play with.

3.1.3 Close icon

This button returns the player to the main menu.

3.2 Settings

Settings option will include Resolution, Master Volume, BGM Volume, SE Volume and Apply. Players will be able to adjust the resolution and the volume of the game.

- Resolution will be adjusted by clicking one of the offered options.
- Volume settings will be adjusted by sliding the volume controller to the desired point.

By clicking the Apply button players will be able to save their setting changes.

3.3 How to Play

The players who would like to learn how to play or to check a rule can click this button. After choosing this option, a screen that displays detailed instructions for playing the game will pop up.

3.4 Credits

By clicking on Credits, players will be able to see the creators of the game. A screen that displays the name of the creators will pop up.

3.5 Exit Game

If the player decides to not play, he/she can click this button. After clicking this button, the game will shut down and the player will be returned to the desktop.

4 NonFunctional Requirements

4.1 In-Game and Server Performance

We are using a database to interact and exchange data between users. Data flow between database and computers are really constricted, without any loss of information. Some parts of the game works only on the host's computer, and it distributes the crucial and necessary information through the database. Joined computers take the information and display them accordingly. This feature makes the game faster and fluent.

4.2 User-Friendly

The objects and details of the game have smooth graphic designs. We are planning to please the eyes of users as much as possible with soft drawings and chirpy colors. We have some options in order to maintain the control of the user as giving them more flexibility. Besides, users will not be trapped in huge amounts of buttons and control units. We will sustain the simplicity so as not to cause confusion.

4.3 User Interface

We adopt the thought that we need to protect the traditional monopoly structure. However, we tried to expand it with some animations and ideas. In game, cities (squares of the board) are demonstrated as 3D objects. In a turn, when a card is played, as a chance card or as a property, it is displayed fully then the game goes on. Players can trade their cards and money through a trade window. There are some compatible ideas like these in order to give the best experience to the players.

4.4 Extendibility

4.4.1 More Players

In the first release, we will have multiplayer mode that can be played only with shared id. We are planning to add a platform where the player can join some public servers, and meet strangers online to enjoy the game. Having strangers getting together will require communication through the game, so there can be a chat and a voice chat too.

4.4.2 Single Player Mode

With the help of artificial intelligence, we are planning to add a single player mode which can be played with bots. These bots will have at least three levels of difficulties. Users can develop their skills without any frustration due to skill level difference.
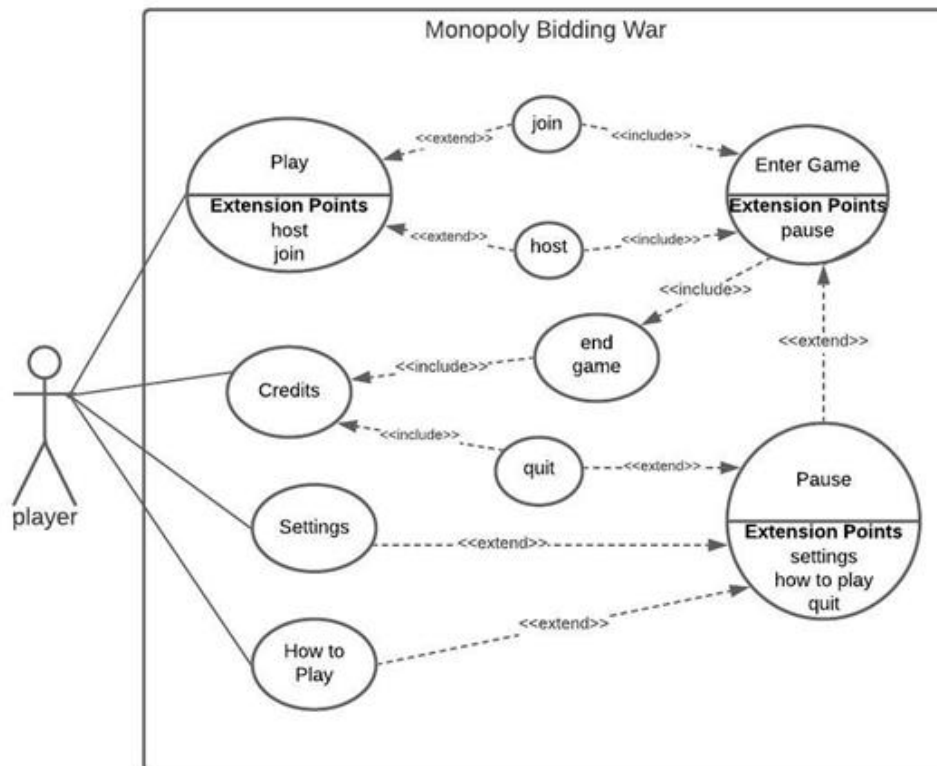
4.4.3 Customization

We are planning to add more tables and carry this adventure to another level. Players will enjoy playing in their nation's cities, also they will be able to select their national animals and traditional objects as pawns. We are planning to customize the dice as well.

5   System Models

5.1 Use-Case Model

Use Case Descriptions

Use case Name: Play

Participating actor: Player

Entry condition:

·   Player is in the main menu and selects the "Play" button.

Exit condition:

·   Player clicks the close icon and returns to the main menu.

·   Player enters the game by clicking the "Enter Game" button.

Main Flow of events:

1.   Player selects the "Play" button on the main menu.

2.   Player chooses the "Join" option.

3.   System displays a screen that asks the player to enter a nickname and the id of the game s/he wants to join.

4.   Player enters a nickname that s/he chooses, and the game id which is chosen by the host player.

5.   System checks whether the given game id is matched with an existing game id.

6.   A proper id is entered, so the system enables the "Start Game" button.

7.   Player pushes the "Start Game" button.

8.   Game starts.

Alternative flow of events:

2. a. Player chooses the "Host" option.

   1.   System opens up a screen where the player can write a nickname and a game id.

   2.   Player enters a nickname and a game id which s/he chooses.

   3.   Event flow continues from the step 5 of the main flow of events.

2. b. Player changes his/her mind and presses on the close icon.

   1.   System navigates the player to the main menu.

6. a. The game id is incorrect.

   1.   System displays an error message.

   2.   Event flow continues from the step 2 of the main flow of events.

Use case Name: Enter Game

Participating actor: Player

Entry condition:

·   Player enters the game id as a host player, chooses a pawn and enters a nickname and then clicks the "Enter Game" button.

·   Players must enter the correct game id as a joint player and click the "Enter Game" button.

Exit condition:

·   Player selects the "Pause" menu, and then clicks the "Quit Game" button.

·   Player wins the game by being the last player who has money.

Main Flow of events:

1. The screen for choosing a pawn pops up.

2. Player chooses a pawn.

3. Player clicks the "OK" button.

4. Player rolls dice for the line-up.

5. The game starts.

6. The game ends.

7. Player is returned to the main menu.

Alternative flow of events:

2. a. Player chooses an available pawn.

    1. Events continue from the step 3 of the main flow of events.

3. b. Player chooses an unavailable pawn

,   1. A warning message is displayed.

    2. Events continue from the step 2 of the main flow of events.

6. a. Player wins the game.

    1. A winning message pops up on the screen.

    2. System navigates the player to the credits screen.

6. b. Player loses the game.

    1.A losing message pops up on the screen.

    2. System navigates the player to the credits screen.

8. c. Player quits the game.

    1. System navigates the player to the credits screen.

Use case Name: Pause

Participating actor: Player

Entry condition:

·   Player must be in the game screen and clicks the "Pause" button.

Exit condition:

·   Player selects the "quit" button and returns to the main menu.

·   Player clicks outside of the pop-up screen of the pause menu and returns to the game.

·   Players may also return to the game by clicking the "Resume" button on the pause menu.

Main flow of events:

1. Player selects the "Pause" button.

2. System shows the pause menu as a pop up screen.

3. Player chooses the "Settings" option.

4. Settings screen is displayed.

5. Players may make changes on the settings.

6. Player returns to the pause menu by clicking the close icon.

7. Player closes the pause menu screen by clicking the "Resume" button or outside of the pop-up screen.

Alternative flow of events:

3. a. Player selects the "How to play" option.

       1. Player reads the instructions of the game.

       2. Event flow continues from the step 6 of the main flow of events.

3. b. Player clicks the "Quit Game" button on the pause menu.

       1. System displays a pop up screen that shows the ranking of the player.

       2. Player presses any button on the screen.

       3. Pause use case ends and the system navigates the player to the Credits screen.

Use case Name: Settings

Participating actor: Player

Entry condition:

· Player is in the main menu and clicks the "Settings" button.

· Player is in the pause menu and chooses the "Settings" option while the game continues.

Exit condition:

· Player returns to the main/pause menu by clicking the close icon.

Main flow of events:

1. Player makes changes in the settings.

2. Player saves changes by clicking the "apply" button.

3. Player clicks the close icon and the use case ends.

Alternative flow of events:

1. a. Player does not change anything.

1. Events continue from the step 3 of the main flow of events.

2. a. Player changes her/his mind and does not apply the changes.

1. Events continue from the step 3 of the main flow of events.

Use case Name: Credits

Participating actor: Player

Entry condition:

· Player is in the main menu and clicks the "Credits" button.

· Player is in the pause menu and presses the "quit" button. S/he will proceed to credits afterwards.

· Player is in the game and the game is finished. S/he will proceed to credits afterwards.

Exit condition:

· Player proceeds to the main menu by clicking the "Main Menu" button.

Main flow of events:

1. System opens the screen that displays information about the developers and references.

2. Player clicks the "Main Menu" button and the use case ends.

Use case Name: How to play

Participating actor: Player

Entry condition:

· Player is in the main menu and clicks the "How to play" button.

· Player is in the pause menu and chooses the "How to play" option while the game continues.

Exit condition:

· Player returns to the main menu by clicking the close icon.

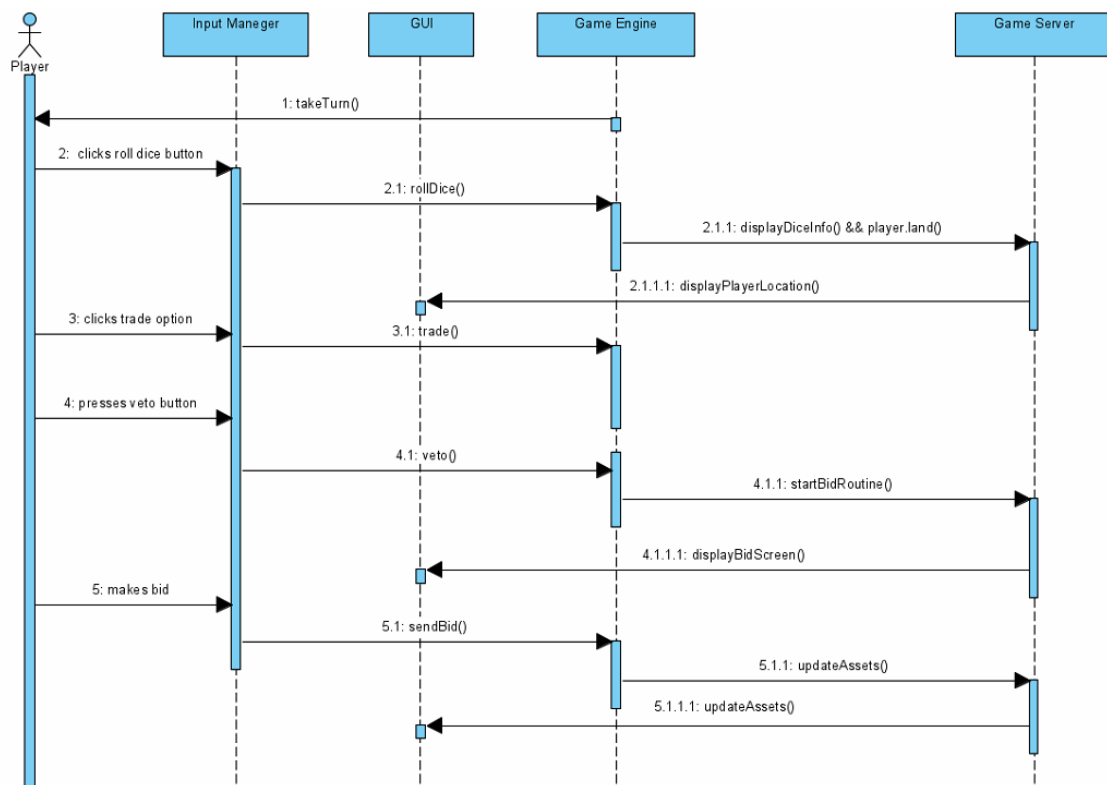· Player returns to the pause menu by clicking the close icon.

Main flow of events:

1. System opens the screen that displays game instructions and rules.

2. Player clicks the close icon and the use case ends.

5.2 Dynamic Models

5.2.1   Sequence Diagrams

5.2.1.1   Trading Scenario

In this scenario, the player is currently playing a game and it's her/his turn. Player rolls dice and the location which s/he lands is displayed on the screen. S/he does not want to trade or veto that asset, so that bidding starts. Game server starts a bid routine and the bid screen is displayed to the players. User makes a bid and the asset's owner is updated according to the higher bid.

5.2.1.2   Join Game Scenario
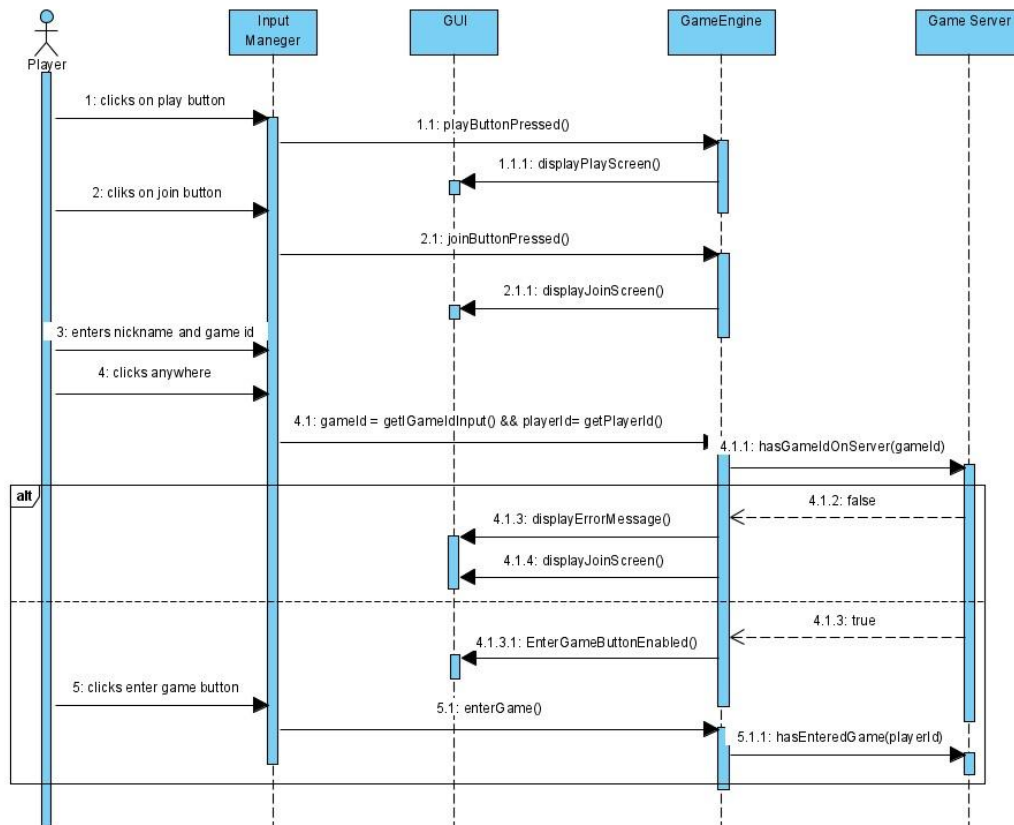
In this scenario, the player wants to join a game which is opened by another player. The user enters a nickname(player id) and a game id on the join screen. S/he must enter the correct game id which is chosen by the host player. When the player presses anywhere on the screen after entering data, the game id is searched on the server. If there is no game with that id, first an error message is displayed and an empty join screen opens. Otherwise, the "Enter Game" button is enabled and the player enters the game.

5.2.1.3   Change Settings Scenario
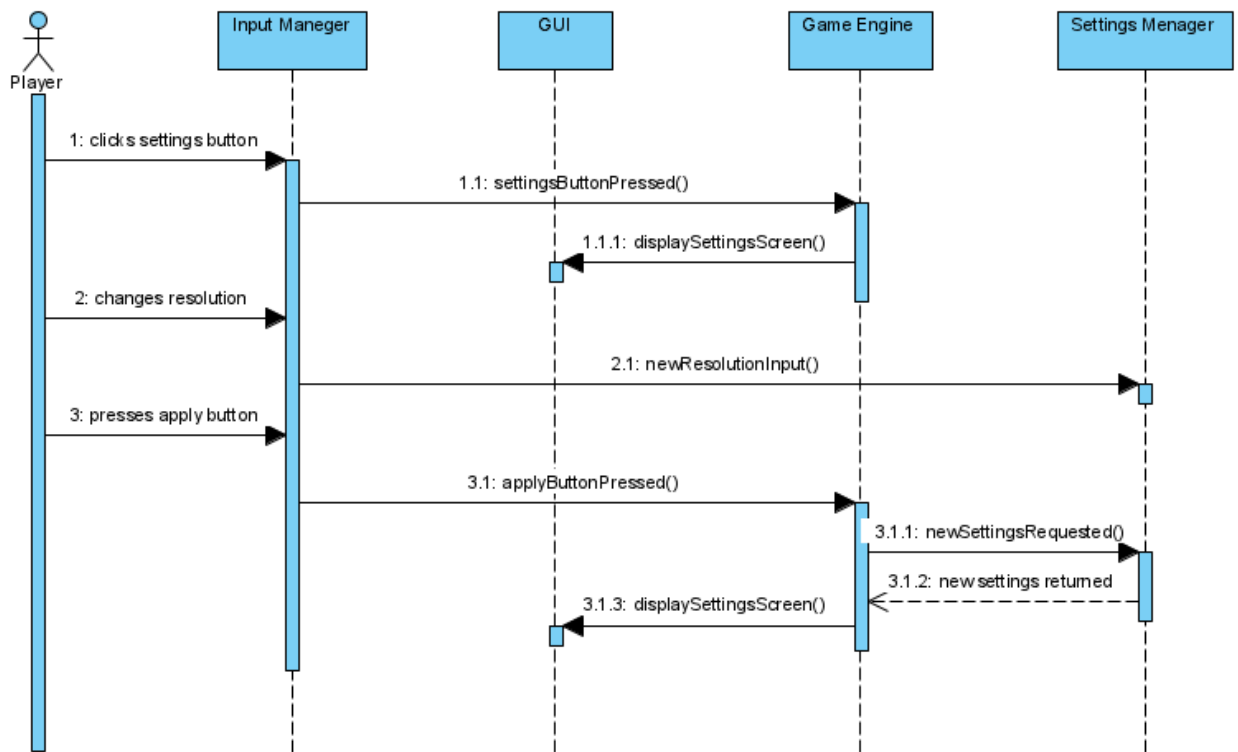
In this scenario, the player wants to change the resolution of the game window. First, the player clicks on the settings button. Game Engine detects it and executes display command. GUI gives the user the corresponding interface. Player changes the resolution, the settings manager catches the change. When the player clicks the apply button, Game Engine detects it and it requests new settings properties from the settings manager. Settings manager returns those properties and finally, Game Engine sends settings to GUI to display all again.

5.3 Object and Class Model

Player
- -player_name : string
- -playerId : int
- -isJailed : bool
- -totalPropCount : const int
- -totalUtilityCount : const int
- -totalRailroadCount : const int
- -totalCardCount : const int
- +totalAssetCount : const int
- +assetCount : int
- -isHost : bool
- -hasVeto : bool
- -pos : int
- +money : int
- -assets : Assets[]

- +Player(bool, int)
- +addAssets(Assets)
- +bid() : int
- +canBuild() : bool
- +canPlay() : bool
- +drawCard()
- +displayAssets()
- +getId() : int
- +goToJail()
- +info()
- +init()
- +land()
- +move(int[]) : int
- +removeAsset(int) : Assets
- +rollDice() : int[]
- +takeTurn() : int
- +toString() : string
- +trade()
- +veto() : bool

GameSquare
- +isOwned : bool
- +owner:Player
- +p:Property
- +u:Utility
- +r:Railroad
- +c:Card
- -aser:Assets[]

- +GameSquare(AssetType)
- +init(p:(AssetType)
- +getAssetType() : AssetType
- +setOwner(Player)
- +displayInfo()
- +getRent() : int
- +getPropOnSq() : Assets
- +getOwnerId() : int

Utility
- -cost : int
- -mortgageCost:int
- -income:int
- -utilityCount:int
- -utilityModifier:const int

- +Utility()

Card

- +Card()

Assets
- -player : Player
- +p_name : string
- +asset_type : AssetType

- +Assets()
- +setPlayer(Player)

Property
- -p_color : string
- -cost : int
- -totalDifferentRentTypes: const int
- -mortgageCost : int
- +income : int[]
- +houseCount : int
- -upgradeCost:int
- -countre:int

- +Property()
- +Property(String)
- +displayProperty()

Railroad
- -cost : int
- -mortgageCost : int
- -railroadModifier : const int
- +income : int
- -countcount : int

- +Railroad()
- +Railroad(string, int, parameter)
- +displayRailroad()

Game
- +activeTrade : bool
- +playerSize:const int
- +sqSize:const int
- +sqarr : GameSquare[]
- +parr : Player[]
- +tradeSender : Assets[]
- +tradeReciever : Assets[]
- +tradeSender_m : int
- +tradeReciever_m : int

- +Game()
- +initPlayers(Player[])
- +init()
- +startTradeRoutine(int)
- +startBidRoutine(int, parameter)
- +displayGameSquare(int)
- +displayPlayerInfo()

game_play_ui
- -components : IContainer = null
- -player1Pawn:PictureBox
- -player2Pawn:PictureBox
- -player3Pawn:PictureBox
- -player4Pawn:PictureBox
- -startGameButton:Guna2Button
- -diceButton:Guna2Button
- -pauseButton:Button
- -resumeButton:Button
- -settingsButton:Button
- -quitButton:Button
- -sellProperty:Button
- -bidButton:Button
- -bidTextBox:TextBox
- -diceAnimation:PictureBox
- -gameBoard:PictureBox
- -playerDisplay:PictureBox
- -chanceCards:PictureBox
- -communtyChestCards:PictureBox
- -pausePanel:Panel
- -bidPanel:Panel
- -buildHousePanel:Panel
- -paxTaxPanel:Panel
- -sellPropertyPanel:Panel

- #Dispose(bool)
- -InitializeComponent()
- -game_play_ui()
- -setPawn(object sender)
- -startButtonGameClick(object sender, EventArgs)
- -diceButtonClick(object sender, EventArgs)
- -pauseButtonClick(object sender, EventArgs)
- -resumeButtonClick(object sender, EventArgs)
- -quitButtonClick(object sender, EventArgs)
- -sellPropertyButtonClick(object sender, EventArgs)
- -bidButtonClick(object sender, EventArgs)
- -bidTextBoxChanged(object sender, EventArgs)
- -animate_dice()
- -animate_movement(int)
- -openBidPanel()
- -openBuildHousePanel()
- -openSellPropertyPanel()

join_game_ui
- -components : IContainer = null
- -playerIDTextBox:Guna2TextBox
- -GameIDTextBox:Guna2TextBox
- -enterButton:Guna2Button
- -joinLabel:Label
- -closeIcon:PictureBox
- -border1:PictureBox
- -border2:PictureBox

- #Dispose(bool)
- -InitializeComponent()
- -join_game_ui()
- +playerIDTextBox_TextChanged(object sender, EventArgs)
- -gameIDTextBox_TextChanged(object sender, EventArgs)
- -enterButtonClick(object sender, EventArgs)
- -closeIconClick(object sender, EventArgs)

host_game_ui
- -components : IContainer = null
- -playerIDTextBox:Guna2TextBox
- -gameIDTextBox:Guna2TextBox
- -enterButton:Guna2Button
- -hostLabel:Label
- -closeIcon:PictureBox
- -border1:PictureBox
- -border2:PictureBox

- #Dispose(bool)
- -InitializeComponent()
- -host_game_ui()
- -playerIDTextBox_TextChanged(object sender, EventArgs)
- -gameIDTextBox_TextChange(object sender, EventArgs)
- -enterButtonClick(object sender, EventArgs)
- -closeIconClick(object sender, EventArgs)

main_screen_ui
- -components : IContainer = null
- -playButton:Guna2Button
- -joinButton:Guna2Button
- -hostButton:Guna2Button
- -backButton:Guna2Button
- -settingsButton:Guna2Button
- -howToPlayButton:Guna2Button
- -creditsButton:Guna2Button
- -exitGameButton:Guna2Button

- #Dispose(bool)
- -InitializeComponent()
- +main_screen_ui()
- -playButtonClick(object sender, EventArgs)
- -joinButtonClick(object sender, EventArgs)
- -hostButtonClick(object sender, EventArgs)
- -backButtonClick(object sender, EventArgs)
- -settingsButtonClick(object sender, EventArgs)
- -howToPlayButtonClick(object sender, EventArgs)
- -creditsButtonClick(object sender, EventArgs)
- -exitGameButtonClick(object sender, EventArgs)

settings_ui
- -components : IContainer = null
- -settingsLabel:Label
- -resolutionLabel:Label
- -masterVolumeLabel:Label
- -bgmVolumeLabel:Label
- -sefVolumeLabel:Label
- -resolutionComboBox:Guna2ComboBox
- -masterVolumeTrackBar:GunaTrackBar
- -bgmVolumeTrackBar:GunaTrackBar
- -sefVolumeTrackBar:GunaTrackBar
- -applyButton:Guna2Button
- -closeIcon:PictureBox
- -border1:PictureBox
- -border2:PictureBox

- #Dispose(bool)
- -InitializeComponent()
- -settings_ui()
- -resolutionComboBoxSelectedTextChanged(object sender, ...
- -masterVolumeTrackBarScroll(object sender, EventArgs)
- -bgmVolumeTrackBarScroll(object sender, EventArgs)
- -sefVolumeTrackBarScroll(object sender, EventArgs)
- -applyButtonClick(object sender, EventArgs)
- -closeIconClick(object sender, EventArgs)

how_to_play_ui
- -components : IContainer = null
- -howToPlayLabel:Label
- -rulesTextBox:TextBox
- -closeIcon:PictureBox
- -border1:PictureBox
- -border2:PictureBox

- #Dispose(bool)
- -InitializeComponent()
- -how_to_play_ui()
- -closeIconClick()

credits_ui
- -components : IContainer = null
- -headLabel:Label
- -nameLabel1:Label
- -nameLabel2:Label
- -nameLabel3:Label
- -nameLabel4:Label
- -nameLabel5:Label
- -mainMenuButton:Guna2Button
- -border1:PictureBox
- -border2:PictureBox

- #Dispose(bool)
- -InitializeComponent()
- -mainMenuClick (object sender, EventArgs)

GameSquare holds its owner

GameSquare contains an asset

Player has a wallet of assets

Game has players

Game initializes and handles the state of the Game Squares

They communicate through database

Initializes game play screen

Initializes game play screen

Initializes join game screen

Initializes host game screen

Initializes settings screen

Initializes credits screen

Initializes how to play screen

21

The class diagram of our game "Monopoly Bidding War" currently has 15 classes. The game starts at the opening screen where players will have options about what they want to do, these visual aspects and their effects on the game will be mainly managed by the GUI classes, the main driver of the game is actually the class "Game". This class takes a lot of information from the other object classes such as "Assets", "GameSquare" and "Player".

Game Class

This class is responsible for all the main components of the game, even though each object is generally responsible for their own actions, properties or changes, this class handles concepts or actions regarding more than one player or object. This class initializes the game, cards, players and assets, handles trading, bidding, buying or selling properties etc.. This class is also responsible for the games communication with both the database and gui. Most of the calculations will be done on the hosts computer, will be sent to the database via this class, and then also will be received by this class on other players computers and the regarding received information will be distributed to the object classes accordingly.

GameSquare Class

This class represents an individual game square on the board, the squares can be a property, utility, railroad, start square or a card square. An array located at the game class will hold 40 instances of this class. This class will be responsible for being the middleman between the GUI, game class , player class and other objects, since this is the one key aspect of the game where all of the pieces come together. Displaying a square will be done through this class. For example if a player wants to buy a property or wants to collect rent from another player, those actions will be done through this class.

Assets Class and It's Children

These classes are here to represent every interactable / tradable object of the game, the assets class is the parent, and other classes that will be listed shortly are its children. They will inherit its methods, but some will be not inherited but individually declared in most of them, since all of these classes are very similar but sometimes act differently, which could be dangerous and buggy if not executed carefully. The children of the assets class are; property, utility, railroad and card. All of these classes have names, and except from the card

class, they all have rents, costs, mortgage values. The property class can have houses and hotels built on them, so even though they are all very similar and all are technically assets, some key distinctions are present between them. Thus, inherited or individually declared methods should be planned and written with care.

UI Classes

main_screen_ui Class

This is the first screen the player is prompted with, here the player can host or join a game, it can also view credits, and adjust the settings. The player can also go back to the original start screen from the play game and settings screens via a back button. If the player clicks on play a game, he will be prompted with three options, join a game, host a game, or go back. If s/he presses back, s/he will be taken back to the original starting screen, otherwise s/he will either go to the host_game_ui or join_game_ui class.

join_game_ui Class

This is a page containing two text boxes and an enter button, the player can enter his/her id, and the game id he/she wants to join.

host_game_ui Class

This page is identical to the join_game_ui page, only difference being instead of entering an id for the game, the player will set the id for others to join.

settings_ui Class

Here the player can adjust the sound level of the overall voice of the game, the sound effects voice level, or the music of the game. It can also change the resolution of the game.

how_to_play_ui Class

This is a simple page with a text field containing instructions, rules and tips & tricks for the game. This page is for new players, it will explain the basic concepts of monopoly,

also the rules relating to our version of the game. Since this is a new concept, new players can have questions relating to the rules, and they can view this window to answer those questions.

credits_ui Class

This class also contains a single text field which has credits written in it.

game_play_ui Class

This is the main GUI class, this is what the players will see most of the time, this is where the whole game takes place. All of the information sent to the database from the Game class will be displayed here. This class will contain the map, the game squares, the icons for the players, their money & assets, the chance cards, community chest cards, money collected for taxes, dice animations, bidding screens and so on. The player can open the settings window from here, it can concede, or quit the game.

5.4 User Interface - Navigational Paths and Screen Mock-ups
5.4.a   Main Menu Screen

There are five buttons for the player to select what to do.

Figure 1:

If the player presses the "play" button "settings", "how to play", "credits", "exit game" buttons disappear, and the "play" button becomes disabled. Also "join", "host", "back" buttons appear.

Figure 2:

If the player presses the "exit game" button a popup screen comes up to verify.

5.4.b   Settings Screen

In the settings screen there are four different options to adjust which are resolution, master volume, BGM volume, and SE volume. Also, a button to apply changes and a close button.
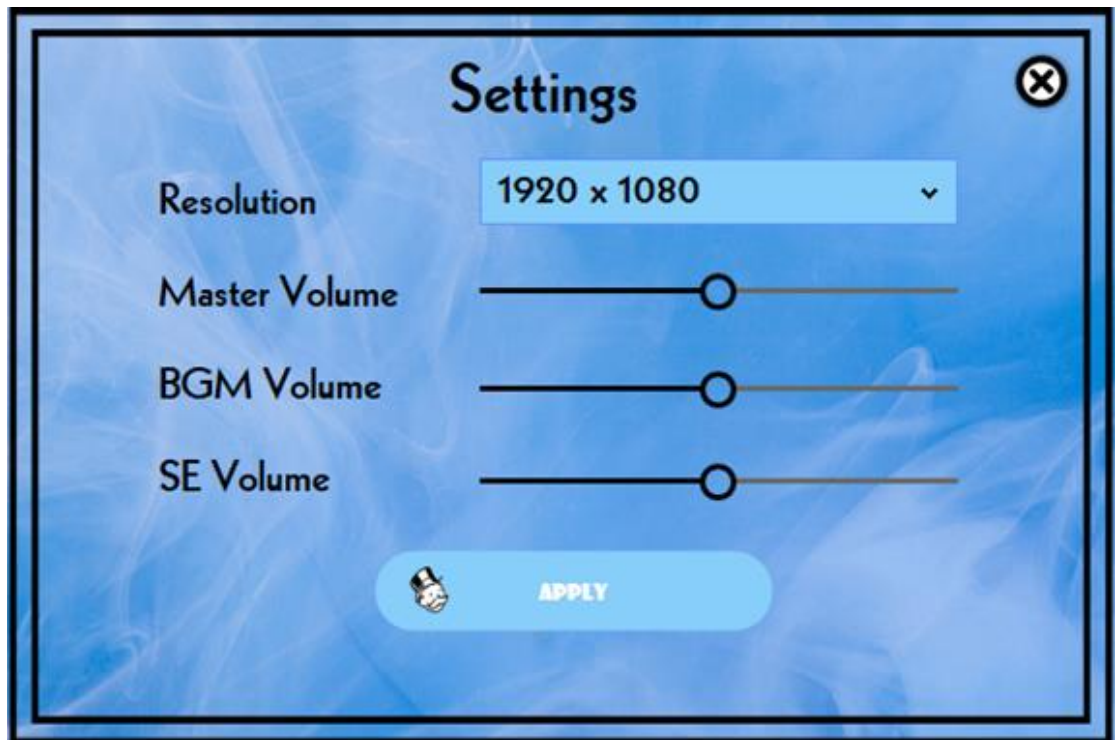
Figure 3:

5.4.c   How to Play Screen

How to play screen contains game rules.
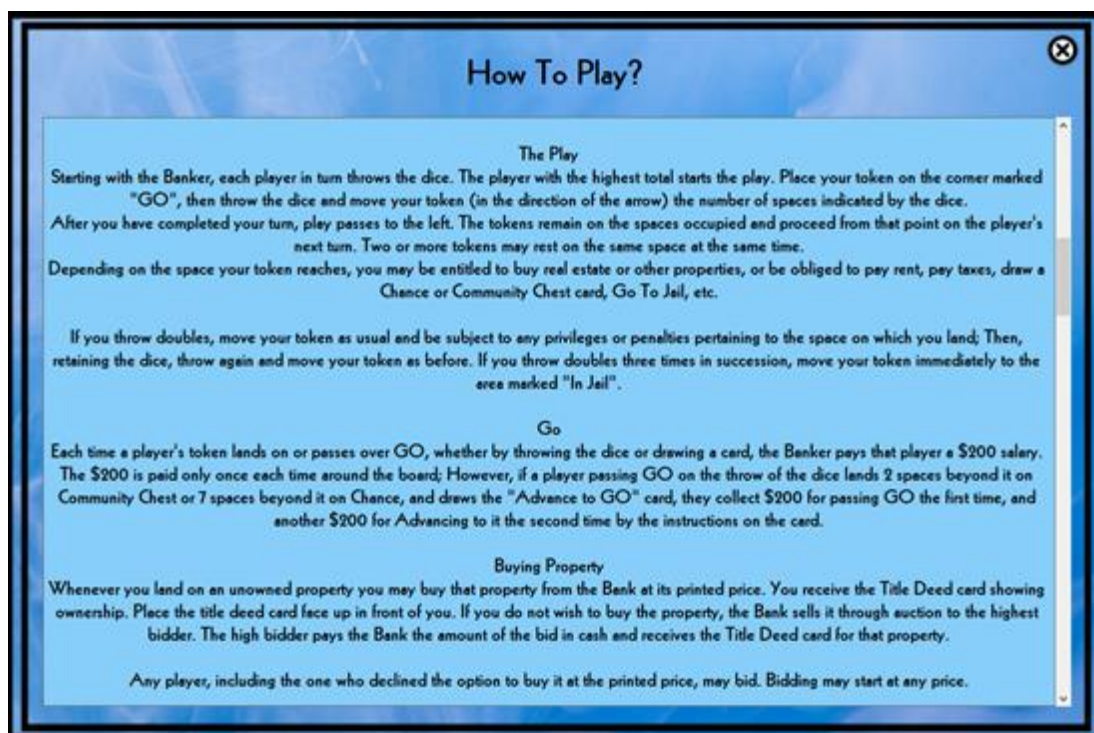
Figure 4:

5.4.d   Credits Screen

Figure 5:

5.4.e   Game Board

5.4.f   Places

5.4.i   Properties
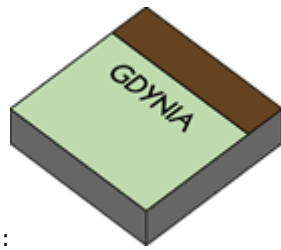
Gdynia:
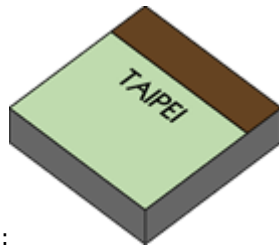
Taipei:

Tokyo:
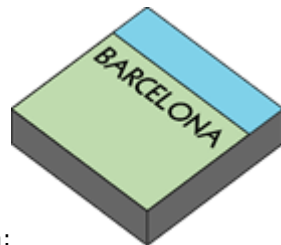
Barcelona:
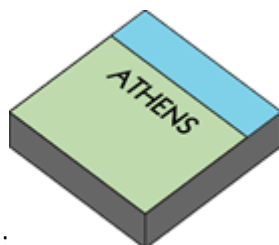
Athens:
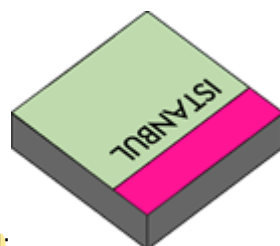
Istanbul:
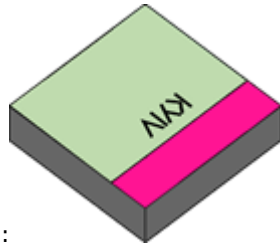
Kyiv:
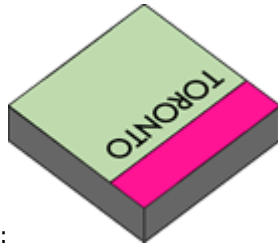
Toronto:
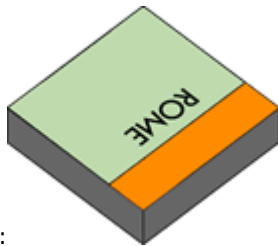
Rome:

Shanghai:
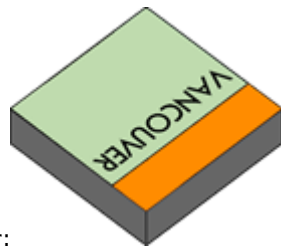
Vancouver:
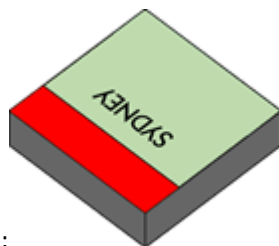
Sydney:
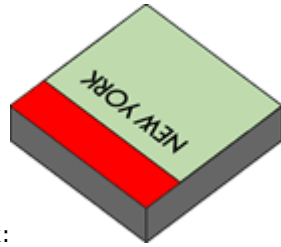
New York:
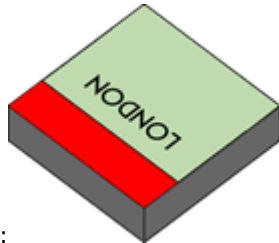
London:
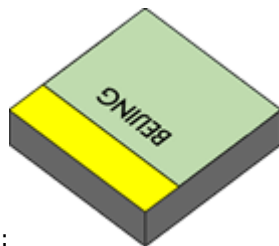
Beijing:
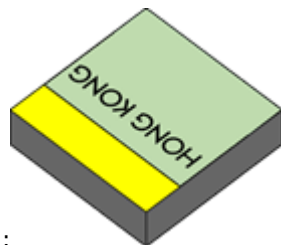
Hong Kong:
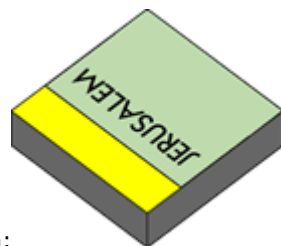
Jerusalem:
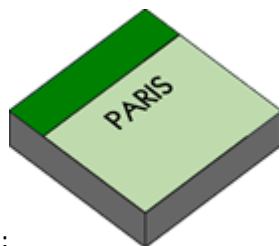
Paris:

Belgrade:

Cape Town:

Riga:

Montreal:

5.4.ii   Utilities

Grand Central Station:
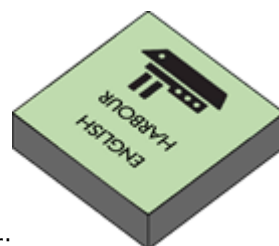
English Harbour:
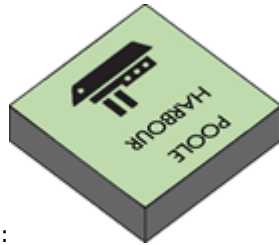
Poole Harbour:

Heathrow Airport:
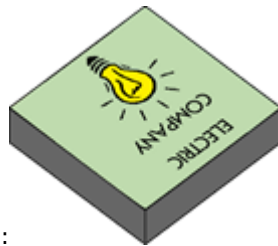
Electric Company:
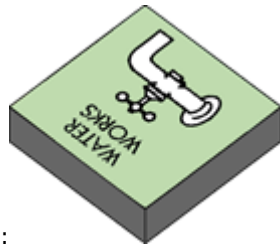
Water Works:

5.4.iii Special Places
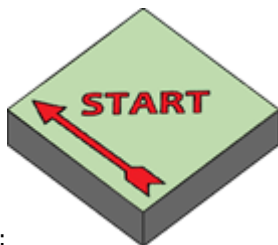
Start:

Income Tax:
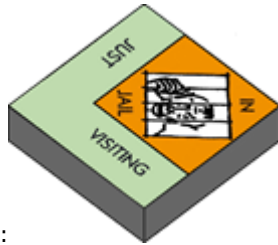
Visitor:
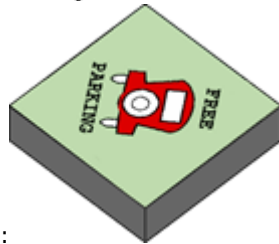
Free Parking:
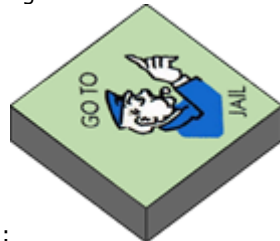
Go to Jail:

Luxury Tax:

Lower Left Chance:
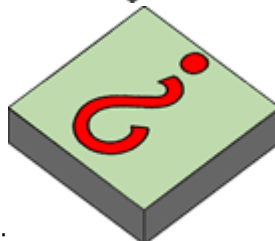
Upper Right chance:

Lower Right Chance:
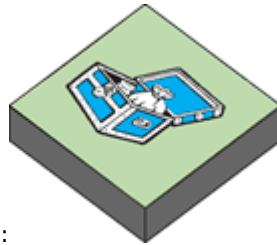
Lower Left Community Chest:
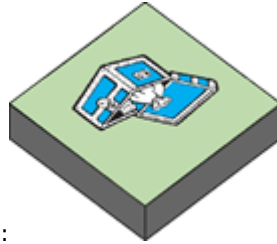
Upper Left Community Chest:
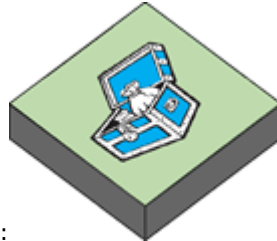
Lower Right Community Chest:

5.4.g   Property Cards

GDYNIA	
PRICE $ 60 RENT $ 2	
With 1 house	$ 10
With 2 house	$ 30
With 3 house	$ 90
With 4 house	$ 160
With HOTEL	$ 250
One house costs $ 50	
Mortgage value $ 30	

TAIPEI	
PRICE $ 60 RENT $ 4	
With 1 house	$ 20
With 2 house	$ 60
With 3 house	$ 180
With 4 house	$ 320
With HOTEL	$ 450
One house costs $ 50	
Mortgage value $ 30	

TOKYO	
PRICE $ 100 RENT $ 6	
With 1 house	$ 30
With 2 house	$ 90
With 3 house	$ 270
With 4 house	$ 400
With HOTEL	$ 550
One house costs $ 50	
Mortgage value $ 50	

BARCELONA	
PRICE $ 100 RENT $ 6	
With 1 house	$ 30
With 2 house	$ 90
With 3 house	$ 270
With 4 house	$ 400
With HOTEL	$ 550
One house costs $ 50	
Mortgage value $ 50	

ATHENS	
PRICE $ 120 RENT $ 8	
With 1 house	$ 40
With 2 house	$ 100
With 3 house	$ 300
With 4 house	$ 450
With HOTEL	$ 600
One house costs $ 50	
Mortgage value $ 60	

ISTANBUL	
PRICE $ 140 RENT $ 10	
With 1 house	$ 50
With 2 house	$ 150
With 3 house	$ 450
With 4 house	$ 625
With HOTEL	$ 750
One house costs $ 100	
Mortgage value $ 70	

KYIV	
PRICE $ 140 RENT $ 10	
With 1 house	$ 50
With 2 house	$ 150
With 3 house	$ 450
With 4 house	$ 625
With HOTEL	$ 750
One house costs $ 100	
Mortgage value $ 70	

TORONTO	
PRICE $ 160 RENT $ 12	
With 1 house	$ 60
With 2 house	$ 180
With 3 house	$ 500
With 4 house	$ 700
With HOTEL	$ 900
One house costs $ 100	
Mortgage value $ 80	

ROME	
PRICE $ 180 RENT $ 14	
With 1 house	$ 70
With 2 house	$ 200
With 3 house	$ 550
With 4 house	$ 700
With HOTEL	$ 900
One house costs $ 100	
Mortgage value $ 90	

SHANGHAI	
PRICE $ 180 RENT $ 14	
With 1 house	$ 70
With 2 house	$ 200
With 3 house	$ 550
With 4 house	$ 700
With HOTEL	$ 950
One house costs $ 100	
Mortgage value $ 90	

VANCOUVER	
PRICE $ 200 RENT $ 16	
With 1 house	$ 80
With 2 house	$ 220
With 3 house	$ 600
With 4 house	$ 800
With HOTEL	$ 1000
One house costs $ 100	
Mortgage value $ 100	

SYDNEY	
PRICE $ 220 RENT $ 18	
With 1 house	$ 90
With 2 house	$ 250
With 3 house	$ 700
With 4 house	$ 875
With HOTEL	$ 1050
One house costs $ 150	
Mortgage value $ 110	

NEW YORK	
PRICE $ 220 RENT $ 18	
With 1 house	$ 90
With 2 house	$ 250
With 3 house	$ 750
With 4 house	$ 875
With HOTEL	$ 1050
One house costs $ 150	
Mortgage value $ 120	

LONDON	
PRICE $ 240 RENT $ 20	
With 1 house	$ 100
With 2 house	$ 300
With 3 house	$ 750
With 4 house	$ 925
With HOTEL	$ 1100
One house costs $ 150	
Mortgage value $ 120	

BEIJING	
PRICE $ 260 RENT $ 22	
With 1 house	$ 110
With 2 house	$ 330
With 3 house	$ 800
With 4 house	$ 975
With HOTEL	$ 1150
One house costs $ 150	
Mortgage value $ 130	

HONG KONG	
PRICE $ 260 RENT $ 22	
With 1 house	$ 110
With 2 house	$ 330
With 3 house	$ 800
With 4 house	$ 975
With HOTEL	$ 1150
One house costs $ 150	
Mortgage value $ 130	

JERUSALEM	
PRICE $ 280 RENT $ 24	
With 1 house	$ 120
With 2 house	$ 360
With 3 house	$ 850
With 4 house	$ 1025
With HOTEL	$ 1200
One house costs $ 150	
Mortgage value $ 140	

PARIS	
PRICE $ 300 RENT $ 26	
With 1 house	$ 130
With 2 house	$ 390
With 3 house	$ 900
With 4 house	$ 1100
With HOTEL	$ 1275
One house costs $ 200	
Mortgage value $ 150	

BELGRADE	
PRICE $ 300 RENT $ 26	
With 1 house	$ 130
With 2 house	$ 390
With 3 house	$ 900
With 4 house	$ 1100
With HOTEL	$ 1275
One house costs $ 200	
Mortgage value $ 150	

CAPE TOWN	
PRICE $ 320 RENT $ 28	
With 1 house	$ 150
With 2 house	$ 450
With 3 house	$ 1000
With 4 house	$ 1200
With HOTEL	$ 1400
One house costs $ 200	
Mortgage value $ 160	

RIGA	
PRICE $ 350 RENT $ 35	
With 1 house	$ 175
With 2 house	$ 500
With 3 house	$ 1100
With 4 house	$ 1300
With HOTEL	$ 1500
One house costs $ 200	
Mortgage value $ 175	

MONTREAL	
PRICE $ 400 RENT $ 50	
With 1 house	$ 200
With 2 house	$ 600
With 3 house	$ 1400
With 4 house	$ 1700
With HOTEL	$ 2000
One house costs $ 200	
Mortgage value $ 200	

5.4.h   Utility Cards

GRAND CENTRAL STATION	ENGLISH HARBOUR	POOLE HARBOUR
PRICE $ 200	PRICE $ 200	PRICE $ 200
If 1 owned $ 25	If 1 owned $ 25	If 1 owned $ 25
If 2 owned $ 50	If 2 owned $ 50	If 2 owned $ 50
If 3 owned $ 100	If 3 owned $ 100	If 3 owned $ 100
If 4 owned $ 200	If 4 owned $ 200	If 4 owned $ 200
Mortgage value $ 100	Mortgage value $ 100	Mortgage value $ 100

HEATHROW AIRPORT	ELECTRIC COMPANY	WATER WORKS
PRICE $ 200	PRICE $ 150	PRICE $ 150
If 1 owned $ 25	If 1 owned, rent equals	If 1 owned, rent equals
If 2 owned $ 50	4 times dice roll	4 times dice roll
If 3 owned $ 100	If 2 owned, rent equals	If 2 owned, rent equals
If 4 owned $ 200	10 times dice roll	10 times dice roll
Mortgage value $ 100	Mortgage value $ 75	Mortgage value $ 75

5.4.i   Chance Cards

CHANCE	CHANCE
ADVANCE TO "GO" (COLLECT $ 200)	ADVANCE TO ISTANBUL IF YOU PASS "GO" COLLECT $ 200
CHANCE	CHANCE
ADVANCE TO LONDON IF YOU PASS "GO" COLLECT $ 200	ADVANCE TO MONTREAL

CHANCE

ADVANCE TO
NEAREST STATION.
IF UNOWNED, you may
buy it from Bank.
IF OWNED, pay owner
twice the rental to which
they are otherwise entitled.

CHANCE

BANKS PAY YOU
DIVIDEND OF $ 50

CHANCE

YOU HAVE WON A
CROSSWORD COMPETITION
COLLECT $ 100

CHANCE

YOUR BUILDING
LOAN MATURES
COLLECT $ 150

CHANCE

YOU HAVE BEEN ELECTED
CHAIRMAN OF THE BOARD
PAY EACH PLAYER $ 50

CHANCE

MAKE GENERAL REPAIRS
ON ALL YOUR PROPERTY
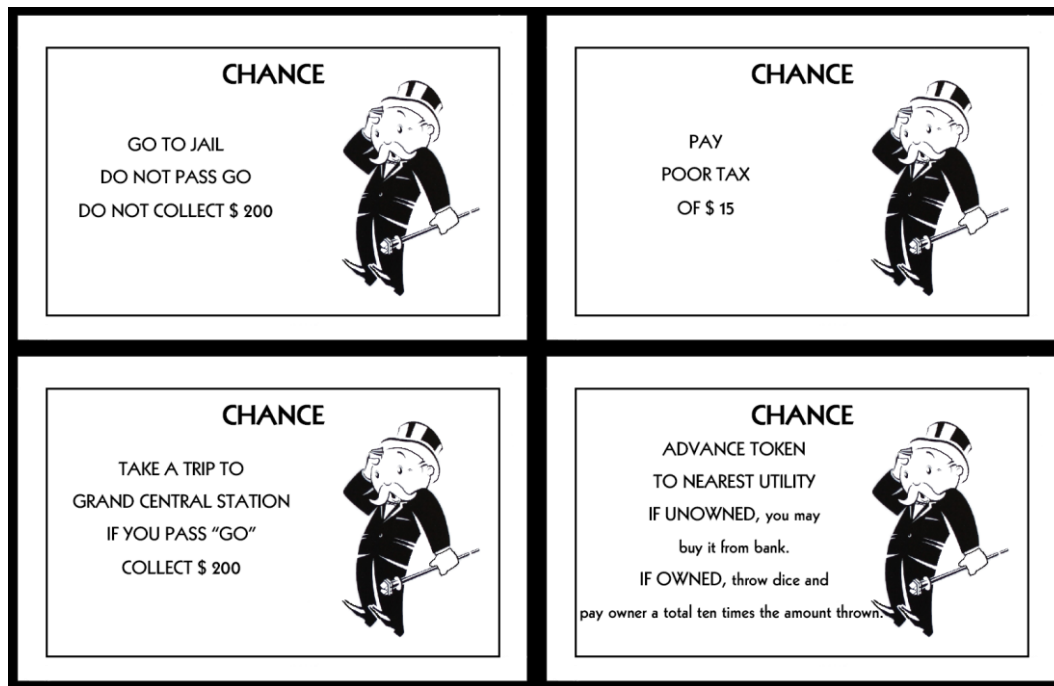FOR EACH HOUSE PAY $ 25
FOR EACH HOTEL PAY $ 100

CHANCE

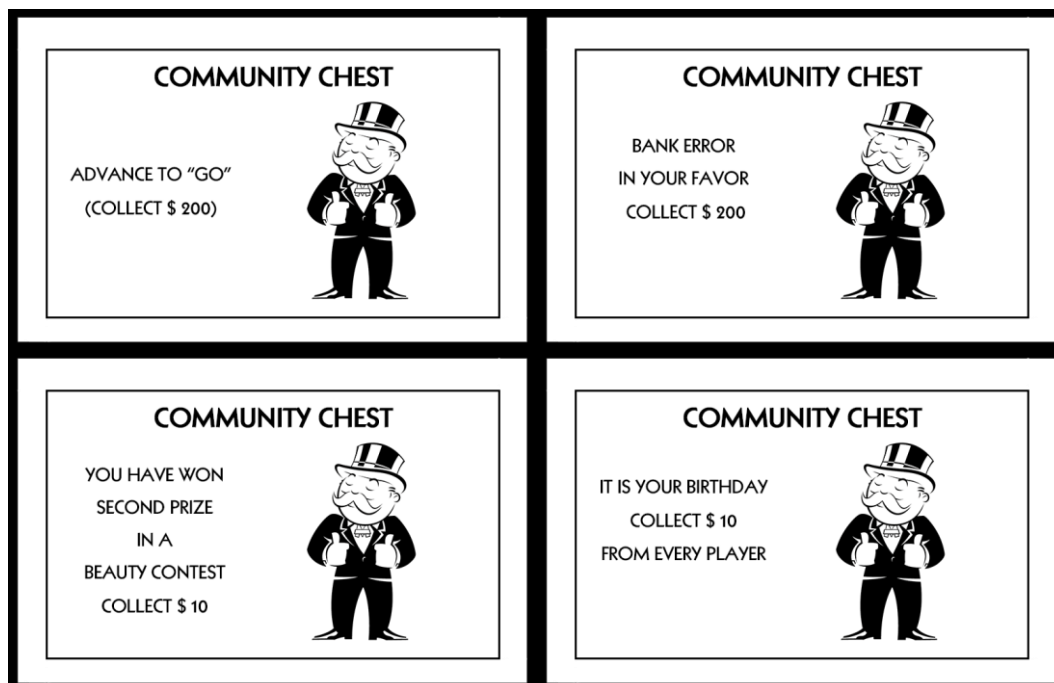GET OUT OF
JAIL FREE
THIS CARD MAY BE KEPT
UNTIL NEEDED, OR SOLD

CHANCE

GO BACK
THREE SPACES

CHANCE

GO TO JAIL
DO NOT PASS GO
DO NOT COLLECT $ 200

CHANCE

PAY
POOR TAX
OF $ 15

CHANCE

TAKE A TRIP TO
GRAND CENTRAL STATION
IF YOU PASS "GO"
COLLECT $ 200

CHANCE

ADVANCE TOKEN
TO NEAREST UTILITY
IF UNOWNED, you may
buy it from bank.
IF OWNED, throw dice and
pay owner a total ten times the amount thrown.

5.4.j   Community Chest Cards

COMMUNITY CHEST

ADVANCE TO "GO"
(COLLECT $ 200)

COMMUNITY CHEST

BANK ERROR
IN YOUR FAVOR
COLLECT $ 200

COMMUNITY CHEST

YOU HAVE WON
SECOND PRIZE
IN A
BEAUTY CONTEST
COLLECT $ 10

COMMUNITY CHEST

IT IS YOUR BIRTHDAY
COLLECT $ 10
FROM EVERY PLAYER

COMMUNITY CHEST

DOCTOR'S FEE

PAY $ 50

COMMUNITY CHEST

GO TO JAIL

DO NOT PASS GO

DO NOT COLLECT $ 200

COMMUNITY CHEST

GET OUT JAIL, FREE

THIS CARD MAY BE KEPT

UNTIL NEEDED,

OR SOLD

COMMUNITY CHEST

HOSPITAL'S FEE

PAY $ 50

COMMUNITY CHEST

INCOME TAX

REFUND

COLLECT $ 20

COMMUNITY CHEST

YOU INHERIT

$ 100

COMMUNITY CHEST

LIFE INSURANCE

MATURES

COLLECT $ 100

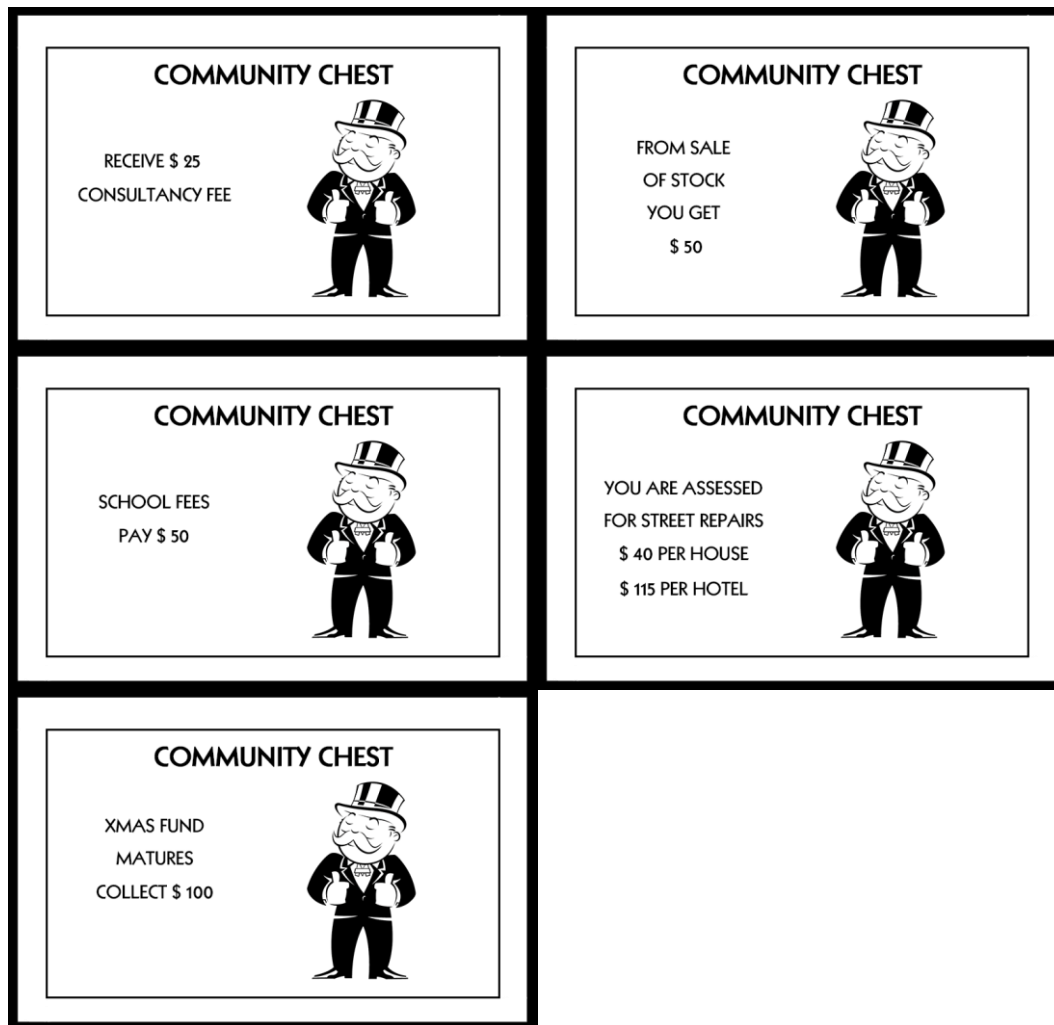**COMMUNITY CHEST**

GRAND OPERA OPENING

COLLECT $ 50 from

Every Player for

Opening Night

Seats

6 Glossary & References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 3rd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2010, ISBN-10: 0136066836.

[2] https://www.hasbro.com/common/instruct/monins.pdf

[3] Applying UML and Patterns - An Introduction to Object-Oriented Analysis and Design and Iterative Development, by Craig Larman, Prentice Hall, 2004, ISBN: 0-13-148906-2.