Product Planning

Tycho van Heems - tvheems - 4464567
Dex van Leeuwen - dqvanleeuwen - 4475461
Martijn Straatman - martijnstraatm - 4442504
Jochem Tiessen - jtiessen - 4478339
Michael Tran - michaeltran - 4499638

May 4, 2017

Contents

1 Introduction **3**

2 Product **4**
- 2.1 High-level Product Backlog 4
- 2.2 Roadmap 4

3 Product Backlog **5**
- 3.1 User Stories of Features 5
- 3.2 User Stories of Technical Improvements 5
- 3.3 User Stories of Know-How Acquisition 6
- 3.4 Initial Release Plan 6

4 Definition of Done **7**
- 4.1 Features 7
- 4.2 Sprints 7
- 4.3 Milestones 7
- 4.4 Final Product 7

1 Introduction

DNA analysis is a growing field in the science due to increased possibilities of researching DNA and the amount of information that can be extracted from DNA. DNA analysis is used to check whether people are related, to find certain people and recently research is moving towards predicting traits of people by studying their DNA.
Every living entity on this earth has their own DNA. This DNA consists of four different bases: Adenine, Cytosine, Guanine and Thymine. The sequencing of these different bases is what determines the traits of living entities. By studying these sequences, researchers are able to predict some personal traits, like whether a person might get a certain disease or not or how tall a person is going to be. With more research, more traits will become predictable, which can improve the health care since possible diseases can be detected before these are even present in the body using this DNA analysis.
The problem with analyzing DNA is that DNA contains so much information that it is hard to efficiently view all the information contained in the DNA, making it easy to overlook small details, which may be important for the research. Because of this, we will make a genome browser, which will make it possible to layout the sequence of those four bases efficiently, so researchers have a clear overview of the DNA. The program will also include functions to simplify the research, like aligning multiple different DNA sequences.
In this product plan, it will be explained how the product is planned to be made by us. This will be done by looking at the features and functionalities that are needed in the product, which will be done by making a product backlog(reference naar een site die uitlegt wat het is). Using this product backlog, an initial release will be made, which will specify what is expected to be done at certain moments. This release plan will be constructed by using so-called milestones, where each milestone represents a major functionality or a series of smaller features.

2 Product

In this section, a high-level product backlog and a roadmap will be created to show the needed functionalities and the expectations of the process during the development. The product backlog focuses more around the features that need to be made and the prioritization of those, while the roadmap focuses more around what is expected to be done at what moment.

2.1 High-level Product Backlog

In the high-level product backlog, different aspects of the product that need to be created are put in a clear overview. Not all aspects will be put in this backlog, but only the most important features. The others will be covered in Chapter 3. These will be sorted in three different categories: epics, semi-clear and clear. The epics category contain the tasks that will most likely be the most amount of work, the semi-clear will take an average amount of work and the clear category will contain the tasks that will take the least amount of work. The following table shows the current product backlog.

Table 1: High-level Product Backlog

Epics	Semi-clear	Clear
Create an efficient data structure	.gfa file parsing	File choosing
Make the layout of the graphs efficient	Create subgraphs when a center node is chosen	Make it possible to view data of selected parts of the graph in a console
Intuitive navigation around the graph	Semantic zooming	Visually encode data in the graph
	Create a possibility to highlight data in the graph	

2.2 Roadmap

The roadmap shows which features are planned to be finished at what moment. For this, milestones are created which specify the minimum functionalities it should have at that point. There could be more features, but there should not be less than specified in this plan. There should be a runnable version of the product at each of the dates.

Table 2: Roadmap

Milestone	Date	Minimum Functionalities
1	12-05-2017	File parsing, basic GUI and basic visualization of graphs
2	19-05-2017	Visualize subgraphs by using centernodes with some layout
3	02-06-2017	Efficient layout of graphs and navigation in the graph
4	09-06-2017	Use visuals to encode information(by using colors for example)
5	16-06-2017	Path highlighting, bookmarks and text search on extra fields
6	23-06-2017	Semantic zooming
7	30-06-2017	Extra functionalities as eye-candy

3 Product Backlog

In this section, the product backlog will be fully explained, by specifying what each feature in the backlog exactly is and by making user stories. User stories are a way of determining what features have to be in the product by imagining what the users want to do with the application. A user story is usually a small story where a a part of the program is used. Each user story should cover one feature. we created user stories for loose features, possible defects that should be prevented, possible improvements and know-how acquisition.

3.1 User Stories of Features

The features cover the functionalities that the program should contain, the following user stories show how we imagine that these features are used in the program.

- The user wants to be able to parse a .gfa file, so a graph can be made of it.
- The user wants to be able to choose a .gfa file himself.
- The user wants to be able to view and select the data in nodes/edges of the graph.
- The user wants to be able to make a subgraph after choosing a center node on the graph, which only contains information of around the center node.
- The user wants the graph to be easy to understand without overlapping elements.
- The user wants to see basic information in the graph without having to extensively check the specific nodes by looking at the colors of nodes for example.
- The user wants to navigate around the graph in an intuitive way, which is not confusing.
- The user wants to dynamically change a subgraph that is viewed at the moment, by changing the center node or the amount of nodes around the center node.
- The user wants to be able the highlight and bookmark data of choice in a graph.

3.2 User Stories of Technical Improvements

The technical improvements cover the aspects of the product that ensure that the number of bugs and defects is kept as low as possible, while also covering non-functional aspects of the product, like performance.

- The user doesn't want to have to wait very long while performing actions on the graph.
- The user wants to be sure the product is of quality and doesn't contain bugs or defects.
- The user wants to be able to use this product on Windows, Apple and Linux operating systems.

3.3 User Stories of Know-How Acquisition

Know-how acquisition is the act of acquiring information about how to use the product. User stories regarding this revolve about in what way it is clear to the user how to handle the product.

- The user wants to be able to read a document containing all specific information on how to use the product.
- The user doesn't want to be confused by any of the text used in the program.
- If the user cannot figure out how to fix a problem by reading the document, the user should be able to contact the developers for help.

3.4 Initial Release Plan

The initial release plan describes how we imagine the product to be released. It will cover releases that are not final and the final release.
In the roadmap, see Chapter 2, different milestones are specified which cover what features should be done at which moments. At each of those moments, it is expected to bring out an executable product and demonstrate it in a meeting in front of the customer. The customer can give his viewpoints on the current state of the project and he can ensure that the product is moving in the right direction. This means that at every milestone, a non final version of the product will be released.
The last milestone will cover the final release of the product, which should contain all features in the product backlog including some other bonus features if possible. It should also make sure the product is of a certain quality. During the final meeting, the customer will tell the development team whether he is content with the final product.

4 Definition of Done

In the definition of done, it is defined when a part of the project is done according our standards. A definition is created for features, sprints, milestones and the final product. All these definitions of done specified here, are made according to the definition of done of the development team for this product.

4.1 Features

A feature is considered when its functionalities are completely working and if the features is tested extensively. For testing, automated tests are written to guarantee the quality of the final product. For features, it will mostly be unit testing, where we consider a feature tested properly when at least 70% of the branches in its code are tested and when every test succeeds.

4.2 Sprints

A sprint will be considered done when all the user stories specified in the corresponding are either finished or when there is a legit reason for it not to be finished, e.g the specified user story is not applicable to the product anymore. Also, the product needs to have a runnable version at the end of each sprint, which should be tested extensively. A boundary of 80% branch coverage over all the code was chosen for the code to be considered tested well enough.

4.3 Milestones

Every milestone which is specified earlier will be considered done when all the functionalities specified for the milestone are included in the product. Since these functionalities are seen as features, a testing boundary of 70% branch coverage was chosen for the functionalities in the boundaries.

4.4 Final Product

The final product will be considered done when a number of things are achieved.
First, the final product should be able to compile and run on Windows, Apple and Linux.
Second, all features in the must-haves have to be in the final product, because when these are not present, the product will not have one of its core functionalities. The specified should-haves should be included as much as possible. However, in contrast to the must-haves, it is not as strict, which means that not every should-have has to be made, although it is the preferred result to include all of them.
Third, the customer should be content with the final result of the product. If the customer is not content, the production has been for nothing, so the customer has to be content.
Last, the whole product has to be tested extensively to guarantee the quality of the product and to reduce the number of bugs. As in the sprint, the product is considered to be well tested if it 80% of its branches are covered with the automated tests.

References

"The Scrum Product Backlog - International Scrum Institute". Scrum-institute.org. N.p., 2017. Web. 4 May 2017.

"Agile Development Release Planning". VersionOne. N.p., 2017. Web. 4 May 2017.

"Agile Roadmaps: Build, Share, Use, Evolve The Agile Coach". Atlassian. N.p., 2017. Web. 4 May 2017.