Product Vision Document

Course: TI2806

ContextProject: Tools for Software Engineering

GitHub Division

28 April 2016

Maarten Sijm
Robin van Heukelum
Mathias Meuleman
Mitchell Dingjan
Youri Arkesteijn

SE TA: Bastiaan Reijm
Context TA: Aaron Ang

Context-Teacher: Alberto Bacchelli

Table of Contents

Abstract

In this document you will find our vision towards building the product we call Octopeer.

> *"Instead of offering revolutionary tools we strive to enhance the way people use the existing tools."*

This is a quote from chapter 5. It explains what we are trying to make here. Octopeer will make use of data analytics to enhance the way people use Pull Requests on GitHub. The engineers that will use our tool will receive useful feedback on the way they utilize the tools that can be found on the internet, such as BitBucket and in our case GitHub.

Team members

Name	NetID	StudentID
Mathias Meuleman	mmeuleman	4375629
Mitchell Dingjan	mdingjan	4348516
Maarten Sijm	msijm	4361083
Youri Arkesteijn	yarkesteijn	4357876
Robin van Heukelum	rvanheukelum	4369815

Introduction

In the modern era Software Engineering is dominated by Git. It is a tool that allows software engineers to commit and version pieces of software in a simple manner. Its origin lies in the fact that Linus Torvalds, known for the creation of the Linux kernel, was frustrated by the fact that the Software Versioning systems of that time (2005) did not meet his needs [1]. Therefore he decided to start off a revolution. A revolution in the Software Engineering culture called Git.

Considering the popularity of Git the GitHub platform was developed around 2008 [2]. It is a web-based Git repository hosting service. In June 2015 roughly 32 million people were reported to visit the site monthly [3]. GitHub offers users the ease of using an user interface to explore their own repository, or even contribute to other, open, software repositories. Also it allows other peers to review your code in a so-called Pull Request. The research possibilities are endless. This has been proven by mining its repositories in many different ways. One example is demonstrated in the paper written by Andy Zaidman et al. [4]. In their paper they showed ways to retrieve data from the rich sources GitHub offers.

In our personal project we want to elaborate on these researches by using a slightly different perspective. We do not want to look at the numbers generated by GitHub but rather at the behaviour of a peer reviewer in a Pull Request. Combining this data with the Data from GitHub we might be able to give new insights in the way people review code. This is our product, which we call Octopeer.

In the next chapters we will explain the basic ideas behind this product and the needs it addresses. We will start off in chapter 1 in which we will introduce the customers of our product. In chapter 2 the needs of the customer are shown and we explain why our product satisfy these needs. In chapter 3 the success factors of our product will be shown. Chapter 4 will explain why our product is unique, this will give us a certain advantage over the existing tools. Chapter 5 will show a brief overview of the technical sides of our project.

1. Target audience

Before we are going into details on our product we need to explain why someone would want to use Octopeer. In other words: what is our target audience?

This is a rather easy question: Software Engineers. Our product is designed for people that review code on GitHub regularly. These people can be your co-workers or other people from all over the world reviewing your piece of code. If they do this, questions will arise: what are the things engineers look at while reviewing code? Are there ways to optimize this process? The process of software review is prone to human errors and laziness. That is why we will develop Octopeer.

Octopeer will give insights in your personal style of doing code reviews based on your behaviour on GitHub. It will show engineers the time they spent on certain pieces of code and therefore people might consider changing their style of doing code reviews. The data we collect using our plugin will be analysed and shown to the user. These tips might be of good use for a Software Engineer.

2. Needs of the customer

Our plugin addresses a part of Software engineering most engineers do not like. Most engineers want to develop a good piece of working software. Peer reviewing other code is not the favorite task of an engineer. Besides, studies show that the ratio of functional fixes over other fixes (maintainability, etc.) is 25:75 [5]. This paper concludes with a certain vision:

> *"Our vision is that one day, code reviewers can input the amount of time they want to spend on review, and a tool will suggest the best-fitting task for them to review."*

This can be seen as a certain need for the customer. Octopeer will help an engineer realise where time might be lost. The data our plugin captures will be translated to a visualisation in which it will become clear what tricks might help optimize the reviewing process of the engineer.

3. Success Factors

There are several factors that enable us to deliver a successful product. These factors can be split up into 2 parts. The factors generated by the influence and behaviour of a user of our product, but also factors generated by the product.

The plugin is dependent on the reviewer itself. If they do not fully commit to reviewing we might get some noisy data from that plugin. Doing this, the data analytics might get noisy results as well. This means that we do not satisfy the needs of the user. On the other hand, if a user behaves excellent it becomes a success factor. In that case both we and the user get an overview of their behaviour. Further research can deduce certain optimization and the users know how they would have to adapt their style.

The product has an obvious success factor: the translation from data to visuals. If the correct data is analysed the user gets useful results. However, interpreting the data in a faulty manner, or collecting it wrong in the first place, will yield false results. This is something we must evade at all cost. This cannot happen because it is the core of our product. It the user is not satisfied with the things they get to see, they will de-install our plugin. The problem that arises here is that we also want to use the data for research purposes. Therefore we need a large amount of active users.

In short, we need to keep the customer happy by providing useful insights. In return an engineer has to be serious and consistent in its reviewing style, in order to get a fair image when analyzing the data.

4. Product uniqueness

As said in the introduction, it has already been shown that mining software repositories is possible using tools such as GHTorrent [4]. This enables researchers to view the data of many GitHub repositories. However this has to be checked and will not give the user any information based on their behaviour. Our product will give a user both feedback on their own behaviour as well as collect data.

Other than data collection there exist a lot of tools that help an engineer optimize code review. Examples of these tools are: Gitcolony, Upsource (by Jetbrains). Both of these tools help an engineer by providing an extensive toolset that enhances the way they perform code review. The crucial difference is that these tools try to change the behaviour of the engineer by offering the toolset rather than watching their behaviour. This means Octopeer will try to optimize the process with the tools that are already there. Instead of offering revolutionary tools we strive to enhance the way people use the existing tools.

5. Timeframe and Release

Due to the nature of the organization we need to finish our plugin within preferably 6 weeks. During these weeks we will research, implement and document the plugin. After that time we will need to cooperate with the other TSE-context teams in order to finish off the data analytics and the visualization part. If we have done this the product is ready for release.

Bibliography

1. Barr, Joe (2005, April 11) BitKeeper and Linux: The end of the road? Retrieved April 27, 2016, from https://www.linux.com/news/bitkeeper-and-linux-end-road
2. Preston-Werner, Tom (2008, October 20) GitHub Turns One! Retrieved April 27, 2016, from https://github.com/blog/185-github-turns-one
3. Sawers, P. (2015, June 17). GitHub by the numbers. Retrieved April 27, 2016, from http://venturebeat.com/2015/06/17/github-by-the-numbers-32m-people-visit-each-month-74-from-outside-the-u-s-36-from-europe/
4. Gousios, G., Vasilescu, B., Serebrenik, A., & Zaidman, A. (2014). Lean GHTorrent: GitHub data on demand. *Proceedings of the 11th Working Conference on Mining Software Repositories*
5. Beller, M., Bacchelli, A., Zaidman, A., & Juergens, E. (2014). Modern code reviews in open-source projects: Which problems do they fix? *Proceedings of the 11th Working Conference on Mining Software Repositories*