DARTS - Aims & Motivations About The Change

Andrea Cupito
Università degli Studi di Salerno
a.cupito@studenti.unisa.it

Giosuè Sulipano
Università degli Studi di Salerno
g.sulipano@studenti.unisa.it

Fabio Palomba
Università degli Studi di Salerno
f.palomba@unisa.it

ABSTRACT

Mining software repositories (MSR) is a software engineering field where software developers use data mining techniques to analyze the data in software repositories to extract useful and actionable information produced during the development process.

Types of software repositories include source control repositories, bug repositories, code repositories and even archived developer communications including mailing lists. In this document, we present the idea about the investigation of production code and the accompanying tests to value how they co-evolve by exploring a project's versioning system.

KEYWORDS

Test Smells, Data Mining, Software repositories, Commit

ACM Reference Format:
Andrea Cupito, Giosuè Sulipano, and Fabio Palomba. 2020. DARTS - Aims & Motivations About The Change. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Lehman has taught us that a software system must evolve, or it becomes progressively less useful. When evolving software, the source code is the main artefact typically considered, as this concept stands central when thinking of software. Software is not just code, it's made by other artifacts e.g. specifications, constraints, documentation, tests, etc. [2]

In this paper, we focus on how tests evolve with regard to the related source code, and how bad design choices made during the writing of production classes, e.g. code smells, could impact on the structure of test code. In order to study the co-evolution of production and test code, we rely on the data that is stored in Version Control Systems (VCS's). Version Control Systems is a category of software tools that help a software team manage changes to artifacts, e.g. source code, over time. It keeps track of every modification in a kind of database. Using a VCS to study co-evolution implies the prerequisite that the tests should be committed to the VCS alongside the production sources.

Our work focuses on DARTS (Detection And Refactoring of Test Smells) [1], an open-source IntelliJ plug-in which implements a detection mechanism to detect instances of three test smell types, i.e., General Fixture, Eager Test, and Lack of Cohesion of Test Methods, at commit-level and enables their automated refactoring through the integrated APIs provided by IntelliJ. We are going to exploit the capability of DARTS to work at commit level, so we would upgrade this feature by implementing an algorithm that enables the plug-in to analyze the VCS change history of the project in order to detect how the production classes change through the development of software projects and how these impact on the test code.

Seeing as DARTS focuses his attention on detecting test smells, our target is to detect the changes in production code that caused the smell itself; in order to do it we are going to choose only information that could help us in accomplishing our main objective, e.g. commit with warnings, errors, etc.

The first step of our work consists of extract data from repositories, to accomplish this objective we are aware of RepoDriller, an open-source Java framework that helps developers on mining software repositories and quickly exports CSV files. We selected it because is written in Java so the integration with DARTS will be more simple and natural. Otherwise, there's another open-source framework named PyDriller that absolves the same job but need some other interfaces to interact with DARTS.

2 MOTIVATIONS

In last years the software development process has been changed several times, in particular, Continuous Integration practices prevailed on the others. Continuous Integration (CI) is a development practice that requires developers to integrate code into a shared repository several times a day. According to this, the use of source code repositories increases significantly to help manage the progress of software projects. Software practitioners and researchers are recognizing the benefits of mining this information to support the maintenance of software systems, improve software design/reuse.
Based on this idea our work is born.

3 AIMS

Our main goal is to increase software developers' awareness about test smells in order to make better design decisions and avoid them next time. Previous research has shown that these smells may decrease both maintainability and effectiveness of tests and also they can be caused by code smells in production code, so it's important for the programmers to know what can cause a smelly code. In addition, they could learn about fault or bugs detected in production code.

REFERENCES

[1] Steafano Lambiase, Andrea Cupito, Fabiano Pecorelli, Andrea De Lucia and Fabio Palombe. *Just-In-Timee Test Smell Detection and Refactoring: The DARTS Project.*. In Proceedings of Seoul '20: ICPC International Conference on Program Comprehension (Seoul '20)

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[2] Andy Zaidman, Bart Van Rompaey, Serge Demeyer, and Arie van Deursen *Mining Software Repositories to Study Co-Evolution of Production & Test Code.*