anOtherGame

Requirements and Analysis Document

Version 1.0
2017-05-22

Miranda Bånnsgård
Amanda Jonsson
Maja Nyberg
Allex Nordgren

This version overrides all previous versions.

1 Introduction

The project aims to create a prototype desktop application of the board game *The Lost Diamond* which is designed by the board game designer Kari Mannerla[1]. To revive this well-selling board game this project will create a Gothenburg edition of the *The Lost Diamond*. This will be done, for example by replacing move by flight to move by tram. It will be a standalone and multi-player application.

General characteristics of the application:

- The game is turn-based. When a player has completed his or her turn the turn proceeds to the next player. The order of priority is randomly generated by the application after inserting the names of the players and started a new game.
- The game can be canceled, or it will end according to the rules.
- There is no time constraint for a round.
- The application will handle the rules.
- The application does not include a computer-player. It is not possible to play the game alone.
- The application does not save interrupted games or collect any statistics.

1.1 Definitions, acronyms and abbreviations

GUI - Graphical User Interface
Java - platform independent language
JRE - Java Runtime Environment. Additional software needed to run a Java application.
Host - a computer where the game will run.

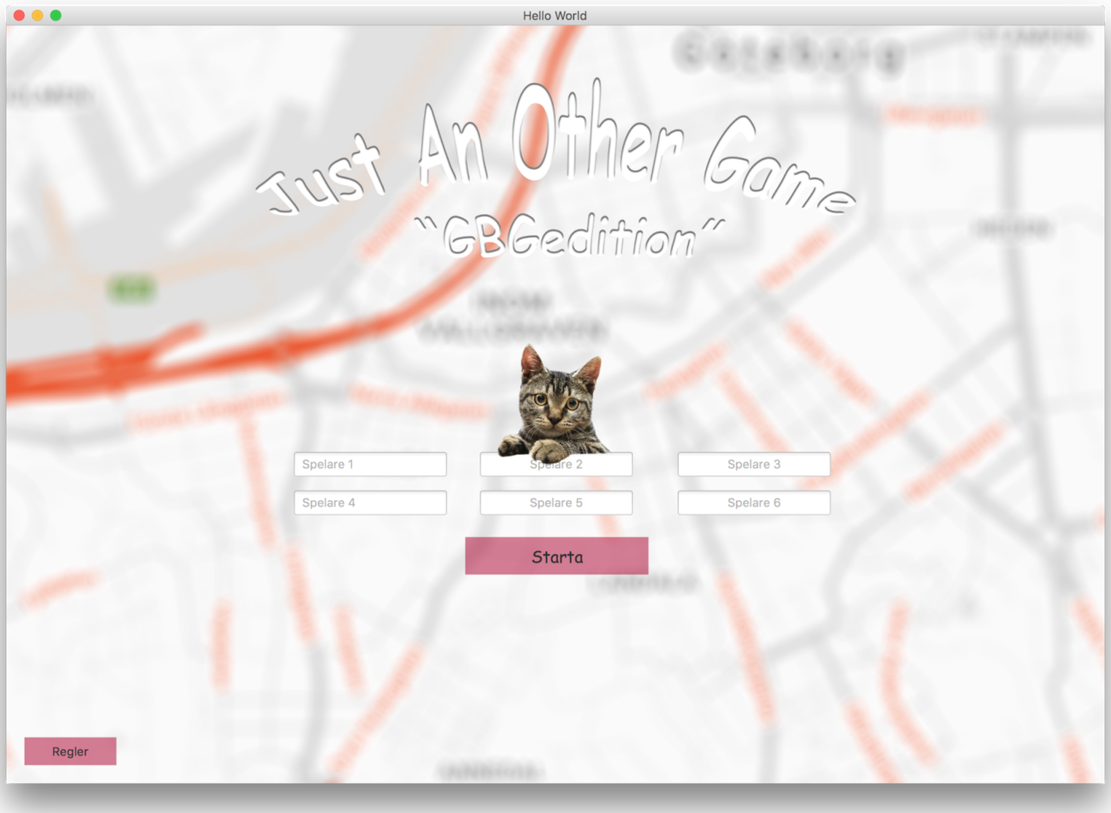
Station - a station is a boat station or a tram station.
Marker - a marker is located on stations. The player can get money, get robbed or find either a cat or tram-card when flipping a marker.
Round - one complete game ending in a winner or possible canceled.
Turn - the turn of each player. The player can only act during his or her turn.

[1] Afrikan tähti, Wikipedia, https://en.wikipedia.org/wiki/Afrikan_tähti

2 Requirements

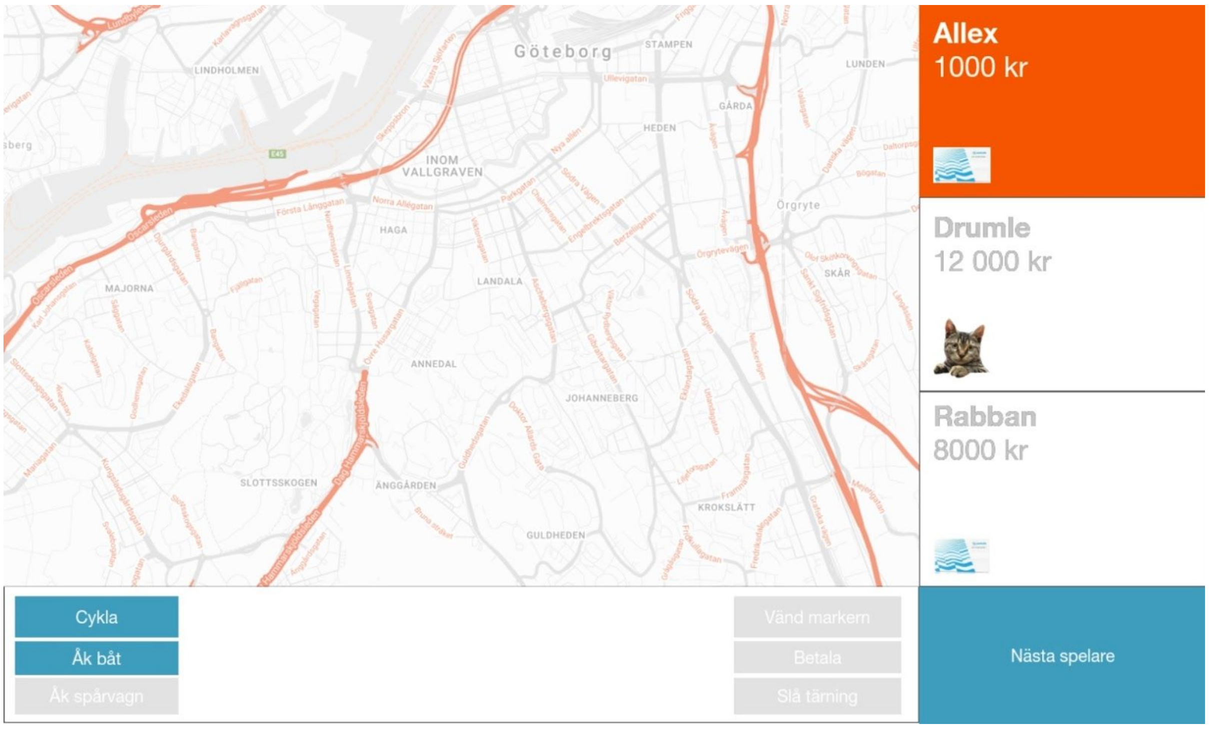
2.1 User interface

When the game is started, the first view that appears is the view below. It has text fields for the players to type in their names as well as two button - one for reading the rules and one for starting the game.



The start view.

The game view will look similar to the ordinary game. It will have a map with places to go to, specified start location and routes between the places. All players will be visible at all time, together with their budget and if they have a tram card or the cat. The lower pane with buttons will show the player what action are available, as well as specifying whose turn it is.



The game view.

The whole GUI must be resizable.

2.2 Functional requirements

The player should be able to:

1. Select how many players should play the game (between 1 and 6).
2. Start a new game by pressing the “Starta spel” button.
3. Do a turn. During the turn the player should be able to:
    a. Roll dices. This should trigger a response from the application showing where the player can go on the map.
    b. Travel by bike, boat and tram.
    c. Flip markers.
    d. End the turn.
4. Read instructions for the game, and the rules, by pressing the “Regler” button.
5. See the game over screen when a round has finished
    a. From here either start a new game or exit the application.

    b. See which player won the game.
6. Exit the application.

Ordering of use cases by priority

1. Move
2. Player's turn
3. Flip marker
4. Do marker
5. Win game
6. Game over
7. New game
8. Start game
9. Move by bike
10. Move by tram
11. Move by boat

2.3 Non-functional requirements

Usability is high priority. The game should be user friendly and easy to understand. Normal users should be able to play the game within a very short period. The player is able to read the rules in the *Rules* screen.

The player should be able to recognize the map and street names from the city Gothenburg.

It should be possible to play a complete round without a crash.

Any actions initiated by the player should not exceed a 5 second response time in worst case. The application's response should take 5 seconds at most.

The game will not be implemented for other platforms than computer. The game should be supported fully by operative systems Windows, Mac and Linux.

The application uses images which we have not been given permission to use.

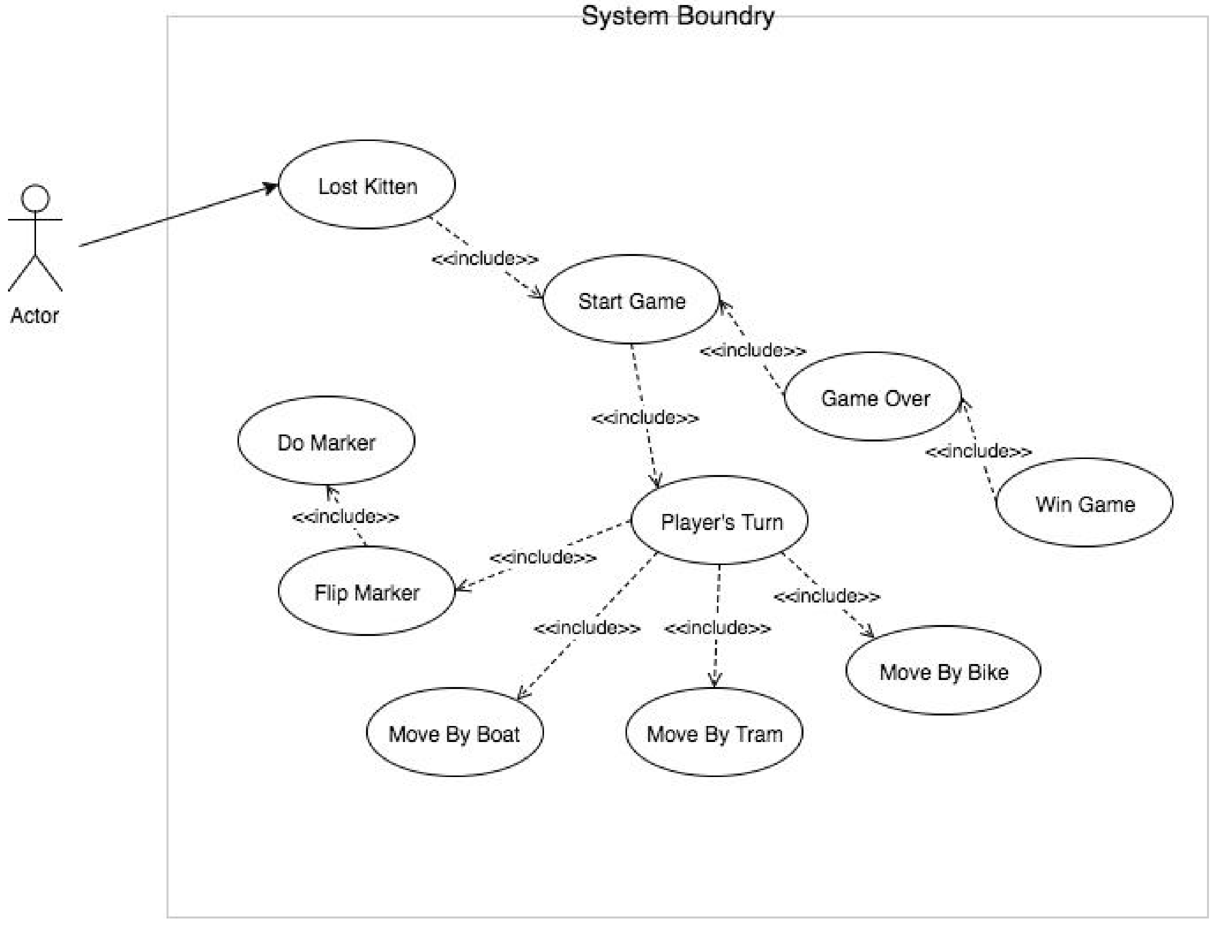
The program will be delivered as a zip-archive containing:

- All needed resources involved in the application.
- A README-file documenting the installation and start of application.

To achieve platform independence, the application will use Java environment. All hosts must have JRE installed and configured. The application needs to be installed on all hosts where it will run.

There should be automated tests for each class. The GUI should be tested manually

3 Use cases


System Boundry
Actor
Lost Kitten
<<include>>
Start Game
<<include>>
Game Over
<<include>>
Win Game
<<include>>
Do Marker
<<include>>
Player's Turn
<<include>>
Flip Marker
<<include>>
<<include>>
<<include>>
Move By Bike
Move By Boat
Move By Tram

3.1 Use case listing

1. Use Case: Lost Kitten

Summary: User opens application Lost Kitten and the start menu is displayed.
Priority: High
Extends: NA
Includes: Start Game
Participators: The player

Normal flow of events: Player opens the desktop application Lost Kitten.

	User	System
1	Chooses to open the application Lost Kitten.	
2		Opens application and displays the start menu.
3	See UC Start Game	

2. Use Case: Start Game

Summary: The start menu is displayed and the player fills out the name of the different players that will participate in a game of Lost Kitten. The player clicks on the “Starta” button on the start menu. A new window with the game board is shown.
Priority: High
Extends: Lost Kitten
Includes: Player’s Turn
Participators: The user

Normal flow of events: The user starts a game without reading the rules first.

	User	System
1	User writes the names of the players (between 1 and 6).	
2	Clicks on button “Starta”.	
3		Sets up a new window with the game board containing the map of Gothenburg and the number of players chosen. Places 30 markers randomly on the 30 stations. The chosen number of players are set up with 5000:- each. One of the players, randomly selected, will be activated and able to start to roll the dice.

Alternate flow: The user starts the game after reading the rules.

	User	System
1.1	Clicks on the “Regler” button.	
1.2		Shows the rules in a pop-up window.
1.3	Exits the pop-up with rules by pressing the “Exit Rules”-button.	
1.4		Closes the pop-up window.

3. Use Case: Player´s Turn

Summary: This case explains how the user moves by bike, which is the standard move. The cycle path is marked by black lines and black dots.
Priority: High
Extends: Start Game
Includes: Move by Bike, Move By Tram, Move By Boat, Flip Marker
Participants: The user

Normal flow of events: The turn before has just ended. The user moves by bike. This requires that the user´s jack is placed on land. It does not cost to move by bike. The player can be placed either on the cycle path or a station. There are some special spaces that are not stations but still have some kind of action.

	Actor	System
1	See UC *Move By Bike*	
2		Player is on the cycle path, nothing happens. Next player's turn
2.1 Player stays at a station with a marker	See UC Flip Marker	
2.2 Player stopped/landed on station Kapstaden		Update player's balance with 5000:-

Alternative flow: The player landed on a special space with action. Player must roll a 1 or a 2 to be able to move on.

	Actor	System
2.3.1	Player has landed on a special space. Needs to roll a 1 or a 2.	
2.3.1.1 If player roll 1 or 2		Player is enabled to roll dice in order to move the next turn
2.3.1.2 If player does not roll 1 or 2		Player has two wait to next turn and try to roll a 1 or 2 again.

Alternative flow: The player is on a station and can choose to move by tram.

	Actor	System
1.1.1	See UC Move By Tram	
1.1.2 If player is on a station with a marker	See UC Flip Marker	

Alternative flow: The player is on a station by the water with boat lines connected and can choose to move by boat.

	Actor	System
1.2.1	See UC Move By Boat	
1.2.2 If player is on a station with a marker	See UC Flip Marker	

Alternative flow: The player is already on a station with a marker (perhaps player did not succeed to flip marker previous turn).

	Actor	System
1.3.1	See UC Flip Marker	

4. Use Case: Move by bike

Summary: The standard way for a move is by bike, it is also free of charge. The cycle path is marked by black lines and black dots.
Priority: high
Extends: Player’s Turn
Includes:
Participators: The user

Normal flow of events: The user moves by bike. This requires that the user´s jack is placed on space on land. It does not cost to move by bike. The player can move as many step shown by the dice (except out on water), but the player can also choose to stop earlier if there is a station

	Actor	System
1	The user clicks on the dice	
2		Roll the dice (animation). Shows the different path the player can take, by lightning(?) up the positions the player would be placed at.
3	Picks a path by clicking on the position.	
4		Moves the jack to the chosen position.
4.1 Player chooses to stay at the station		Moves the jack to a station, which could be fewer steps away than the number of steps shown by the dice.

5. Use Case: Flip Marker

Summary: The player has landed/chosen to stop on a tram/boat station with a marker. The player can choose to flip the marker in the same move.
Priority: High
Extends: Player's Turn
Includes: Do Marker
Participants: Actual player

Normal flow of events: The player chooses to try to get the marker and pays 1000:- for it. Only works if player's balance is over 1000:-. If balance is below 1000:- the player can only roll dice to flip marker.

	User	System
1	Chooses to flip the marker on the stop	
2		Displays different buttons in the "Alternative-window" below.
3	Clicks the "Pay 1000"	
4		Updates the player's balance
5		"Flips" the marker, marker is displayed
6		See 'Do Marker'
7		Next player's turn

Alternative flow: The player can choose to try to get a 4, 5 or a 6 with the dice instead of paying 1000:-.

	User	System
3.1	Clicks on the dice	
3.2		Displays the number on the dice.
3.2.1 If dice shows 4, 5 or a 6		See 'Do Marker'
3.2.2		Next player's turn.
3.2.2 If the dice displays 1, 2 or 3		Next player's turn

Alternative flow: The player lands on a marker but chooses not to pay or roll the dice in order to get the marker.

	User	System
1.1	Chooses not to turn marker	
1.2	Clicks on “End Turn”	
1.3		Next player’s turn.

6. Use Case: Do Marker

Summary: The player has flipped a marker and the system performs the action of the marker. There are seven different kind of markers with different actions.
Priority: High
Extends: Flip Marker
Includes:
Participants: System only

Normal flow of events: The player turns a blank marker.

	User	System
1		Displays the blank marker.
2		Nothing happens.
2.1 The marker were on the station Nordstan		Player will have to skip a turn.

Alternative flow: The player turns a marker with a gem worth 3000:-

	User	System
1.1.1		Displays a marker with a yellow gem
1.1.2		Updates balance of the player with 3000:-.
1.1.2.1 The marker were on the stop Guldheden		Updates balance of the player with 6000 :-.

Alternative flow: The player turns a marker with a gem worth 4000:-

	User	System
1.2.1		Displays a marker with a green gem
1.2.2		Updates balance of the player with 4000:-.
1.2.2.1 The marker were on the stop Guldheden		Updates balance of the player with 8000 :-.

Alternative flow: The player turns a marker with a gem worth 5000:-

	User	System

1.3.1		Displays a marker with a red gem
1.3.2		Updates balance of the player with 5000:-.
1.3.2.1 The marker were on the stop Guldheden		Updates balance of the player with 10.000 :-.

Alternative flow: The player turns a marker with a bandit/pickpocket.

	User	System
1.4.1		Displays a marker with a pickpocket
1.4.2		Updates the balance to 0.

Alternative flow: The player turns a marker with a tram ticket.

	User	System
1.5.1		Displays a marker with a tram ticket
1.5.2		Displays this ticket in the player’s “backpack”.

Alternative flow: The player turns a marker with the kitten!

	User	System
1.6.1		Displays a marker with a cat
1.6.2		Puts the cat in the backpack.

7. Use Case: Move by tram

Summary: This case explains how the user moves by tram, which costs 3000 kr. The tram path is marked by lines between bigger tram stations. To move by tram your jack needs to be placed at a tram station.
Priority: high
Extends: Player's Turn
Includes:
Participants: The user

Normal flow of events: The user pays 3000 kr to move by tram.

	User	System
1	The user pick the alternative to move by tram in the alternative box	
2		Shows a new alternative box with two alternatives: pay 3000 kr to move by tram or gate-crashing.
3	Clicks "Pay 3000"	
4		Calculates and updates player's balance.
5		Shows the different tram paths the user can take by lightening them.
6	Picks one of the paths by clicking on it.	
7		Moves jack to the chosen station.

Alternative flow: The player chooses to gate-crash and gets caught doing it and chooses to pay for it.

	User	System
3.1	Clicks "Skip payment"	
3.2		Shows the different tram paths the user can take by lightening them.
3.3		Shows a random generated number between 1-100 which represent how big the odds are to get caught
3.4	Picks one of the paths by clicking on it	
3.5	The player gets caught	

3.6		Shows a picture of a tram controller
3.7		Shows two alternatives in the alternative box: pay 5000 kr to get to the next stop or do not want to pay and goes back to the departure station.
3.8	Clicks on “Pay Fee”	
3.9		Calculates and updates player’s balance.
3.10		Moves the jack to the chosen station.

Alternative flow: The player chooses to gate-crash and gets caught doing in and chooses to not pay for it. Player then has to return to the departure station.

	User	System
3.7.1	Clicks on “Return to station”	
3.7.2		Moves jack to the departure station again.

Alternative flow: The player chooses to gate-crash and do not get caught.

	User	System
3.4.1		Moves jack to the chosen station

8. Use Case: Move by boat

Summary: This case explains how the user moves by boat. The boat path is marked by blue lines and blue dots.
Priority: Medium
Extends: Player’s Turn
Includes:
Participators: The user

Normal flow of events: The user moves by boat. This requires that the user´s jack is placed on a stop by water with a path crossing the water. Move by boat costs 1000:-.

	Actor	System
1	The user chooses to go by boat by clicking somewhere on the “alternative panel”.	
2		User pays 1000 kr. Calculate how much money the user has left. Update the user’s budget.
3	The user clicks on the dice.	
4		Roll the dice (animation). Show the different paths the player can take (only boat paths), by lighting up the positions the player could be placed at.
5	Picks a path by clicking on the position.	
6		Moves the jack to the chosen position.
7		Next player’s turn

Alternative flow: If the path has a stop within the reach of how many steps the player should take, the player have to stay on the stop

	Actor	System
6.1		Moves jack to next station.

9. Use Case: Win Game

Summary: A player collects the kitten and hurry to get home first in order to win before some other player gets a Västtrafik-card and gets home before the first player.
Priority: High
Extends:
Includes: New game, Game over.
Participators: The user

Normal flow of events: The user collects the cat, wins the game and chooses to play again.

	User	System
1	User collects the cat and goes to either “Emilsborg” or “Olofshöjd” (depending on where the user started) before any other user collects a Västtrafik-card and goes to their starting place.	
2		Show pop-up with the text “Congratulations, Player ## won!”. See UC Game Over.

Alternate flow: The user collects the cat, but another user collects a Västtrafik-card, wins the game and chooses to quit game.

	User	System
1.1	A user collects the cat, but another user collects a Västtrafik-card shortly after and goes to either “Emilsborg” or “Olofshöjd” (depending on where they started) before the user with the cat manage to get there.	
		Show pop-up with the text “Congratulations, Player ## won!” See UC Game Over.

10. Use Case: Game over

Summary: A player has just won and can choose to play the game again or finish the application.
Priority: High
Extends: Win game
Includes: Start Game
Participators: The user

Normal flow of events: The game is finished. The user want to play again.

	Actor	System
1	Clicks button “Nytt spel”.	
		See UC Start Game.

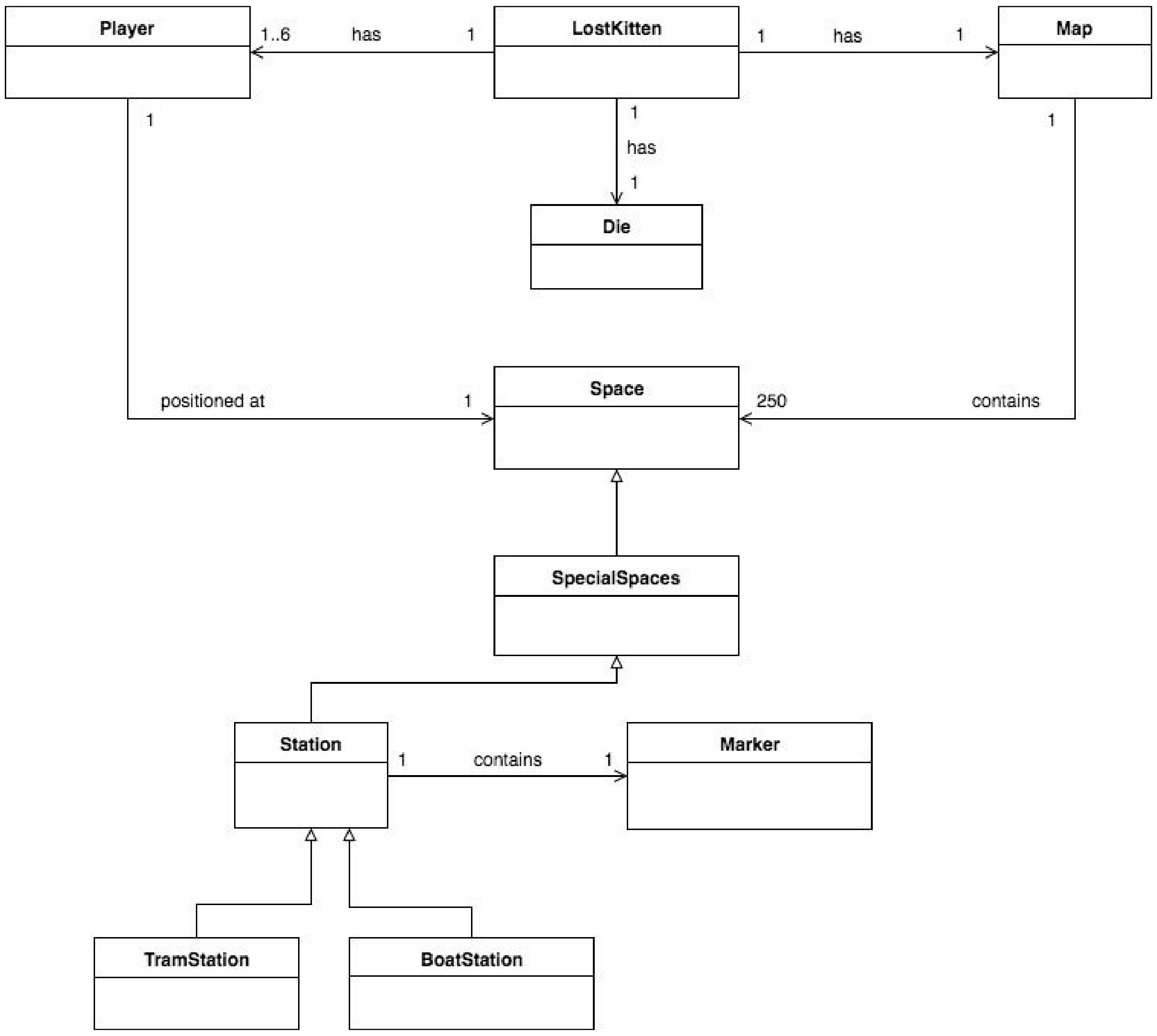
Alternate flow: The game is finished. The user wants to quit the application.

	Actor	System
2.1	Clicks button “Avsluta”.	
		Closes application.

Alternate flow: The game is finished. The user wants to quit the application.

	Actor	System
2.1	Clicks the red cross-button in the corner.	
		Closes application.

4 Domain model



4.1 Class responsibilities

LostKitten

- the overall representation of the game

Player

- There can be 2 to 6 players who play the game. A player is always positioned at a space on the map and the player class knows which space the player is positioned at.
- The player class has variables that knows the players budget and if the player has a tram card or have find the lost kitten.

Map

- The map contains a fixed number of spaces and represent the game board.

Die

- Represents a dice, gives a random number between 1 and 6.

Space

- A space represent a location on the board, which may hold a player.

Station

- A station is a special space which contains a marker.

Marker

- Contains different actions that can occur for a player.

  MoneyMarker

  - Contains gems with a value which will be given to the player if he or she flips a MoneyMarker.

  OtherMarker

  - Contains pickpockets, blank, cat or tram card.

5 References

Afrikan tähti, Wikipedia, https://en.wikipedia.org/wiki/Afrikan_tähti, 2017-04-01