Requirements and Analysis Document for the Sänkoss Project (RAD)

Contents:

Version: 1.4

Date: 24/5 - 2014

Authors: Fredrik Thune, Daniel Eineving, Mikael Malmqvist, Niklas Tegnander

This version overrides all previous versions.

1 Introduction

1.1 Purpose of application

The sole purpose of the project is to create an application that is engineered to entertain and sustain the users in form of a light-weight computer game based on the classic board game Battle Ships. For rules, definitions and further specific details about the game, se references!

1.2 General characteristics of application

The application will be engineered for a desktop environment, (Linux/Windows/OS X) as a multiplayer, based-turned game to be played over network.

The game will, as mentioned use a turn-based player alternation with no time constraint for each turn. This means that a player's turn is over only when he/she has shot at the aim board.
Furthermore the game will come to an end whenever a player has successfully neutralized all of the opponent's battle ships, by hitting the corresponding length of the ship, or when a player decides to end the current game manually.
To represent the game visually a 2D GUI similar to the original game, with a player board and an aim board, will be used.

1.3 Scope of application

The game only supports playing against a human opponent, over network, and not against an artificial intelligence, or a local game played on a single split screen. Therefore network connectivity is a requirement for all players.

1.4 Objectives and success criteria of the project

1. A player should be able to establish a connection to another player through a dedicated server.
2. A player should be able to finish a game against a human opponent using the GUI.

1.5 Definitions, acronyms and abbreviations

- Java: Platform independent programming language
- JVM: Java Virtual Machine
- GUI: Graphical user interface
- JRE: The Java Runtime Environment. Additions software needed to run a Java application
- Host: Computer to run the game on (a player)
- Dedicated server: Server to route the communication between hosts.
- Round: A complete game with a winner and a loser

2 Requirements

2.1 Functional requirements

The player(s) will be able to:

1. Start multi player game.
    a. Create a room.
    b. Join specific room.
2. Start game within a room.
    a. Select nationality.
    b. Place ships and rotate ships on the grid.
3. Take a turn. During the turn, player will be able to:
    a. Fire at a target location on the aim board.
    b. End the turn.
4. Exit the application. Will end turn and round.

2.2 Non-functional requirements
Possible NA (not applicable).

2.2.1 Usability

The game mechanics should be as self explanatory that a normal user are able to play the game right

away, under the condition that they know the ground rules of the game.
User tests with two to four non-computer savvy people should make sure that the interface follows the criteria.

2.2.2 Reliability

The client application needs to be bug free and robust to manage normal game play without crashing. This applies to the server as well.

2.2.3 Performance

A client-side action initiated by the player should not exceed 5 sec. response time, while the server is allowed up to 30 sec response time before the game shuts down because of slow internet connection.

2.2.4 Security

All data must be retrieved in a secure way. This means that the client application must be written in such a way that it is impossible to modify the code in order to access vulnerable data from an opponent.

2.2.5 Supportability

The application only has support for desktop only.
It should be easy to create an AI to play against the player.
It should also be an easy task to switch the GUI to another - e.g switch boats to space ships.

2.2.6 Implementation

The client application and java JRE needs to be installed on all hosts.

2.2.7 Packaging and installation

The application is packaged in a JAR-archive format with all dependencies included. It should be trivial to start this archive through the JVM.

2.2.8 Legal

The license that is being used for the project is the MIT license (see References).

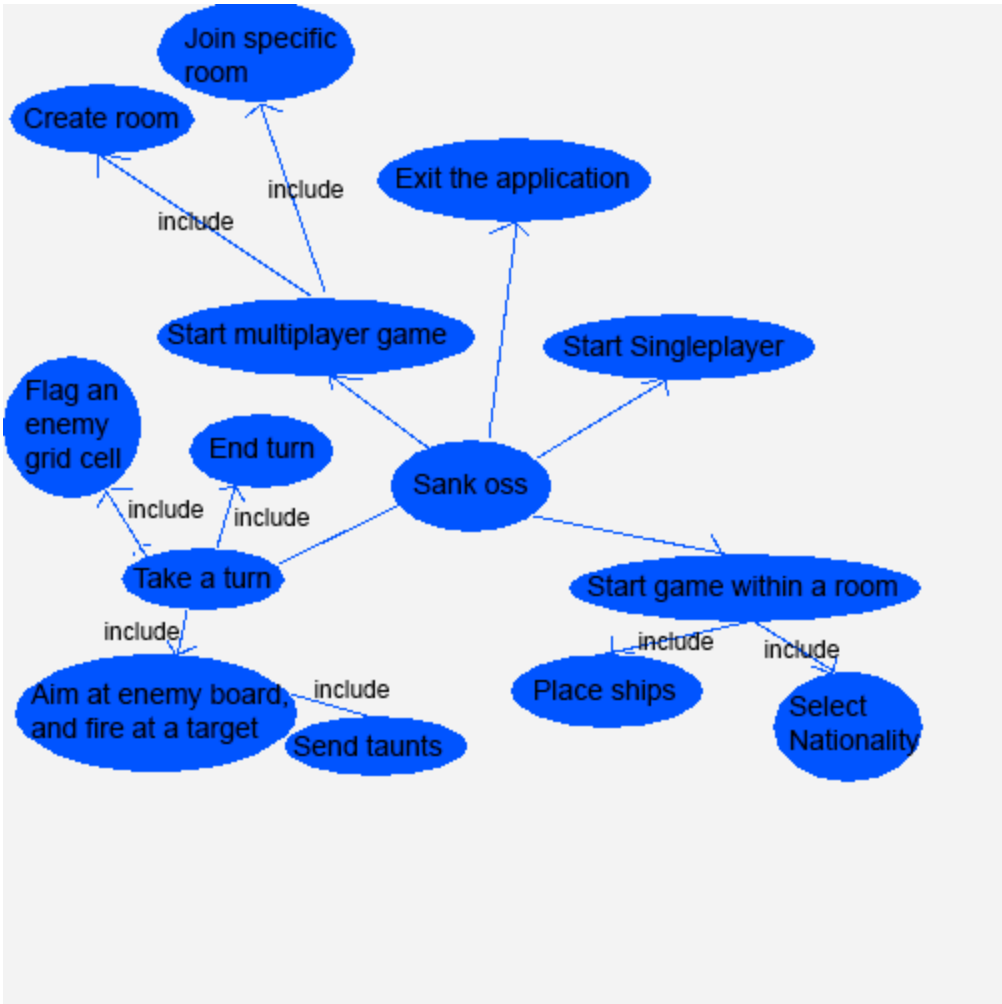
Ship sprites, as well as explosion sprite has been retrieved from external sources and are licensed under CC-BY 3.0 (see References)

2.3 Application models

Use cases:

- Change Names
- Enter Game
- Create Room
- End Game
- Set Nationality
- Place Flags (Aim board)
- Join Room
- Shoot at enemy
- Place Ships
- Singleplayer

2.3.1 Use case model



2.3.2 Use cases priority

1. Join Room
2. Create Room
3. Enter Game
4. Shoot at Enemy
5. Place Ships
6. End Game
7. Singleplayer
8. Place Flags (Aimboard)
9. Change Name

10. Set Nationality

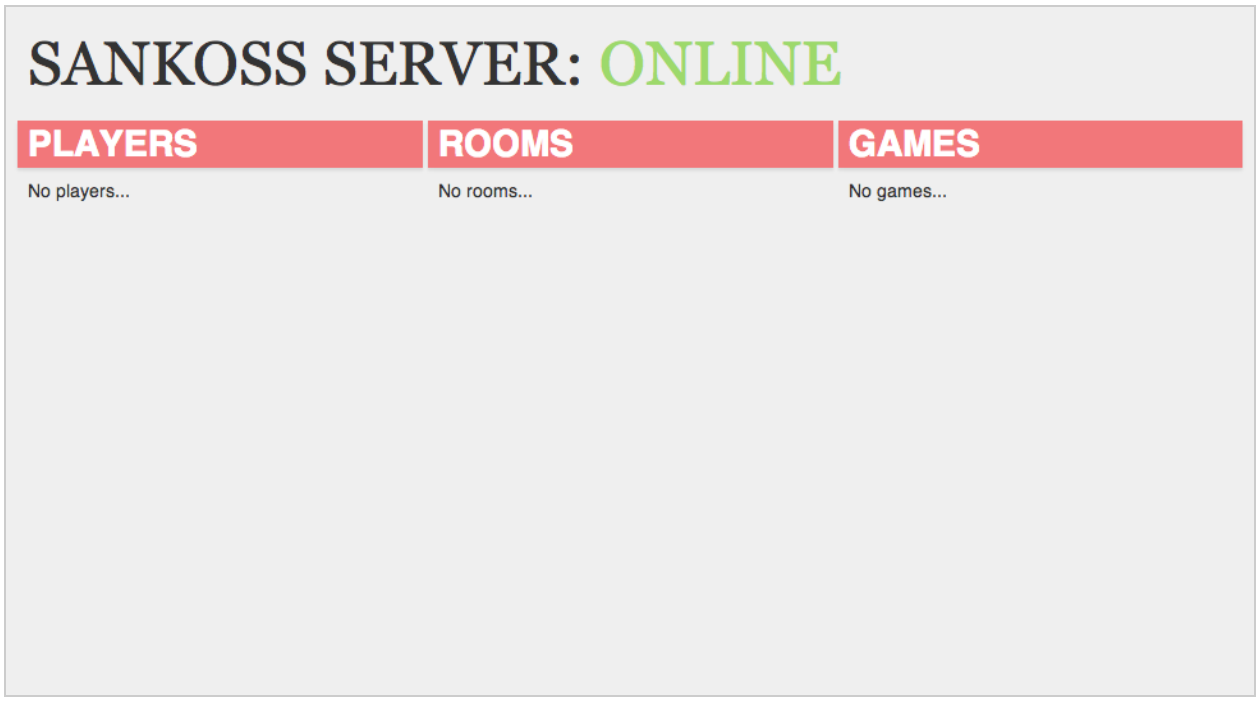
2.3.3 Domain model

See APPENDIX

2.3.4 User interface

The client application will use a fixed GUI, not to be themed or changed. The GUI must be able to adapt to different screen sizes and resolutions. See APPENDIX for screens and navigational paths.

The server side of the application will have a web interface where a user can see a list of connected players, current rooms and ongoing games.



2.4 References

Battle Ships rules (board game)

http://www.hasbro.com/common/instruct/BattleShip_(2002).PDF

MIT license

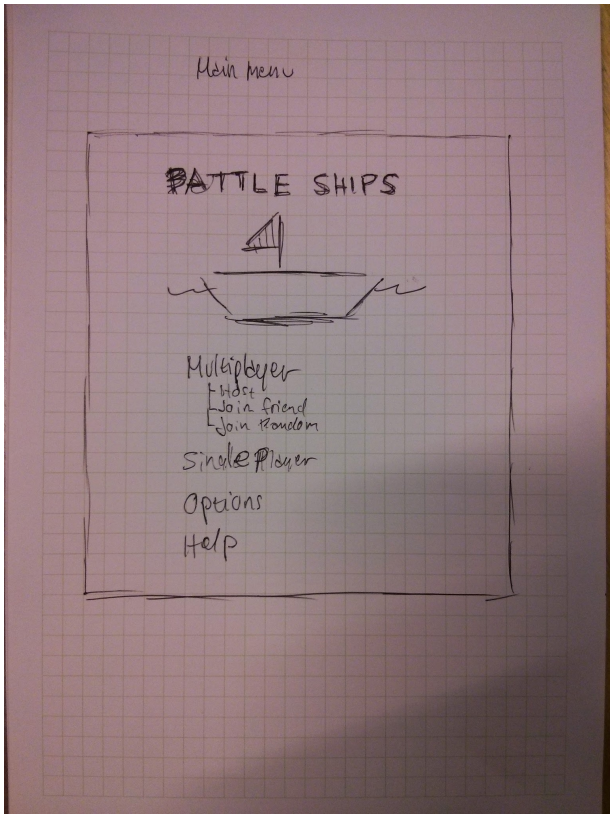
http://opensource.org/licenses/MIT

CC-BY 3.0

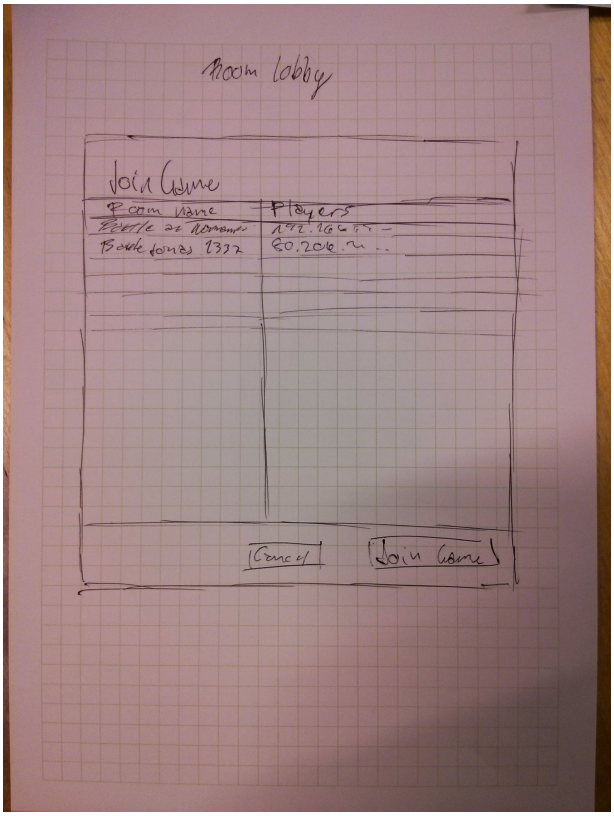
https://creativecommons.org/licenses/by/3.0/

APPENDIX

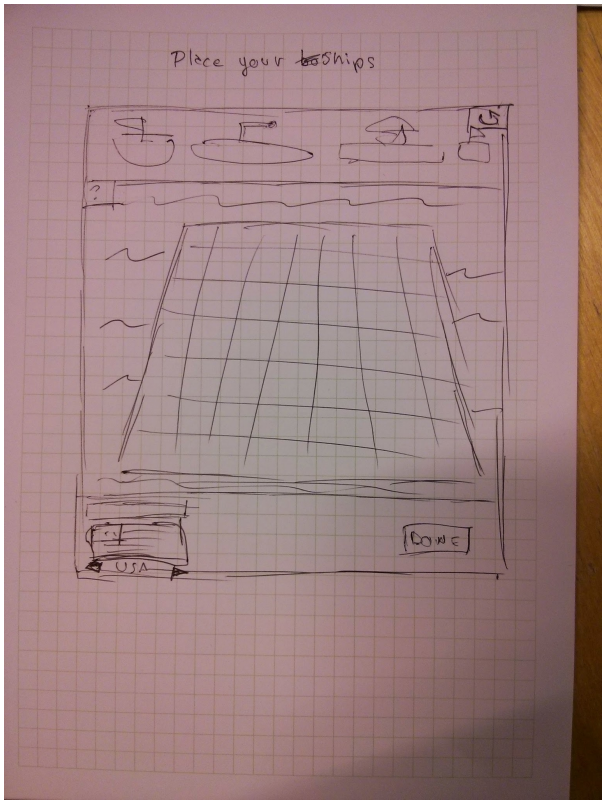
GUI - Mock-up



MainMenu



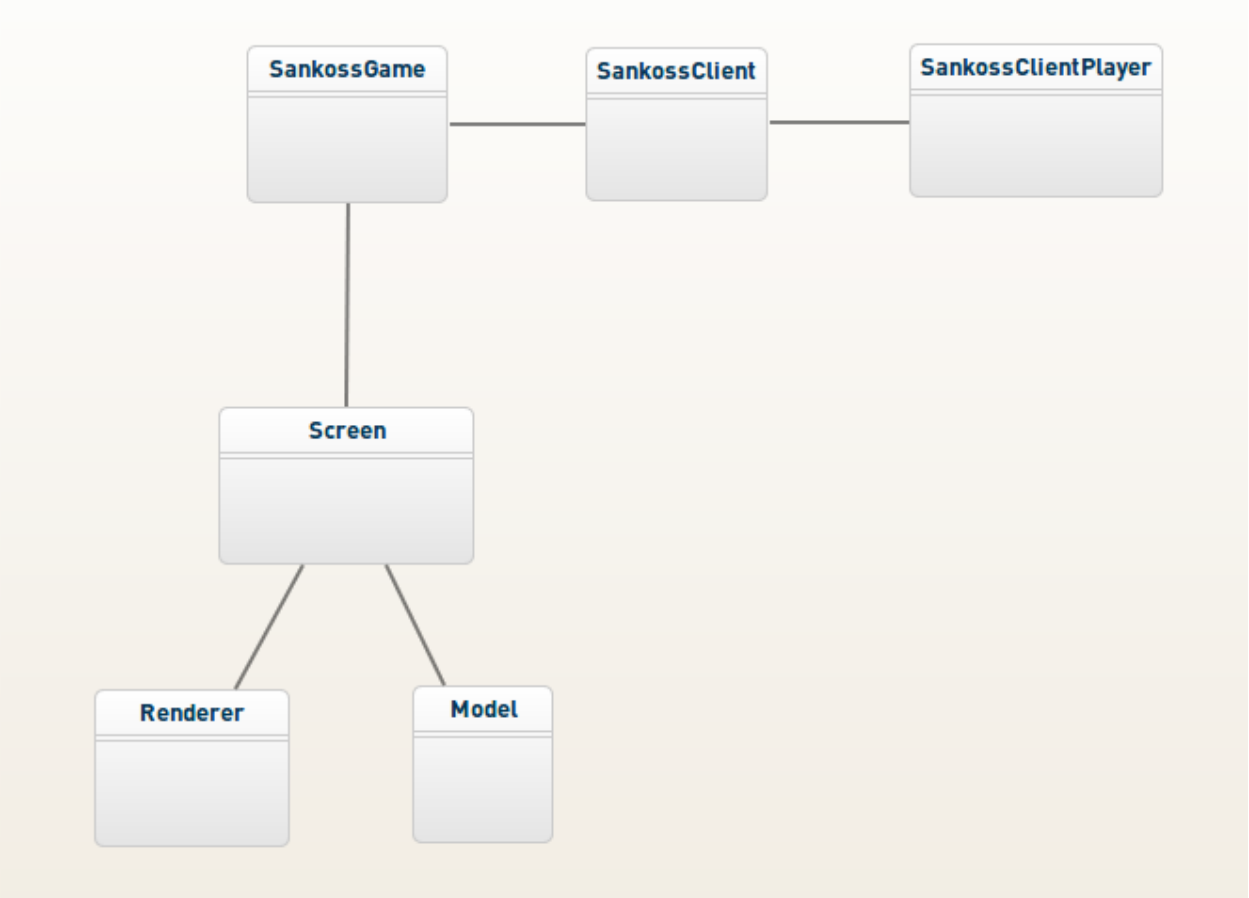
Lobby



Placement

InGame

Domain model



Domain model for the client application.

Use Cases

Use Cases can be found in the attached directory *Use Cases*