Requirements and Analysis Document for the Monopoly project (RAD)

Contents

Version: 2.0

Date 2011-03-03

Author hajo

This version overrides all previous versions.

1 Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

The project aims to create a computer based generic version of the well known board game Monopoly by Parker brothers. Generic in the sense that it's should be possible to adapt the game to different locations and more, see further below. For definitions, terms and rules of the game see references.

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.

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 official instructions) multi-player game (see Definitions) on any of the platforms using a (possible simple) graphical user interface.

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

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.

- 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 (roll dices, buy, sell, etc.). Thou, the player can be affected during other players turns (i.e. pay to actual player, etc.)
- Resources (for players), the total value of the properties, buildings and cash of a single player. A player is bankruptcy when he or she has no more resources.

2 Proposed application

In this section we propose an application.

2.1 Overview

The proposed application will be a traditional desktop application. The application will be turn based. The actual player must explicitly end his or her turn. The next player is chosen by the application from a preset ordering. The ordering is generated randomly by the application at start of the round. There's no time constraints for a round. The application will end according to the rules or possible be canceled. If the game is canceled the player with most resources will be the winner. The application will handle all of the banks responsibilities. The application will use a GUI very similar to the "real" game.

2.2 Functional requirements

See also 2.4.2 . The players(s) should be able to;

1. Select the location for the game (London, Gothenburg, etc.)
2. Select a language for the GUI.
3. Start a new game.
    a) Select how many player for the game, what color for each player and what piece for each player.
4. Do a turn. During the turn, he or she can
    a) Roll a dice. This will possible trigger a response from the application (i.e. move the piece, show a dialog for Chance card, GO to Jail, get money when passing GO, etc.)
    b) Buy or sell properties and building (or mortgage).
    c) End the turn.
5. Exit the application. Will end turn, round and game.

2.3 Non-functional requirements

2.3.1 Usability

Usability is high priority. Normal users should be able to play the game within a very short period.

The application must communicate the state of the game in a very clear fashion. Tests with at least four different non-computer-professional should be performed to verify the usability. Test results should be part of the final documentation.

The round should be possible to internationalize to at least two different languages (example: Swedish and English).

There should be an English user manual, how to play the game.

2.3.2 Reliability

NA

2.3.3 Performance

Any actions initiated by a player should not exceed a 2 sec. response time in worst case.

2.3.4 Supportability

The application must be implemented so that the GUI is easily modifiable to suit other platforms (Web, Mobile Apps, Pads, e.t.c.). There estimated time to adapt the GUI to at web based application should not exceed 1 man-month.

The implementation should prepare for the dividing of the application into a client/server-architecture for net based games. It should be easy to partitioning the applications into a client-server architecture. A time estimation for this should be included.

2.3.5 Implementation

To achieve platform independence the application will use the Java environment. All hosts must have the JRE installed and configured. The application needs to be installed on all hosts where it will run (possibly downloaded).

2.3.6 Verification

There should be automated test verifying all use cases. Code related to the GUI could be tested manually. GUI test should be recorded and included in the final documentation.

2.3.7 Packaging and installation

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

1. A file for the application code (a standard Java jar-file).
2. All needed resources, internationalization and localization files, icons, e.t.c.

3. Start programs (scripts) to start the game on the different platforms.
4. A README-file documenting installation and start of application.

2.3.8 Legal

There could be legal issues regarding rights to the Monopoly game and trade mark. This is not covered here.

2.4 Application models

Here we present an analysis of the domain and the functionality of the application.

2.4.1 Scenarios

The use cases are relatively simple. No need for scenarios.

2.4.2 Use case model

See APPENDIX for UML diagram and textual descriptions. See also 2.2.

Use cases priority

(a longer list with motivations)

1. Move (roll dice).
2. EndTurn
3. BuyProperty
4. SellProperty
5. ...

2.4.3 Static model

See APPENDIX.

2.4.4 Dynamic model

See APPENDIX UML and use cases.

2.4.5 User interface

Application will use a fixed (non skinable, non themeable) GUI following standard conventions. The GUI must take into account different screen sizes, possible very small (minimum size: 320 x 480 (HVGA) at 163 ppi). See APPENDIX for screens and navigational path's.

2.5 Possible future directions

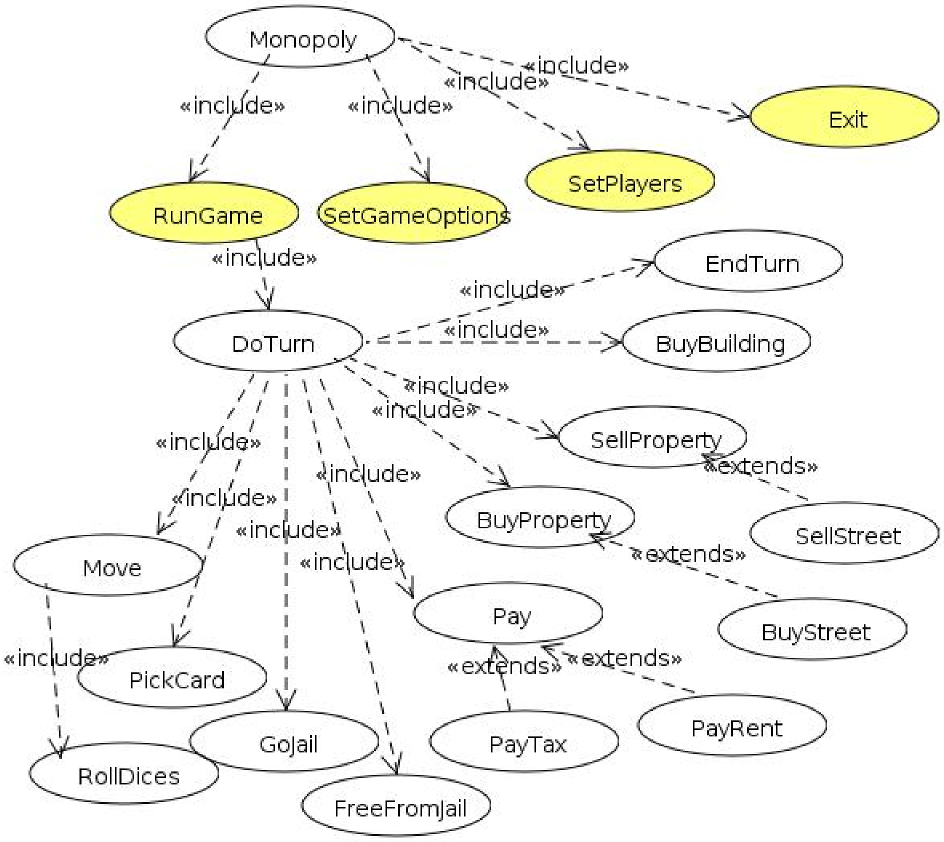
- A more visually attractive GUI. Possible 2D animations or a full 3D GUI.
- Time constraint for rounds. Different time levels (example: Round must end in 20 sec.).
- More configuration possibilities. Number of spaces other categories of spaces.
- Switchable rules.
- A computer-player (AI).
- Net based game.
- Possibilities to save interrupted games.
- Real money.

2.6 References

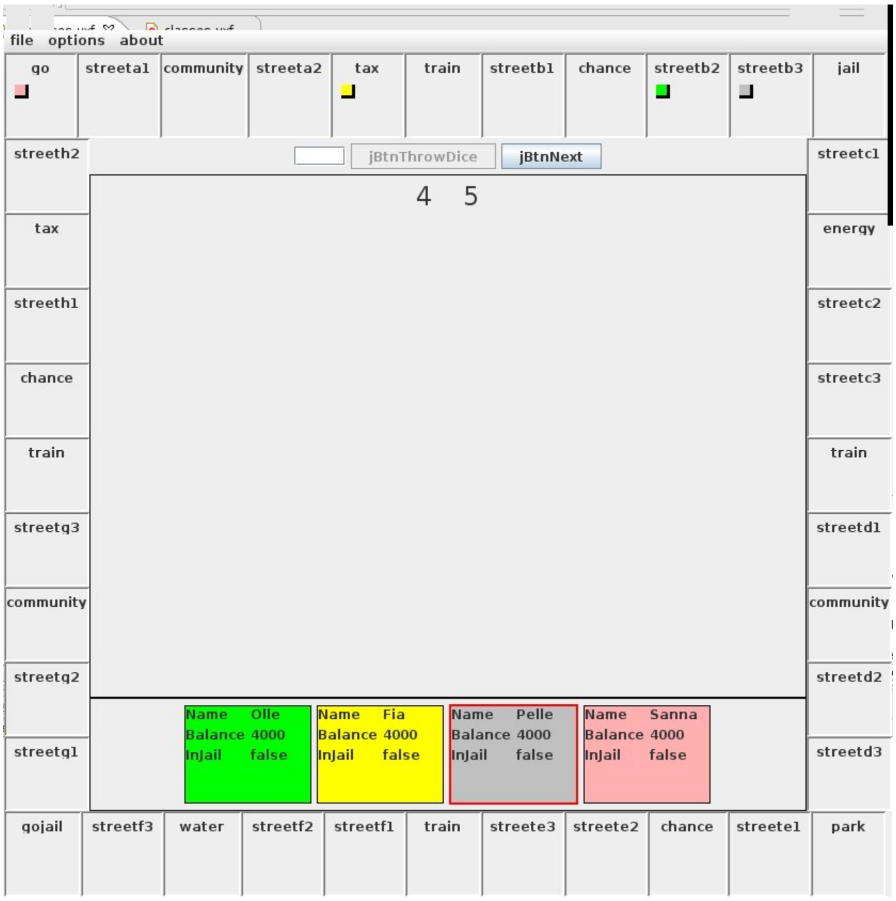
Monopoly game: http://en.wikipedia.org/wiki/Monopoly_game

APPENDIX

Use cases



GUI



Name Olle
Balance 4000
InJail false

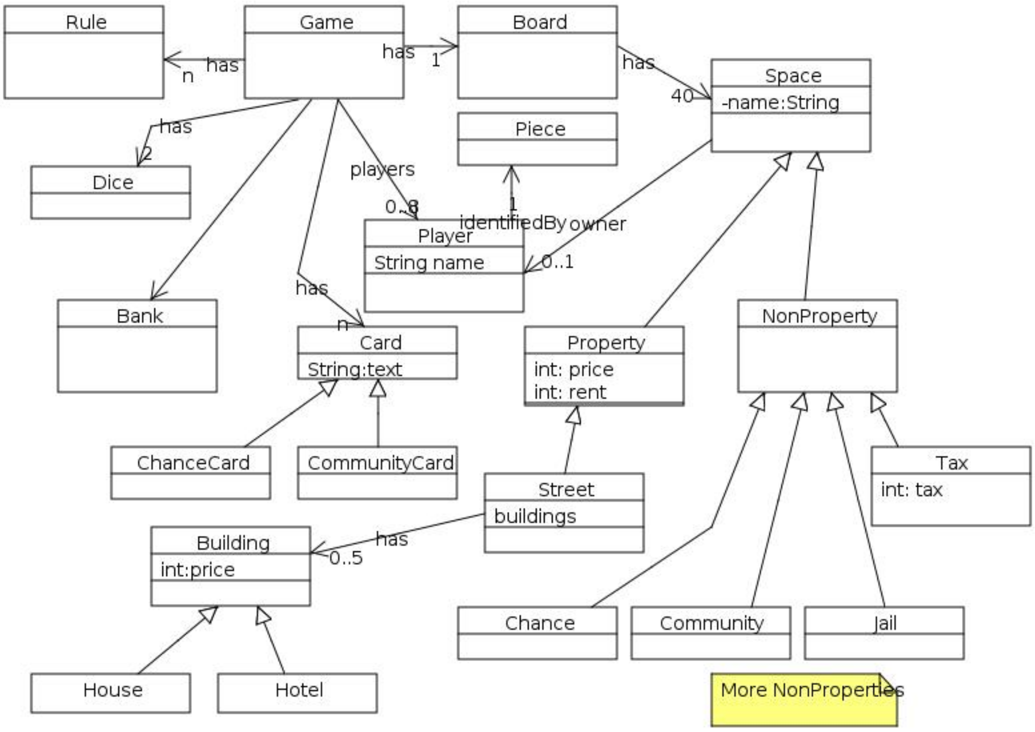
Name Fia
Balance 4000
InJail false

Name Pelle
Balance 4000
InJail false

Name Sanna
Balance 4000
InJail false

Static model

The static analysis model



Dynamic model

The dynamic analysis model
(missing, should be a sequence diagram here)

Use cases

(missing, at least 5 use cases here, use template from course page)