TI2806: Context Project

Tools for Software Engineering

Product Vision

Octopeer Analytics

Group: BussInfraManDevOps

Borek Beker (4118650)
Marco Boom (4393031)
Leendert van Doorn (4373286)
Ahmet Gudek (4307445)
Daan van der Valk (4094751)

Supervisors

Context coordinator	Context T.A.	Software Engineering T.A.
Alberto Bachelli	*Aaron Ang*	*Bastiaan Reijm*

May 4, 2016

Contents

1 Introduction

This document describes the product vision of *Octopeer Analytics*, part of the *Octopeer* project. In this section, some important concepts regarding software peer reviews are introduced. Also, the Octopeer project is introduced and sets up the following sections.

Last couple of years, online services for version control and continuous integration of software, like GitHub and Bitbucket, have become popular tools to make software development easier. A cloud based codebase made it significantly easier for (large) teams to work on the same project, both commercially and non-profit. For both public and private projects, contributors suggest a change by creating a *pull request*. These changes are then inspected by the responsible project members or other contributors in so called *peer reviews*. During a peer review, the suggested changes are examined, possibly commented on, and approved or rejected by the reviewer. This concept is used for code reviews since a couple of decades ago. (Ackerman, Fowler, & Ebenau, 1984)

Although research has shown that the functionality of continuous integration is not always used in practise (Vasilescu, van Schuylenburg, Wulm, Serebrenik, & van den Brand, 2015), the popularity of the such platforms can hardly be disputed. GitHub announced it's 10 millionth repository at the end of 2013. (Doll, 2013) In both GitHub and Bitbucket, managing and reviewing pull requests are important parts of the development work flow. According to Sripada, Raghu Reddy, and Sureka, 2015, using peer code reviews at an undergraduate level Software Engineering course "has been positive in terms of [improving] code quality".

The *Octopeer* project aims to collect data about these peer review sessions through a browser extension. The goal of **Octopeer Analytics** is to present data collected by the Octopeer extension in the most clear and useful way. It is a web application, accessible via the browser extension, and can be adjusted to the users needs. It should give insights in the reviews of individuals, as well as reviews on project or company scale.

2 Target audience

As part of the Octopeer project, the Octopeer Analytics mainly targets Software Engineers, Testers, Managers, Computer Science students, and others involved in software engineering where code reviews are a part of the work cycle. It is essential to the popularity and effectiveness of the project, to become part of as many software projects as possible. This will be achieved by making it a handy tool for both code reviewers and managers. We do not aim for

researchers, making the assumption that they will prefer to study the dataset of the Octopeer project itself. The following personas could all be potential users of the Octopeer Analytics, having different backgrounds and goals.

2.1 Personas

2.1.1 Lars

Lars (48) is a project manager of a software development company. The teams he is involved in produce a lot of code, aiming to produce a working deliverable every two weeks. To recognize and eliminate as much bugs as possible, Lars' company intensively uses peer reviews before branches are merged with the master. Lars would like to make this peer review process as efficient as possible, by measuring the code reviews of his team members. He aims to select his best code reviewers, in terms of speed and number of found bugs, to create a comprehensive protocol for peer reviews, getting the best out of his employees. Lars decides to use Octopeer, hosted on a company server, to use with his teams.

2.1.2 Lisa

Lisa (19) is a first year Computer Science student. Because she is single, boys are giving her always a lot of attention. She has just learned to program in Java, and wants to contribute to Open Source projects on Bitbucket for a study project. The project supervisor demands her to install the Octopeer plugin for Chrome to monitor her code reviews. Lisa thinks it can help her. Because her team mates always complimenting her work to much she never receives negative feedback. Therefore it is hard for her to improve her peer review skills. She hopes that she can use Octopeer Analytics to compare her skills with other team members. Also she have better insights when her review skills are graded.

2.1.3 Eric

Eric (37) is a software developer at an Open Source IT company. He works in a multidisciplinary team in which Eric both commits and reviews code. After 12 years of experience, he knows peer reviews are very important to improve the code quality. Eric would like to get insights on his code review sessions, aiming to do work faster, without missing any problematic parts. He installs the Octopeer browser plugin to get statistics helping him assess his GitHub peer review sessions, taking a look at the data using Octopeer Analytics.

3 Customer needs

The main problem of code reviewing at the current moment is that there is no kind of validation of the quality of code reviewing. For example: even if a pull request is created, it is easy to merge a branch directly without looking one second to the code. If there is no integrated test service, it is also unknown whether the code works correctly or contains bugs. Therefore a little help may be useful. The idea of Octopeer Analytics is that you have insight in how developers deal with pull requests. There are a few different perspectives on the Octopeer Analytics product:

- Generally, software reviewers will mostly be interested in their own statistics, comparing their own development and changes over time, and might want to compare this to team/project averages;
- Project managers may want to see statistics on reviews of their project(s). They can monitor the progress of a project and the performance of a team in terms of quality. For example: when needed, they can adjust developers who perform worse in peer reviewing than other team members.

4 Crucial product attributes

4.1 High-level product backlog

- The product will need an accessible page.
- The page needs to connect to a back-end with the user's data.
- The data will need to be requested from this back-end.
- The received data will need to be mined for useful data.
- The useful data needs to be visualized for the user.
- The visualizations should to be divided into multiple categories.
- The user should be able to only view the desired data category.

4.2 Non-functional requirements

The stated customer needs imply that there are several distinct pages required, corresponding to the scale the user is interested in. The most important product attributes are:

- Usefulness: by presenting useful data, the users will get practical and statistical insight in the peer review process. The Octopeer Analytics not only be fun to play with, but provide helpful analytic tools.
- Performance: as there is a lot of data being generated by the Octopeer extension, Octopeer Analytics should generate graphs and overviews fast.
- Adaptability: different users will focus on different elements of the data. This might not only vary from scale, but also to focus on specific parts of the data: mouse movements, coverage of all changes, comments and so on.

5 Comparison to existing products

Several tools are available to assist peer code reviews, in particular using static code analysis. These tools usually aim to help the reviewer by providing information about the pull request itself. According to Mantere, Uusitalo, and Roning, 2009, who compared three static code analysis tools, "using up-to-date static analysis tools can be recommended for any serious software project". Sprunck, n.d. recommends to use some different tools simultaneously, as they "give just in the combination 100% functionality you may need in your project". Code visualization can be a useful addition to the reviewer. Some concepts of visualized code reviewing, in particular concerning the reviewer-visualization tool relation, are patented by Microsoft. (Wang, Tang, Xuan, & Damata, 2015). Octopeer Analytics will not compete with these tools, as it has a completely different target (namely to provide information about the *reviews*).

Another field of competition may arise regarding the problem of finding the most appropriate peer reviewer(s) for a pull request. Several methods have been developed to achieve this. According to Zanjani, Kagdi, and Bird, 2015, their developed tool *cHRev* outperforms other existing tools, by combining factors like recency of other reviews, number of comments, addition to the file, etc. Although this is not the main focus of Octopeer Analytics, it may include a tool for this purpose. To be competitive, Octopeer Analytics should not (only) aim to rank the peer reviewers, but provide insight in this ranking process.

6 Project time span

This project takes 10 weeks, and will contain 8 one-week sprints with a working deliverable. The project will be completed on June 17, 2016, and the final report will be handed in on June 23, 2016. After this, Octopeer Analytics should be merged with the Octopeer project.

References

Ackerman, A., Fowler, P., & Ebenau, R. (1984). Software inspections and the industrial production of software. In *Proc. of a symposium on software validation: inspection-testing-verification-alternatives* (pp. 13–40). Darmstadt, Germany: Elsevier North-Holland, Inc.

Doll, B. (2013, December). 10 million repositories. Retrieved from https://github.com/blog/1724-10-million-repositories.

Mantere, M., Uusitalo, I., & Roning, J. (2009, June). Comparison of static code analysis tools. In *2009 third international conference on emerging security information, systems and technologies* (pp. 15–22). doi:10.1109/SECURWARE.2009.10

Sprunck, M. (n.d.). Comparison of static code analysis tools for java - findbugs vs pmd vs checkstyle. Retrieved from http://www.sw-engineering-candies.com/blog-1/comparison-of-findbugs-pmd-and-checkstyle.

Sripada, S., Raghu Reddy, Y., & Sureka, A. (2015). In support of peer code review and inspection in an undergraduate software engineering course. doi:10.1109/CSEET.2015.8

Vasilescu, B., van Schuylenburg, S., Wulm, J., Serebrenik, A., & van den Brand, G. (2015). Continuous integration in a social-coding world: empirical evidence from github. doi:10.1109/ICSME.2014.62

Wang, J., Tang, L., Xuan, Y., & Damata, R. (2015, April). Visualized code review. EP Patent App. EP20,130,728,050. Google Patents. Retrieved from https://www.google.nl/patents/EP2859452A2?cl=en

Zanjani, M., Kagdi, H., & Bird, C. (2015). Automatically recommending peer reviewers in modern code review. *IEEE Transactions on Software Engineering (Volume:PP, Issue: 99)*. doi:10.1109/TSE.2015.2500238