Requirements and Analysis Document for Mosquito Simulator

Version: 1.4

Date: 31/05-15

Author: Anton Solback, Rasmus Davidsson, Alexander Sandberg, Johan Ljungberg

This version overrides all previous versions.

1 Introduction

1.1 Purpose of application

The purpose of this project is to make a 3D game purely for entertainment.

1.2 General characteristics of application

The application will be a single-player, desktop game. The player will fly through the 3D world using the keyboard. The game will be highscore based which means the player will get a score after each run which he/she then will try to improve the next run. A run will end when the player runs out of energy. Energy can be gained during a run by sucking blood. If the player runs out of energy, his/her score will be saved and then he/she can try to improve the previous score by restarting with the simple press of a button.

1.3 Scope of application

The application does not support any multiplayer services. It is not possible to play over the internet.

1.4 Objectives and success criteria of the project

1. Fly around in a 3D environment as a mosquito.
2. To be able to collide with humans and solid objects.
3. Have a simple menu at start with the options to play, quit game, see highscore and reset highscore.
4. Get score over time.
5. Save highscore.
6. Use the blood you get from humans as energy.
7. Have one objective that the player can complete.

1.5 Definitions, acronyms and abbreviations

- Java - a independent programming language
- 3D - three dimensional
- jMonkeyEngine - A game engine for making 3D games in the java language.
- JRE - The Java Runtime Environment, software needed to run an Java application.
- GUI - Graphical User Interface
- Blender - A drawing tool for making 3D models.
- IDE - Integrated development environment.
- Eclipse - An IDE program.
- Score - The player will gain score constantly while playing.
- Energy - The player start with a full energy bar, which will successively drop while flying. By actively sucking blood from humans the player can refill the energy.
- Objective - There will be optional tasks you can complete which will give you more score.

2 Requirements

In this section we specify all requirements

2.1 Functional requirements

Create a list of high level functions here (from the use cases).

1. Click “Play” on the main menu to start a new game.
2. Navigate through the 3D room.
    a. Move forward, backwards, sideways, up and down.
    b. Change the facing of the mosquito.
3. Suck blood from human
4. Save highscore

2.2 Non-functional requirements

2.2.1 Usability

Usability is a very important part of this program. There should be no question on how to play our game. It will use some of the commonly used controls to move the mosquito. Then we are going to add some custom controls to suck blood and for rotation.

2.2.2 Reliability

NA

2.2.3 Performance

All the graphical components except the text, on the screen should always be seen, and scaled properly to all different resolutions. Every action you do ingame should not take more than 2 seconds to complete.

2.2.4 Supportability

The application should only be implemented to suit PC and Mac.

There should be tests that checks all the use cases and the main parts of the model.

2.2.5 Implementation

To achieve platform independence the application will use the Java environment. All hosts must have the JRE installed and configured as well as the .jar-file for the application.

2.2.6 Packaging and installation

This application will be delivered as an zip-archive containing;

1. An executable file for the game (a Java jar-file).
2. All needed resources such as files, models, images and icons etc.
3. A README-file with description of installation and usability.

2.2.7 Legal

The mosquito sound effect we have in our game is not under a Creative Commons licence. Therefore if we were to publicly publish this game we would have to expect to pay a license fee.

2.3 Application models

2.3.1 Use cases priority

1. Be able to fly around in a 3D environment.
2. Suck blood from human models.
3. Achieve score by sucking blood.
4. Complete objectives

2.3.2 Domain model

See appendix.

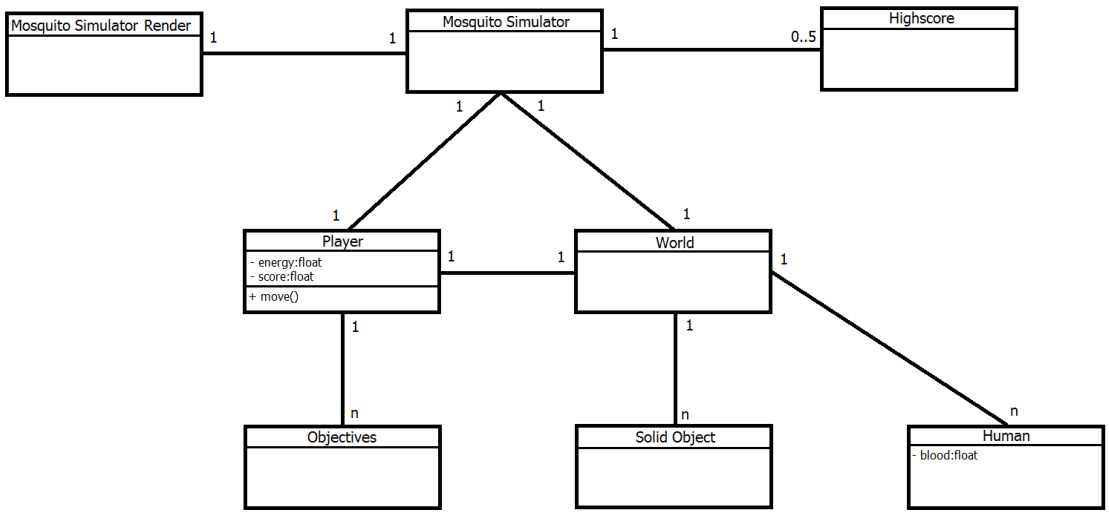
2.3.3 User interface

Application will start with a screen resolution options menu of the game. A simple menu with only a couple of buttons. Then another simple menu will appear when you click on continue. The window will also be resizable to the biggest scale as well as the smallest one. You can not edit nor change the GUI.

2.4 References

APPENDIX

Domain model



Use Case: Fly Forward

Summary: This is how the player moves forward in the 3D-world.

Priority: Very high

Requirements: Run game

Includes: ?

Participators: The player

Normal Flow:

	Actor	**Application**
1	Presses down ‘W’	
2		Moves the mosquito forward
3	Releases ‘W’	
4		Mosquito stops moving forward

Alternative flows:

2.1 Mosquito collides with human

	Actor	**Application**
2.1.1		Possibility to suck blood with ‘q’-button

2.2 Mosquito collides with non-human

	Actor	**Application**
2.2.1		Mosquito stops moving forward

Use case: Suck blood

Summary: This is how the player sucks blood from humans, and thereby gain energy.

Priority: Normal

Includes: ?

Participators: The player

Requirements: Collide with human

Normal flow:

	Actor	**Application**
1.	Presses down ‘Q’	
2.		Gain energy
3.		Human lose blood
4.	Releases ‘Q’	
5.		Lose energy

Alternative flows:

3.1 No available blood

	Actor	**Application**
3.1.1		Lose energy

Use case: Look around

Summary: Rotates the camera and the direction of the mosquito.

Priority: High

Includes: ?

Participators: The player

Requirements: Run game

Normal flow:

	Actor	**Application**
1.	Press left/right arrow key	
2.		Rotates the mosquito left or right

Use Case: Fly Backwards

Summary: This is how the player moves backwards in the 3D-world.

Priority: Very high

Requirements: Run game

Includes: ?

Participators: The player

Normal Flow:

	Actor	**Application**
1	Presses down ‘S’’	
2		Moves the mosquito backwards
3	Releases ‘S’	
4		Mosquito stops moving backwards

Alternative flows:

2.1 Mosquito collides with human

	Actor	**Application**
2.1.1		Possibility to suck blood with ‘q’-button

2.2 Mosquito collides with non-human

	Actor	**Application**
2.2.1		Mosquito stops moving backwards