TDT4240 Software Architecture

Requirements Specification

Stigespillet - A Simple Norwegian Game

Group 26

Kristoffer Ravik Andresen

Vegard Bjerkli Bugge

Steinar Bækkedal

Mathieu Remaut Lund

Christoffer Evjen Ottesen

Stian Torjussen

Primary quality attribute: Modifiability

Secondary quality attributes: Usability, Testability

Chosen COTS: Android SDK, LibGDX

March 7, 2016

Spring 2016

Table of Contents

Introduction

Our assignment in this project was to create a multiplayer game. We chose the Android platform as we are more familiar with this than iOS, and we also decided to make the multiplayer functionality local to avoid network complexity. To design the game we will make use of the game-development application framework LibGDX, which has an informative wiki at its own GitHub page [LibGDX 2016]. An important requirement for the game is that it must fulfill quality requirements on modifiability specified in the requirements specification. This document discusses the requirements part of the project, and it will therefore define functional and quality requirements. It will also contain an introduction to the concept of the game we are implementing.

Concept

The game is a 2D board-game for minimum 2 players, inspired by the classical Snakes and Ladders game. Our MVP (minimum viable product) will be very similar to the original game, where all players begin at the start field and move towards the finish field based on the outcome of the dice. Whenever a player lands on a field with a ladder or a snake, they follow it to the end. The first player to reach the finish field is the winner.

For now, we have the following ideas for extensions:

- Adding a chance field (like in Monopoly) that may give the player either an advantage or a disadvantage.
- Colour coded ladders that limits which ladders a player can climb (for instance a player can be restricted to only climb red ladders).

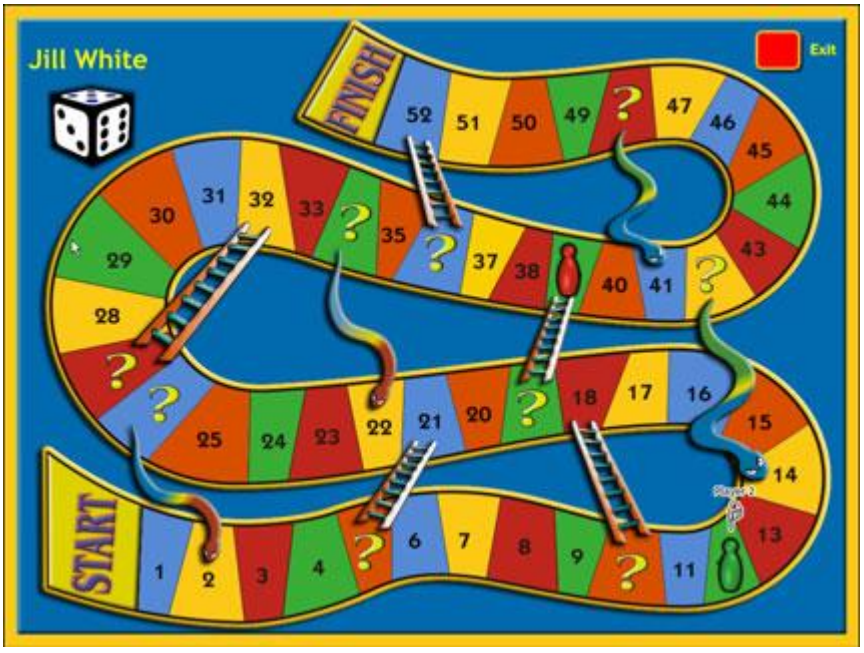

Figure 1: An example board of the game "Ladders And Snakes".

Functional Requirements

The Beginning of the Game

FR1 – A user starts the game by entering the number of participating players.

FR2 – The system registers age and nickname for all players.

FR3 – Each player is allowed to choose a token from a library.

FR4 – The youngest player in the game gets the first turn.

Playing the Game

FR5 – The player in turn rolls the dice by clicking on a dice simulator, changing its number 5 times per second.

> FR5.1 – When the player has rolled the dice, his token is moved the same number of fields that the dice shows.

FR6 – If a player land on the bottom of a ladder, he climbs up to the field where the ladder ends.

FR7 – If a player land on the top of ladder (snake), he climbs down to the field where the ladder (snake) ends.

FR8 – If a player lands on a chance field, one of the following things happens with equal probability:

- The player is teleported x fields forward.
- The player is teleported x fields backwards.
- The player can choose not to climb (up or down) the next ladder he hits (keep card).
- The player gets a new turn.
- The player moves twice the distance on the dice on the next turn.

FR9 – A player should be able to give up the game on the beginning of his turn.

The End of the Game

FR10 – The game ends when a player lands on the last field on the board (the finish field).

FR11 – The system should save the result, so the winner can retrieve it and gloat on appropriate occasions.

Note: These functional requirements are subject to change, as we take more decisions on how our Snakes & Ladders game should be different from the original game. The core rules for the snakes and ladders game are retrieved from [Snakes 2016].

Quality Requirements

Table 1: List of quality attribute scenarios. Mad with a little help from [BCK 2012, fig. 42, p. 70].

QA	Source	Artifact	Stimulus	Environment	Response	Response Measure
U1: Usability	User	System	Plays the game with friends	Runtime	Understands the rules, and is able to explain how the game works	Within the user's first playthrough
U2: Usability	User	System	Plays the game with friends	Runtime	User understands what he is supposed to do next when it's his turn	Within 1 minute
M1: Modifiability	Developer	Code	Wishes to add a new level	Design Time	New level created	Only one controller file and one view file created
M2: Modifiability	Developer	Code	Wished to change a game rule	Design Time	New rule created and Unit Tested	In 2 hours
T1: Testability	Unit Tester	Code Unit	Code Unit Completed	Development	Results Captured	100% Function Coverage
T2: Testability	Unit Tester	Code Unit	Code Unit Completed	Development	Results Captured	90% Statement Coverage
T3: Testability	Unit Tester	Code Unit	Code Unit Completed	Development	Results Captured	80% Branch Coverage
T4: Testability	Unit Tester	Code Unit	Code Unit Completed	Development	Results Captured	70% Condition Coverage

On Chosen COTS Components

After careful considerations (and googling) the group has determined that neither Android SDK nor LibGDX [LibGDX 2016] will hinder our architecture implementation. An issue related to libGDX, however, is found in the scene graph package Scene2D [LibGDX 2016]. When using Scene2D to build not only a UI, but an entire application, its Actor classes couple view logic (draw() - methods) with logic related to editing the model (act() - methods). We will resolve this issue by providing our own way of interacting with models, and we will make the scene graph interact with our own Controller intermediaries. In fact, we may as well consider utilizing LibGDX for the View Layer only. We may revise this section as the implementation processes progress and further increments are added to the system.

Issues

The group spent a total of 20 man hours to determine both functional requirements, architectural constraints, architectural drivers (but not their licenses) and Quality Attribute Scenarios. Most of the production time was in fact put into the ADD. The group members are familiar with basic requirements engineering, which contributed to an efficient writing process of the requirements. We are however aware that we possibly need to make edits on this document as much as we are with the architecture specification.

We have also yet to decide how other implementation choices constrain the software system itself (other than the fact that it cannot run on anything else than Android - devices). And we haven't even decided what versions of the operating system we are going – We may want to go for at least 90 per cent coverage in the Android, though, which means using the version 15 or 16 of the API the SDK delivers. Urgghh…

See also the same section in the Architetural Description Document for organization related issues.

References

- [BCK 2012]: Len Bass, Paul Clements, Rick Kazman. "Software Architecture in Practice – Third edition". Addison Wesley. September 2012.
- [LibGDX 2016]: LibGDX Wiki. https://github.com/libgdx/libgdx/wiki. Accessed 6.3.2016.
- [Snakes 2016]: Wikihow – Play Snakes And Ladders. Available at http://www.wikihow.com/Play-Snakes-and-Ladders. Retrieved 01/03-2015.