Requirements and Analysis Document

Version 2.0
Date: 29/5-16

Group 7

Andrea Buchholz
Rebecca Finne
Maija Happonen
Lisa Larsson

This version overrides all previous versions.

Table of Content

1 Introduction

The following chapter will give a brief description and overview of the project.

1.1 Purpose of application

The goal for the project is to create a computer based generic version of a 2D, top-down action game. The game will have similar characteristics as Pokemon Silver [1], but instead of the aim of the game being to leve and catch Pokémon, the project will focus on a more simple gathering of articles through missions. While the game will be created generically, a demo will be created - illustrating part of Chalmers campus and the collectable articles will be mascot of different programmes at Chalmers.

1.2 General characteristics of application

The application will be a standalone, single-player, desktop application for Windows/Mac/Linux platforms.

Since the application is single player the game will be solely controlled by the main user.

1.3 Scope of application

The application is a single player game. The game will include interactions with a set amount of characters of two different types; Mascots and Humans. Interacting with the humans gives you two different options, recieving information about a programme or about the location of a mascot. Interacting with a mascot will instead involve answering a question with four alternative answers to chose from. If right, than the mascot is caugt, if not, the mascot will run away in a set pattern.

1.4 Objectives and success criteria of the project

1. The objective is for the game to be playable, where mascots can be caught, information can be gathered from humans - and more basic - for the player to have collision and move around in the game and to different worlds.
2. The humans, mascots and world will be created with Chalmers in mind, and even though we would like to create all of the campus/mascots, a few will due.

1.5 Definitions, acronyms and abbreviations

- GUI - A graphical user interface
- Java - The programming language used in this project [2].
- Mascot - A figure that is a symbol of a certain programme at Chalmers.

- World - The map in which the player interacts (there are many in each game, for different facilities).
- IntelliJ - The IDE (Integrated Development Environment) used for developing the application.
- TileMap - The development environment for the interface.
- Progress bar - A visual bar in the game that shows the progress of the game.

2 Requirements

The following chapter will give a deeper description of the requirements of the project.

2.1 Functional requirements

Our list of high level functions that are crusial to the game are:

1. Start a new game
2. Move around in the world. While moving he or she can:
    a) Interact with human
    b) Enter new worlds (i.e when going through a door/around campus)
    c) Catch mascots
3. Progress bar which shows the progress
4. Exit the application

2.2 Non-functional requirements

2.2.1 Usability

The language used will be English - as for the game to be available to as many people as possible. The information will be displayed in a fashion which is easily grasped quickly. There will be information that can be displayed in the beginning of the game that tells the user what the goal is and how to accomplish the tasks.

The usability will be tested on friends and family throughout the development process as to communicate the right amount of information.

2.2.2 Reliability

Not applicable to this project

2.2.3 Performance

When the user gives any input, i.e. press a workable key, there should not be a long wait for reaction from the game - at least not longer than 2 seconds.

2.2.4 Supportability

The application will be able to work on different computers. The application will be able to run on a screen with at least 1366x768 resolution.

The application will not be a networked based game, therefore the user will not be dependent on access to a network.

The application will be tested both through on-going GUI testing throughout the development, but also through JUnit-testing of specific methods.

Since the specific game is a demo, the main application should be able to work with any given map (world), any given mascot (sprite) and any given information (text).

2.2.5 Implementation

To be able to run the application a JRE have to be installed - which in our case will be IntelliJ.

2.2.6 Packaging and installation

The application will be able to download from git. A README-file will be written to assist in this.

2.2.7 Legal

This is not a commercial product. We will use trademarked figures and the game is overall a lot like Pokémon Silver, this may give some legal issues if this was a commercial product.

2.3 Application models

2.3.1 Use case model

The five most important use cases, which are further defined in Appendix C are:

- Move
- Interact with human
- Enter new world
- Catch mascot
- Progress bar

2.3.2 Use cases priority

The use cases will be prioritized in the following order:

1. Move
2. Enter new world
3. Catch mascot
4. Progress bar
5. Interact with human

2.3.3 Domain model

The domain model for the project can be found in Appendix A.

CRAP is the key model-class, where all the Characters, i.e the player, the mascots and the humans are initialized. The world is also defined in this class. All the characters also have positions, as well as the mascot having question and human information. CRAP also have progress, in which the count of caught mascots are stored.

2.3.4 User interface

The application will be able to integrate most type of worlds with the same set of environments and ground rules, as the maps will be created in TileMap [3].

The main screens for the user is starting the game, and playing the game.Can be seen in appendrix B.

References

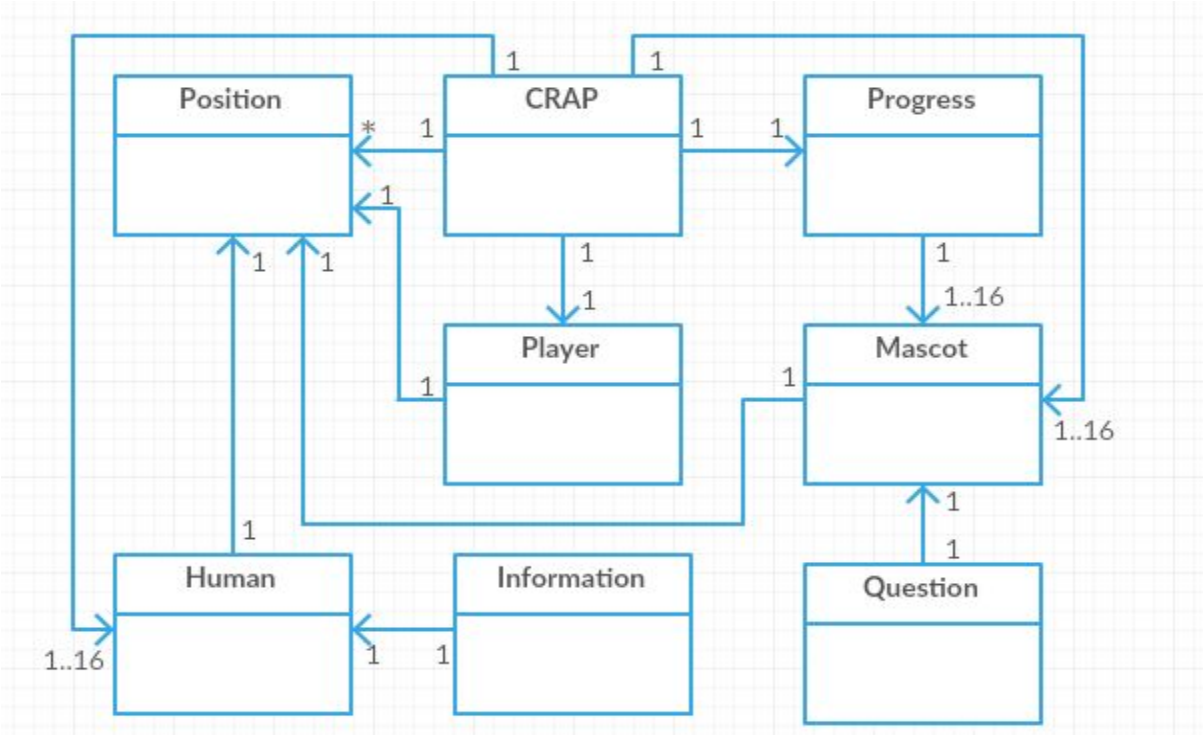
1] Wikipedia, retrieved 2016-04-11:
https://en.wikipedia.org/wiki/Pok%C3%A9mon_Gold_and_Silver

2] Wikipedia, retrieved 2016-04-11:
https://en.wikipedia.org/wiki/Java_(programming_language)

3] Tiled, retrieved 2016-04-11
http://www.mapeditor.org/

Appendix

Appendix A



FigureA: The Domain model of the game CRAP.

Appendix B

Figure B1: The start screen of the game as well as the playable game, where the small person is our main player.

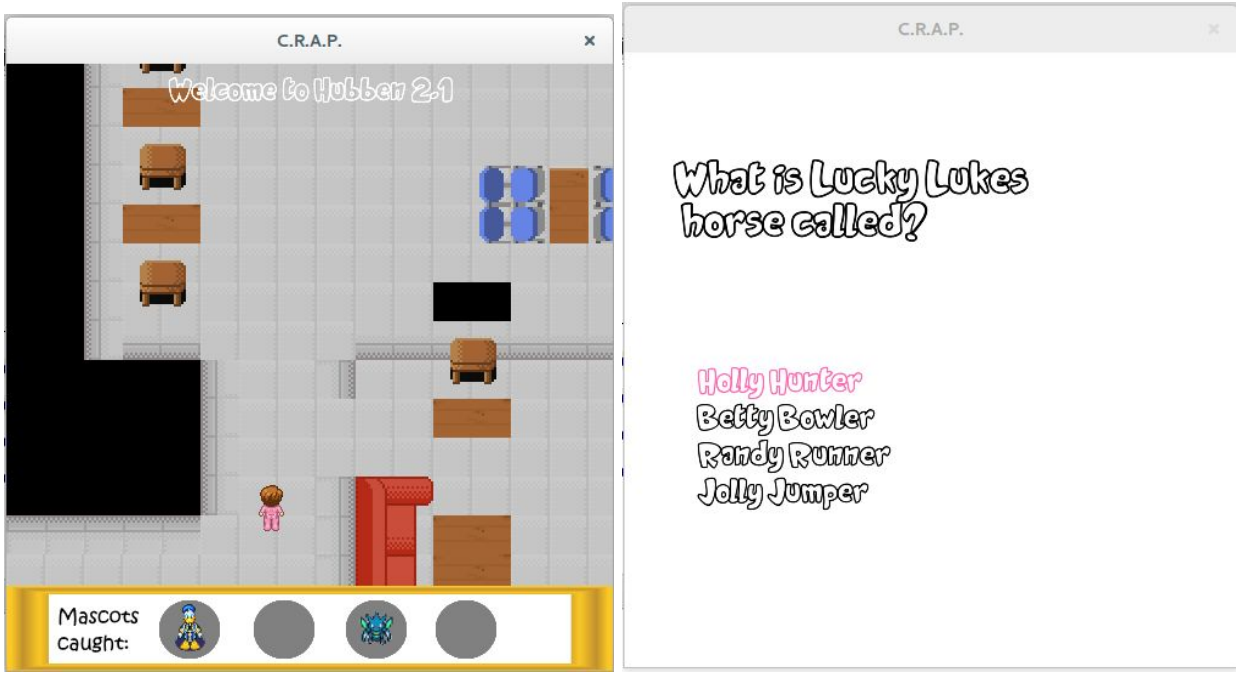


Figure B2: The view when entering a room (with progress bar filled with some mascots), as well as the view when interacting with a character.

Appendix C.

Use Case: EnterNewWorld

Summary:

The player moves into a new world such as a new building or a contniues of the same area. When this happens a new world is loaded onto the screen.

Priority: High

Extends: -

Includes: Move

Participators: The player and the application.

Normal flow of events:

The player walks up to the door of a building.

	Actor	System
1	Player moves off the screen.	
2		The world/surroundings is changed to the new one.
3		Text message showing the name of the new world.

Exceptional flow

There is no exceptional flow.

Use Case: Move

Summary: This is how the player moves their character throughout the world.

Priority: High

Extends: -

Includes: -

Participators: The actual player

Normal flow of events:

Moving one step without any obstacles in the way.

	Actor	System
1	Clicks one of the 4 buttons that make the player move (up, down, left, right)	
2		The character moves in the direction of the button pressed

Alternate flows:

2.1 Moving into a solid object, will make the character not able to move

	Actor	System
2.1.1		Character walks into an obstacle i.e. tree, stone or wall.
2.1.2		Character can not walk through the obstacle and stays in the same position.

2.2 Character moves in areas with tough terrain

	Actor	System
2.2.1		Character walks into a tougher terrain i.e. bushes or steps.
2.2.2		Character walks slower in the tougher terrain.

Exceptional flow

There is no exceptional flow.

Use Case: CatchMascot

Summary:

This is how the player get progress in the game. In this demo the player catches the mascots from different programmes at Chalmers. The player finds mascot and interacts, the mascot asks a question with multiple choices and the player tries to answer right. If the player answers the question right the player “catches” the mascot and the mascot is added to the progress bar, if not the mascot runs away.

Priority: High

Extends: Interact

Includes: Progress bar, Move

Participators: The player and the application

Normal flow of event:

If the actor answers the question wrong.

	Actor	System
1	Walks up to mascot	
2		Get a talk bubble from the mascot with a question and alternative.
3	Player answers wrong	
4		Mascot runs away a few steps

Alternate flows:

3.1 If answers is right

	Actor	System
3.1.1	Answers right from the alternatives shown.	
3.1.2		Mascot added to the progress bar

Exceptional flow

There is no exceptional flow.

Use Case: ProgressBar

Summary:

A bar that shows how many mascots the player have catched.

Priority: high

Extends: Catch mascot

Includes:

Participators: The player and the application

Normal flow of events:

When player catches a mascot a picture of the mascot are added to a bar in the lower area of the screen.

	Actor	System
1	Catches mascot	
2		Adds a picture of the mascot catched to the progress bar

Alternate flows:

2.1 Progress bar is filled to the maximum.

	Actor	System
2.1.1		Adds a picture of the mascot to the progress bar
2.1.2		Music starts playing and a big pop-up is displayed with congratulations
2.1.3		Returns to the main menu

Exceptional flow

There is no exceptional flow.

Use Case: InteractWithHuman

Summary:

The player interacts with people around campus.

Priority: High

Extends: -

Includes: CatchMascot, Move

Participators: The player and the application

Normal flow of events:
Talk to human, the player walks up to human and chooses if the player wants to talk to the human, ask which programme the human studies, ask for a clue where the mascot for the programme could be or to quit the conversation. The player can then choose to ask more or to exit the conversation.

If “clue where mascot can be” is chosen and then having enough information.

	Actor	System
1	Walks up to human	
2	Choose the “clue where mascot is” from the alternatives.	
3		Talk bubble with information.
4	Chooses “exit” from the inteaction.	
5		Talk bubble closes

Alternate flows:

2.1 If “clue where mascot can be” is chosen

	Actor	System

2.1.1	Choose the “information about programme” alternative.	
2.1.2		Talk bubble with information.

2.2 If option “talk” is chosen

2.2 If choosing to quit the conversation.

	Actor	System
2.2.1	Choose the “cancel” alternative.	
2.2.2		Talk bubble will close.

Exceptional flow

There is no exceptional flow.

Appendix D

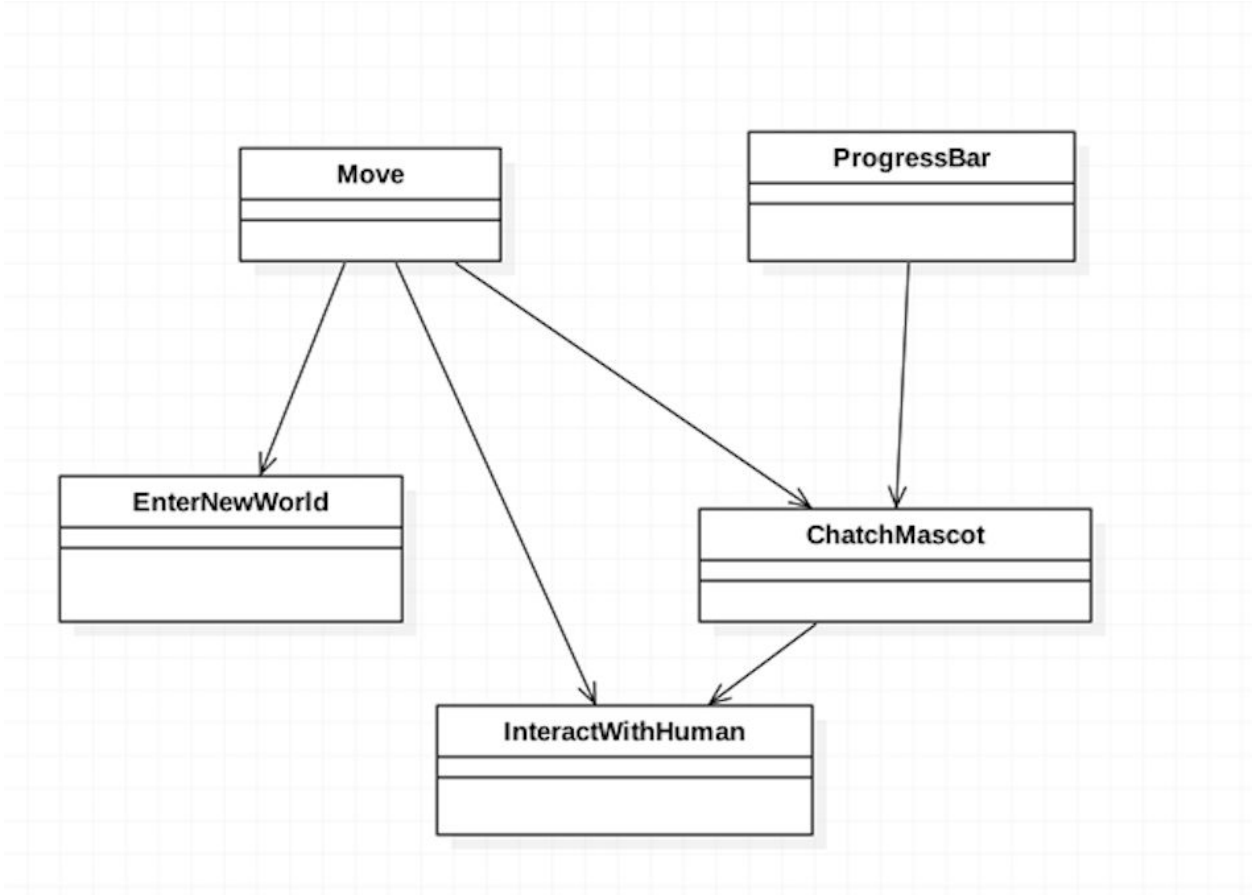


Figure D: The Use case diagram of the game CRAP.