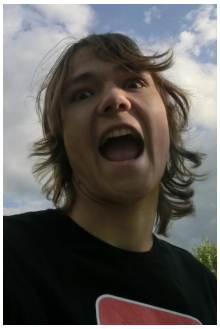
Contextproject Programming Life Emergent Architecture Design

TU Delft



Ruben Bes
rbes
4227492

Mathijs Hoogland
mhhoogland
4237676

Jasper Denkers
jdenkers
4212584

Robbert van Staveren
rhvanstaveren
1527118

Willem Jan Glerum
wglerum
4141040

May 16, 2014

Abstract

Product planning gives a great overview of the features that the application will have. Mainly following the MoSCoW method with Must, Should, Could, Would haves. These will all be backed by their relevant user stories. Also we will give the definition of when a feature is really done.

Contents

1 Introduction

The product planning contains all the important information of the product, the product backlog, roadmap and user stories

2 Product

2.1 High-level product backlog

This section describes the desired features of the application according to the MoSCoW method [1]. Categorized into four groups:

- **Must haves** describe requirements that must be statisfied in the final solution
- **Should haves** describe high-priority requirement that should be included if possible
- **Could haves** describe requirements that are considers desirable but not necessary.
- **Would haves** describe requirement that stakeholders have agreed will not be implemented in the solution, but could be added in the future.

2.1.1 Must haves

- Visualize triodata of father, mother and child
- Looking for known disease mutations
- Retrieving data from existing genetic databases
- Reading VCF (Variant Call Format) files
- Easy to use GUI for doctors.

2.1.2 Should haves

- Uploading VCF files to the server in the background

2.1.3 Could haves

- Exporting visualization data

2.1.4 Would haves

- Spread computational power over multiple threads, cores, or systems.
- Support for mobile web browsers.

2.2 Roadmap

Major release schedule, release goals.

3 Product backlog

First version with estimates and prioritized user stories.

3.1 User stories of features

A child has an illness, while the parents do not. A doctor wants to use our product to find out wether a mutation of a particular gene may be the cause of this disease. For this, he has the following method:

The doctor chooses a VCF file of the parents and the child to analyze. He gets a visualization where he can see what mutations there are in the DNA of the child and whether the locations of these mutations have been associated with the illness of the child. The doctor uses the visualization to deduce what changes may be the cause of the disease.

3.2 Initial release plan

The general roadmap of our product will be as below:

sprint 1

Basic setup of framework, read VCF files and connect to database.

sprint 2

Make usefull query's for the database and find all mutations in the VCF file.

sprint 3

Retrieve data from the database assosieted with the mutations found in the VCF file and make a visualisation of the mutations.

sprint 4

Visualize the found mutations and possible connections with genes associeted with a dissease.

sprint 5

Improve usability of the product.

sprint 6

Improve visualisations.

sprint 7

Finish report and presentation.

4 Definition of Done

This section describes when a feature is really done and ready to be integrated in the system. We will define this for a feature, sprint and end product.

A feature is finished when it is fully tested and the code is accepted by other developers. These test should done with JUnit and the code coverage should be high enough. Furthermore the code must be fully documented with JavaDoc and should match the rules from CheckStyle.

A sprint is finished when the whole application is tested and approved, just like a feature. However the continuous integration system should also accept the build. Next the developers and users will test the system by hand to check for bugs.

The end product is finished if all the Should haves and Could haves are implemented and tested as described above, because only the should haves give a very basic application. The product should have the looks and feels approved by the stakeholders, they should be happy with the product.

Furthermore in addition to all this the code should be well documented, tested, style checked and integrated. This will all be evaluated by the SIG (Software Inprovement Group) and should be improved after the first check.

5 Glossary

VCF - Variant Call Format

References

[1] Kevin Brennan et al. *A Guide to the Business Analysis Body of Knowledger.* Iiba, 2009.