Requirements and Analysis Document for Project Warborn (RAD)

Contents

Date: 2012-03-14

Author: teoost

1 Introduction

This section is to give a brief overview of Project Warborn

1.1 Purpose of application

The goal of this project is to create a turn based strategy game based on the classic RISK, developed by Parker Brothers with slight modifications such as swappable maps and "classes" for simulating role play.

1.2 General characteristics of application

The application will be a desktop, standalone (non-networked), multi-player application with a graphical user interface for the Windows/Mac/Linux platforms.

The game will work with the same basic functionalities as the board game RISK, being a turn based strategy game with the goal of conquering the map by controlling all the Regions which in turn are built up by territories. Each turn will consist of three phases:
Reinforcements, Battle, Troop movement.
There will be no time limit to a turn and the order is randomised at the initialisation of a game. The winner is the one who successfully conquers the whole map or, if the game is cancelled, the one with the most territories. In the event of a draw, the number of troops are counted. Further winning conditions might be added to, if desirable, substitute the standard ones.

1.3 Scope of application

The application does not include a computer-player. It's impossible to play the game alone (a person can of course choose to play against herself). The application does not save interrupted games or collect any statistics (high score or other). See Possible future directions.

1.4 Objectives and success criteria of the project

1. It should be possible to play a full (covering at least 90% of the rules using Parker Brother's social instructions) multi-player game (see Definitions) on any of the platforms using a simple graphical user interface.

2. The game should be possible to play for at least two different locations (sets of regions etc. example: Gothenburg and Stockholm)

1.5 Definitions, acronyms and abbreviations

All definitions and terms regarding the core Monopoly game are as defined in the references section.

- GUI, graphical user interface.
- Java, platform independent programming language.
- JRE, the Java Run time Environment. Additional software needed to run an Java application.
- Multi-player Game, a game in which multiple participant are active.
- Host, a computer where the game will run.
- Round, one complete game ending in a winner or possible canceled.
- Turn, the turn for each player. The player can only act during his or her turn (place units, attack, move, etc.). Thou, the player can be affected during other players turns (i.e. get attacked by active player, etc.)
- Territories, a fixed, coloured location on the map, it can be controlled by only one player at any given time..
- Regions, a geographical collection of territories which may give you a bonus if you

control all territories included.

2 Requirements

2.1 Functional requirements
The players should be able to;
1. Select the location for the game (Stockholm, Gothenburg, etc.)
2. Start a new game.
    a. Select number of players and respective names and colours.
3. Do a turn. During the turn, he or she can
    a. Place extra numbers of units onto the map.
    b. Attack other players territories to conquer them
    c. Relocate currently placed units to other territories.
    d. End the turn.
4. Exit the application. Will end turn, round and game.

2.2 Non-Functional Requirements
2.2.1 Usability
The usability is not a major concern when the application is aimed towards a very IT-oriented audience so the need for user friendliness is diminished. This does not mean that the usability is unimportant but the focus of our efforts will be heavier on the functionalities and structure of the code.
2.2.2 Reliability
Very basic reliability, consisting of minimising the number of crashes.
2.2.3 Performance
Any command from a user should not take longer than one second to perform, worst case.
2.2.4 Supportability
The application should be constructed in accordance with the Model-View-Controller pattern and should be structured in a way so that any part is easily replaceable with a modified version. Furthermore, the Gui should be adaptable to a smartphone version using a server-client relationship for slow paced multiplayer gaming.
2.2.5 Implementation
The application will be a Java application and all platforms will need JRE 1.7 or later to run it.
2.2.6 Packaging and installation
//TODO complete
2.2.7 Legal
N/A

2.3 Application models
2.3.1 Use case model
See APPENDIX for UML diagram and textual descriptions.

2.3.2 Use Cases Priority
1. Move

2. Battle
3. Place troops
4. Use Souls
5. Show credits

2.3.3 Domain model
see APPENDIX.

2.3.4 User Interface
Application will use a simple GUI consisting of a different background image depending on which map is chosen and a matching overlay of buttons. It will also have a small HUD at the bottom of the screen displaying various statuses of interest to the player, along with buttons to navigate through the game and access more functionality.

APPENDIX

//TODO add images of UC diagram and Domain model.