Product Planning

Programming Life
Team Dynamites & Butterflies

Jasper van Tilburg - 4394554
Marc Visser - 4318234
Lex Boleij - 4153197
Eric Dammeijer - 4412265
Jip Rietveld - 4486528

18-5-2017
TU Delft

Table of Contents

Introduction

The following is a report on how Dynamite & Butterflies will create an interactive multiple genome graph browser. This browser will contain multiple different features such as: dynamic view updates, semantic zooming, adding annotations etc. A product created by five engineering students of the TU Delft. In said document the product and its features will be specified, a release schedule and roadmap of said features will be made, as well as the user stories that come with those features.

Product

In the following section is an explanation of what features the product will contain. It involves the product backlog, a section dedicated to the features of the product. Besides that this section involves a roadmap as a general planning for the project and how the product development will happen. Finally we have a MoSCoW method discussing what features have priority over others.

High-level product backlog

This section contains the high-level requirements for the product.

- The application is able to correctly parse and save data from gfa files
- An interactive user interface allows the user to interact with the data presented
- The data should be visualized and presented to the user in a clear way
- The user should be able to pan and zoom through the data
- The application should keep track of the current position in the graph
- The application should make use of semantic zooming
- The user should be able to choose subsets of the genome to display
- The user should be able to annotate the presented graph
- The user should be able to hide certain information he deems unnecessary
- The user should be able to save file specific bookmarks
- The user should be able to search through the genome
- The user should be able to highlight specific paths present in the genome
- The graph should use colours and thickness to make the data more clear
- The application should be able to recognise mutations in the genome.
- All of the above should scale into very large datasets

Roadmap

The product will be developed with Agile programming using the SCRUM methodology. In sprints of one week working products and releases will be presented. Together with a release comes a sprint retrospective, reviewing the previous weeks work. And a sprint backlog, creating a plan for the coming week to reach the next milestone.

Date	**Goals**
12-05	- Setup project with tools - Parse files - Basic UI - Some visualization
19-05	- Parse and load large data sets - Visualize graph based on centre node and radius - Move centre point query
02-06	- Layout the graph with no overlapping edges - Enable panning of graph - Enable zooming of graph
09-06	- Search/filter - Conditional colouring of nodes and edges - Edge thickness of weight attribute - Edge colour highlights specific samples
16-06	- Highlighting paths - Creating bookmarks in the graph - Highlight subsections of the graph
23-06	- Semantic zooming - Stacking of multiple overlapping annotation paths
28-06	- Eye-candy - Bug fixes - Codebase is clean

MoSCoW method

The following section will break down our product into the MoSCoW method. On advice of our client we decided to set this up. In the MoSCoW method we explain what features we feel our application must have. What features it should have, what features it could have. And eventually, at the end of the cycle, what features it won't have.

This application must:

- Enable us to interactively explore a sequence graph representing the genome architecture of multiple strains
- Provide a scaleable layout of the graph representing multiple genomes
- Position nodes automatically with an efficient graph layout algorithm
- Identify mutations in the graph in the context of well-known reference genomes
- Provide semantic zooming to enable useful visual interpretation at various zoom levels from whole-genome to individual mutations
- Load graph data format gfa
- Define the relationship between the length of a node and the length of its sequence
- Be able to highlight different paths in the genome
- Contain conditional colouring of nodes

This application should:

- Provide useful quality-of-life features to share, bookmark and annotate graphs
- Identify mutations and determine the type of variant (insertion, deletion, SNP) uniformly across the samples
- Have visual encodings for different classes of mutations and the ability to filter on mutation class

This application could:

- Reposition and reshape nodes by clicking and dragging with the mouse.
- Implement different features with intuitive keyboard shortcuts

This application won't:

Product backlog

In this section we will discuss the elements that should be implemented during the course of the project.

User stories of features

As a user,
When I start the application,
And I have not yet done anything,
I should be able to choose a gfa file to load into the application.

As a user,
When I load a gfa file into the application,
I should be able to choose a centre node and radius to be displayed.

As a user,
When I am viewing the graph,
I should be able to add annotations to the graph.

As a user,
When I am viewing the graph,
I should be able to pan and zoom through the graph.

As a user,
When I am zooming in or out of the graph,
The application should make use of semantic zooming.

As a user,
When I am viewing the graph,
I should be able to gain more information on nodes and edges by clicking them.

As a user,
When I am interacting with the application,
I want the application to feel intuitive.

User stories of technical improvements

As a programmer,
When reading the code,
I want clear code in which I can quickly identify the functionality of different methods and variables.

As a programmer,
When I want to contribute to the project,
I want to be able to clone the repository and create a pull request easily.

As a programmer,
When I want to run the application,
And after cloning the master branch,
I want it to build without failures.

User stories of know-how acquisition

As a development team,
When we make a decision in a field,
And we are not too knowledgeable in that field,
We will read literature to ensure we make a knowledgeable decision.

As a designer,
When designing the user interface,
I design it so to ensure it is intuitive.

Definition of Done

This section describes when something is done. This is to ensure all project members are on the same line when rounding out certain aspects of this project. We will define finished for the following three subjects.

- Features
- Releases
- Sprints

Feature

A feature is considered done when:

- The feature is reviewed by two group members who did not work on the feature
- The feature is thoroughly tested and has a test coverage of >80%
- The code is of good quality and it is readable by the entire team (and outsiders). We ensure this by use of analysis tools such as Checkstyle and comments in the code
- The feature meets the requirements set in the user story

Releases

A release is considered done when:

- It successfully builds on Travis
- The application can be started on the following operating systems:
    - Linux Ubuntu 16.04
    - Windows 10
    - macOS 10.12

Sprint

A sprint is considered done when:

- A sprint backlog has been made for coming sprint
- A sprint retrospective has been made reviewing the sprint and its accomplishments
- Most (if not all) open pull requests have been reviewed and merged into the master branch

Glossary

Feature
A feature is a certain functionality or service that can be added to the software.

Genome
A genome is an organism's complete set of DNA, including all of its genes.

MoSCoW method
A method used to determine priorities of a project. It defines must haves, should haves, could haves and eventually, won't haves. By splitting different features into these different sections the development team can know what to prioritise on.

Scrum
Scrum is a framework for project management that emphasizes teamwork, accountability and iterative progress toward a well-defined goal.

Sprint
A sprint is part of the Scrum methodology. A sprint is the time frame (between 1 week and 2 months) the development team has to bring out a new release of the software they are building. In this project a sprint is 1 week.