Requirements and Analysis Document for "How do I fly this thing"

Table of Contents

Version: 0.1

Date: 2014-03-19

Author: Joakim Thorén, Francine Mäkelä, Mathias Carlsson, Martin Nilsson

This version overrides all previous versions.

1 Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

The projects aims to create a networked multiplayer game in which players control a spaceship with firing capability. The players battle in a restricted zone within space and are supposed to destroy player opponents.

1.2 General characteristics of application

The application will be a desktop, standalone, networked multi-player application
with a graphical user interface for the Windows/Mac/Linux platforms/ using only the keyboard for ensuring optimal laptop experience.

The application will be real-time. A user hosts a game, to which another player can connect to with IP. Directly when two players are inside the game a round starts. In this round the goal is to eliminate other players by maneuvering the ship with thrusters, which alters the spaceships velocity, and shooting projectiles at other players. There are abandonned space-stations in the space "arena" in which upgrades and/or powerups will be found. Colliding with a space-station will severly damage the hull of the ship. If another players joins the game he will have to wait for next round to start in order to spawn. Last man standing wins.

1.3 Scope of application

There is no reason to play this game alone, and therefore this will be impossible. The game won't allow you to save a game. There won't be any server application, only direct connection to host. Graphics are entierly 2D and very basic 2D, no special effects. No stats are saved permanently (no database). If host disconnects or shutdown his game session, there will be no host-transfer.

1.4 Objectives and success criteria of the project

Bellow are features of the game which should be implemented in order to consider de project a success:

- Main menu
    - Host game
    - Join game
    - Options (to adjust video settings and audio)
    - Exit

- 2D graphics representing game world
- Audio (sound effects when shooting, getting hit etc)
- Maneuverable spaceship with 3 different thrusters which move the ship and 1 gun which can fire
- Thrusters should move the ship based on where they are located. Thruster on bottom-right rotates ship anti-clockwise and move forward slightly, bottom-left thruster rotates clockwise and move forward slightly, middle thruster move forward fast. The general "feel" of maneuvering the ship should be similar to the game Rakete, see reference for link to Rakete-website[1].
- Spaceship keep their speed if thrusters isnt used (no gravity)
- Camera (screen) is adjusting itself to always show what's infront of the ship
- Users able to connect to an host via IP-adress
- When 2 or more players are connected a round will start
- Temporary statboard for each game session
- Procedurally generated abadonned spacestations each round in which upgrades/powerups/equipment can be found
- Whenever a player enters a host and a new round is about to begin he spawns with a spaceship
- Spaceships have:
    - Hull (hitpoints)
    - Sheild (hitpoints that regenerates. Not as strong as hull)
    - Weight (alters movement speed. Increases if gun is a heavy-gun or if ship has much hull.)
    - 1 Gun
    - 3 thrusers which are independently controlled with key on keyboard

[1] http://www.mariov.ch/portfolio/project/rakete, Rakete by Mario von Rickenbach

- Players can get hit by bullets shot by other players
- Players can collide with structures which causes massive amount of damage directly to hull
- Player spaceship explodes if it's hull is 0.
- Once there is a last man standing the temporary staboard is displayed and a new round commences automatically within couple of seconds
- Users can quit the game at any time via ESC -> Quit game
- Users can disconnect from a game at any time via ESC -> Disconnect

1.5 Definitions, acronyms and abbreviations

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).

2.2 Non-functional requirements

Possible NA (not applicable).

2.2.1 Usability

2.2.2 Reliability

2.2.3 Performance

2.2.4 Supportability

2.2.5 Implementation

2.2.6 Packaging and installation

2.2.7 Legal

2.3 Application models

2.3.1 Use case model

//UML and a list of UC names (text for all in appendix)
See appendix for UML.

2.3.2 Use cases priority

High:

- Start game
- Host
- Exit
- Move ship
- Shoot

Medium:

- Join
- Join game

2.3.3 Domain model

UML, possible some text.

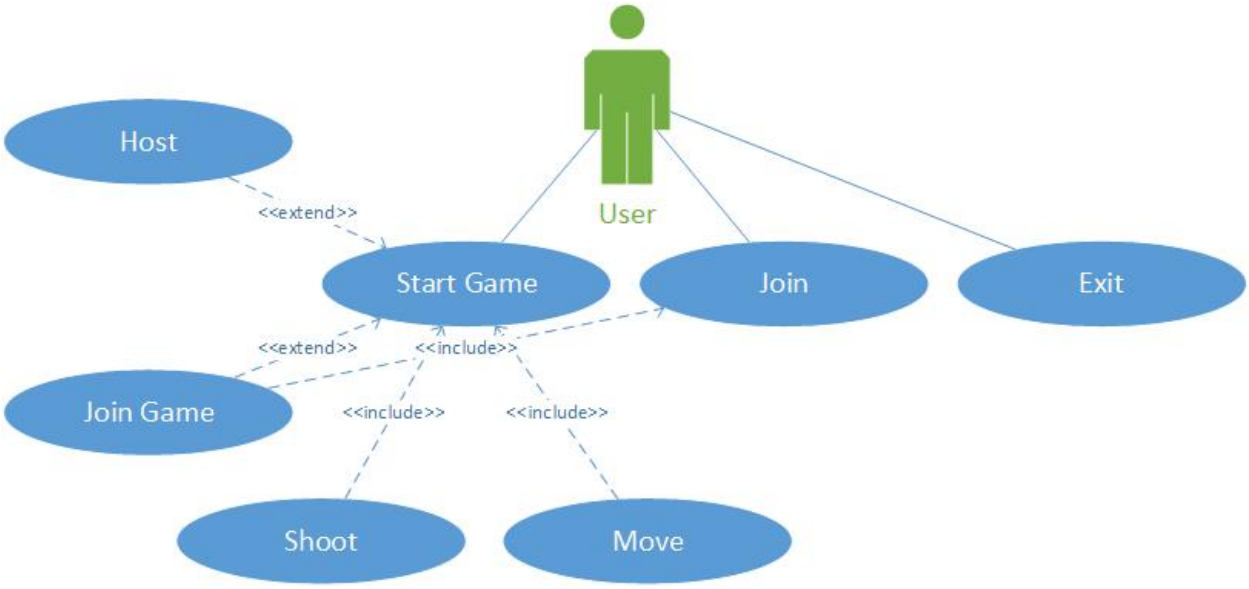
2.3.4 User interface

Text to motivate a picture.

2.4 References

APPENDIX

Use cases



Use case: Start game

Summary: A user can start a game which removes main menu and displays the world (space) in which he can control his spaceship. UC Host and UC Join game extends this UC.

Priority: High

Extends: -

Includes: -

Participators: -

Normal flow of events

Game world is loaded

	Actor	System
1	Clicks button which loads game world	

2		Load game world
3		Display game world

Use case: Host

Summary: This is how the user host a game for others to join. Top alternative in main menu which is shown on application startup.

Priority: High

Extends: Start game

Includes: -

Participator: User

Normal flow of events

User hosts a game

	Actor	System
1	Clicks the host button	
2		Stop displaying menu
3		Start game-loop

Use case: Join

Summary: This is how an user connects to an already hosted game. Alternative in main menu which is shown on application startup.

Priority: Medium

Extends: -

Includes: -

Participators: User

Normal flow of events

User is prompted for IP to host

	Actor	System
1	Clicks the Join button	
2		Stop displaying menu
3		Prompts for IP to host

Use case: Join game

Summary: Preceded by UC Join. When IP to host is entered and join game button is pressed, user will join the game.

Priority: Medium

Extends: Start game

Includes: Join

Participators: User

Normal flow of events:

User joins the host

	Actor	System
1	Clicks the join game button	
2		Connect to host
3		Display game world from host

4.		Spawn ship on next round

Use Case: Exit

Summary: On selecting exit alternative in main menu

Priority: High

Extends: -

Includes: -

Participator: User

Normal flow of events

Application is closed

	Actor	System
1	Clicks the Exit button	
2		Application is closed

Use Case: Move ship

Summary: When user presses any of the thruster keys the ship will activate corresponding thruster and move accodringly to laws of physics.

Priority: High

Extends: -

Includes: UC Start game

Participators: Player

Normal flow of events

Player ship moves

	Actor	System
1	Clicks any thruster key	
2		Move ship according to thruster

Use Case: Shoot

Summary: On pressing the shoot button in game

Priority: High

Extends: -

Includes: -

Participator: User

Normal flow of events

Projectiles fires from the ship, in the proper direction.

	Actor	System
1	Presses shoot button	
2		Ship fires shots.

GUI

Domain model

Use case texts