GitCanary: A Tool for Analyzing Student Contributions in Group Programming Assignments

Jan Jaap Sandee
Open University of the Netherlands
Saxion University of Applied Sciences
j.m.j.sandee@saxion.nl

Efthimia Aivaloglou
Leiden Institute of Advanced Computer Science
Open University of the Netherlands
fai@ou.nl

ABSTRACT

Courses in computer science curricula often involve group programming assignments. In this paper, we present the GitCanary tool for monitoring project progress and member contributions using development productivity metrics. The tool is developed to support teachers and students with an overview that they can use during their guidance sessions. Feedback on the tool and its metrics was collected from students and teachers using interviews and questionnaires after utilizing it in a software development course. The results that the tool provides were found to be valuable in project guidance and to promote transparency in work distribution.

CCS CONCEPTS

• **Social and professional topics** → Computing education.

KEYWORDS

Group programming assignments, Git, Productivity, Metrics

ACM Reference Format:
Jan Jaap Sandee and Efthimia Aivaloglou. 2020. GitCanary: A Tool for Analyzing Student Contributions in Group Programming Assignments. In *Koli Calling*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Group programming assignments are a common aspect of software development courses in higher education. These projects enable students to practice and improve both their programming and their collaboration skills. However, students might not participate equally or contribute to programming aspects. Related work shows that group work is often not divided equally [2, 3], while students do not always inform their teacher of this problem. It has also been found that work equity concerns negatively impact students' attitudes toward group work, and that the role of the teacher is important to help students view group work positively [2].

In order for teachers to monitor and guide student groups, they would have to inspect the code written by each team member. One advantage that software projects have is that they are commonly maintained using a version management system like Git, which makes student activities transparent and enables continuous monitoring of student contributions over the course of their assignments [1, 9]. Automated tools have been proposed for monitoring student collaboration. The Teamwork Dashboard [8] analyzes online team discussion transcripts to visualize team mood, role distribution and emotional climate. TeamAnalytics [5] utilizes students' wikis and software version control system (svn) repositories, from which it extracts summaries of the number of files added, modified or deleted by each student. At the same time, in the industry and the open source software development world, various more fine-grained productivity metrics have been proposed for quantifying developer contributions using the source code repositories [4, 6].

In this paper, we present the GitCanary tool for monitoring project progress and contributions using development productivity metrics. Our aim for developing GitCanary is to support teachers and students with an overview they can use to assist in their guidance sessions. This tool was experimentally evaluated during a 1st year software development course in the spring semester of 2020. The 147 computer science students of the course were assigned to develop a game for Android in groups of an average of four students during three Scrum sprints of two weeks per sprint. Seven teachers were involved in guiding these groups and using GitCanary. At the end of the course, we explored teacher perceptions about the tool through six online interviews, and student perceptions using an anonymous questionnaire which was filled in by 93 students.

2 GITCANARY

The GitCanary tool[1] was developed iteratively, using meetings with the course teaching team to inform its design decisions. GitCanary utilizes PyDriller [7] to analyze Git repositories and generate a JSON report. This report is stored in a Django-based REST server. A front-end system built in VueJS and ChartJS is used to display this data using charts.

GitCanary allows users to select a date range and specific branch for examining project activity. GitCanary gives an overview of the following metrics, in order of display in Figure 1:

Lines of Code (LOC) Cumulative LOC added, modified, or deleted.

Comments Cumulative comment lines added, modified, or deleted.

Documentation Cumulative lines added, modified, or deleted in .md files in the project.

Git Blame Who last modified each line at the master version.

Complexity Cumulative number of methods someone introduced or modified with complexity *n*.

[1]The source code of GitCanary, is available at https://gitlab.com/gitcanary.

Koli Calling, August 10–12, 2020, online

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Figure 1: GitCanary results for a team of four students

Method Size Cumulative number of methods someone introduced or modified with size *n*.

Time of Commit Scatter plot of time of day of commits.

The tool provides explanations on all metrics, as well as more detailed data upon selection of specific results.

3 TEACHER AND STUDENT FEEDBACK

The results of the teacher interviews and the student questionnaires indicate that both parties were positive towards using metrics for monitoring progress and contributions.

All six interviewed teachers indicate wishing to use this tool in the future. The primary approach of teachers was opening the tool together with students and using the metrics results to guide their discussion and ask directed questions. They also used it as way of explaining concepts of software quality using examples from the results. The use of the tool was found to promote balance and support planning. For example, when two students were found to contribute more at the early sprint, the team planned for complex tasks to be allocated to the other two students in the later sprints. An additional advantage that students indicated in their open text responses was that they did not have to point out slacking members themselves, which is in line with prior work where it has been found that students do not enjoy ratting out fellow students [3]. There were also students who drew motivation from seeing the numbers go up and getting competitive within their team.

Negative feedback was also provided. Two teachers expressed distrust on the calculated metric results due to their fragmented understanding of their meaning or calculation. Teachers also stressed the importance of explaining to students what the metrics represent and how they will use the results in the course. One teacher expressed that the tool could be used for grading, but all others agreed that a method such as this should not affect grading and should require a teacher interpreting and discussing the results with the students. Several students expressed in their open text responses that the benefits of the tool are limited unless it is used and known from the beginning of their projects, while one student expressed being uncomfortable being monitored through the tool.

In terms of the used metrics, both teachers and students were positive about LOC, Complexity, and Method Size. LOC was found to give a clear indication of productivity because, due to its simplicity, it is easy to understand. Complexity and Method Size gave the opportunity to discuss quality concepts, especially when used side by side with LOC. Conversely, the Git Blame metric turned out to be not as useful because it was found hard to understand and utilize. Finally, several students indicated that the produced results across metrics are not representative when they practice pair programming or switch accounts.

4 LIMITATIONS AND FUTURE WORK

The threats to the validity of the feedback received for the tool include that it was collected from one run of a single course, which was also given remotely due to the Corona pandemic. Moreover, the feedback might have been biased due to interviewees being co-workers of the first author and the students being asked to fill out the survey during class activities.

As future work, we would like to enrich the tool with code quality metrics and to explore in more depth its application to later courses where students have more programming experience. Moreover, experiments on different setups for the use of the tool in courses could provide more insight on its effect on student behavior.

REFERENCES

[1] K. Buffardi. 2020. Assessing Individual Contributions to Software Engineering Projects with Git Logs and User Stories. In *Proceedings of the 51st ACM SIGCSE*. ACM, 650–656.

[2] K. J. Chapman and S. Van Auken. 2001. Creating positive group project experiences: An examination of the role of the instructor on students' perceptions of group projects. *Journal of Marketing Education* 23, 2 (2001), 117–127.

[3] C. L. Colbeck, S. E. Campbell, and S. A. Bjorklund. 2000. Grouping in the dark: What college students learn from group projects. *The Journal of Higher Education* 71, 1 (2000), 60–83.

[4] G. Gousios, E. Kalliamvakou, and D. Spinellis. 2008. Measuring developer contribution from software repository data. In *Proceedings of the 2008 international working conference on Mining software repositories*. ACM, 129–132.

[5] J. Kim, E. Shaw, H. Xu, and G. V. Adarsh. 2012. Assisting Instructional Assessment of Undergraduate Collaborative Wiki and SVN Activities. In *International Conference on Educational Data Mining*.

[6] J. Lima, C. Treude, F. Figueira Filho, and U. Kulesza. 2015. Assessing developer contribution with repository mining-based metrics. In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 536–540.

[7] D. Spadini, M. Aniche, and A. Bacchelli. 2018. Pydriller: Python framework for mining software repositories. In *Proceedings of the 26th ESEC/FSE*. ACM, 908–911.

[8] H. Tarmazdi, R. Vivian, C. Szabo, K. Falkner, and N. Falkner. 2015. Using Learning Analytics to Visualise Computer Science Teamwork. In *Proceedings of the ACM ITiCSE Conference*. 165–170.

[9] A. Zagalsky, J. Feliciano, M. Storey, Y. Zhao, and W. Wang. 2015. The Emergence of GitHub as a Collaborative Platform for Education. In *Proceedings of the 18th ACM CSCW*. ACM, 1906–1917.