Product Planning

Health Informatics 3

May 6, 2015

1 Introduction

TODO

2 Product

TODO an introduction about his section

2.1 High-level product backlog

In this section we describe the product backlog according the MoSCoW method. Therefor we divide the features into four group. The features are divided based on their priority. Section 3 describes for some features a user story.

2.1.1 Must Haves

These features are essential for the product. Without these features the product is not usable.

- Language in which the user can describe different analysis
- Executing an analysis that is defined in a file.
- Load data in the program based on a description file.
- Indicating the data connections between the different datafiles
- Indicating the meaning of the various data inputs
- Load data from different sources using the one data description file
- The 8 C's for data analysis
  - Chunk analysis
  - Comments
  - Codes
  - Connections
  - Comparisons
  - Constraints

  - Conversions
  - Computations
- Specifying the output and output format
- Visualizations from the data analyzed
  - Frequency bars
  - Line graph
- Manual for the analysis description language

2.1.2 Should Haves

These features are very useful. However without these features the product is still usable.

- Visualizations
  - Box plot
  - Stem leave
  - State transition matrix
  - Lag analysis
- Exporting the visualizations to images
- Implement some example analysis in our analysis description language

2.1.3 Could Haves

These features will only be done when there is enough time.

- Visualizations
  - Histogram
  - Markov chain graph transition diagram dingen
- Editor for inputting the analysis description
- Mass input for batch processing
- Preview of the output from the analysis

2.1.4 Would Haves

These features will not be implemented during this project. If this project is followed up by another project, these feature might be interesting.

- Easy to use GUI for specifying the analysis

2.2 Roadmap

This section will describe the planning for the product. In this roadmap we will plan the mayor releases. For a detailed overview of the task for each week see section 3.1. The numbers of the week correspond to the week of quarter 4. A week ends on Friday.

Week	goals
1	Setup project
2	Minimal design of the user interface and basic architecture of the product
3	Product vision and the program must be able to read and write data
4	Product planning and it must be possible to perform half of the data analyses
5	The user can perform most of data analyses
6	The user can perform all the types of data analyses and the most important visualizations
7	The program is able to show the almost all the planned visualizations
8	The program is able to show all planned visualizations
9	Final product
10	Final report and presentation

3 Product backlog

user stories according the priorities

3.1 Initial release plan

This section will describe the planning for the product. The release plan is based on sprints of one week. The numbers of the week correspond to the week of quarter 4. A new iteration starts on every Friday. For each week we will list which features the product should have and which additional task must be done.

3.1.1 Week 4.1

- Setup the software that is used during the project.
- Obtain the requirements.

3.1.2 Week 4.2

- A basic architecture for the product.
- A design for the user interface.
- A draft version of the product vision.

3.1.3 Week 4.3

- A minimal user interface according the design of week 4.2.
- The final version of the product vision.
- A draft version of the product planning.

- The user must be able to specify in a data description file how a file should be read by the program.
- The user must be able to specify which data must be written to a file.
- The user must be able to perform constraint analyses.

3.1.4 Week 4.4

- The final version of the product planning.
- The user must be able the perform chunking analyses.
- The user must be able the perform connections analyses.
- The user must be able the perform computation analyses.

3.1.5 Week 4.5

- The user must be able the perform codes analyses.
- The user must be able the perform comparisons analyses.
- It must be possible to show the data as frequency bars.
- It must be possible to show the data as a line graph.

3.1.6 Week 4.6

- The user must be able the perform comments analyses.
- The user must be able the perform conversions analyses.
- It must be possible to show the data as a box plot.
- It must be possible to show the data as a Stem-and-Leaf plot.
- Input for SIG.

3.1.7 Week 4.7

- It must be possible to show the data as a state transition matrix.
- Show the data with Lag analysis
- The user must be able to export the visualizations as an image.
- It must be possible to show the data as a Histogram.

3.1.8 Week 4.8

- It must be possible to show the data as a Markov chain.
- Specify multiple files that all will be analyzed individually.
- Implement certain analyses functions in our language.

3.1.9 Week 4.9

This is the last week where it is possible to work on the code. This week will add no new features. In this way we will be able to handle some delay during the process. Furthermore this week is used to repair the last bugs. Therefore there is a feature freeze on Wednesday June 17.

- Final input for SIG
- Draft version of the final report

3.1.10 Week 4.10

- Final report
- Product presentation

4 Definition of Done

In this section we will discuss when a task is considered as done. In general a task is done when there is nothing left to do for that task. We will discuss the definition of done for backlog items, sprints and releases.

4.1 Backlog Item

A backlog item is done if it is implemented as described and it follows the description of the user stories. Furthermore the code should have been tested with unit tests. All the other features should still work and all the test should pass. The code must be reviewed by at least two persons who has not worked on that specific item. The code should be clear and when needed, it should contain comments. Furthermore the code should follow the languages conventions and it should have clear names for the variables. When the item meets all these requirements, than it is considered done.

4.2 Sprint

Each sprint should have a sprint plan and a sprint reflection. Any deliverable that has a due in or at the end of the sprint should have been made and hand in. Furthermore if needed, relevant documents, such as the architecture design, should have been updated. Critical bugs and errors that are discovered during the sprint should be fixed. If it is not possible to fix them during the sprint, than they have to be solved in the next sprint. Finally all the task of the sprint should be completed as described in the previous section. If it is not possible to complete a certain task in a sprint, than the sprint reflection should explain why it is was not possible to finish the task.

4.3 Release

Each sprint ends with a new version of the product. Sections 2.2 and 3.5 give an overview of the planned features for each release. Based on that, each sprint will ad some new features to the product. A release is only allowed to contain

features that are considered done, see section 3.1 Therefore all the features in a release are tested and the code should be proper. Additionally we must test whether the features work correctly together. Furthermore a release may not have any critical bug or error. Finally for each release a demo has to be prepared and demonstrated.