Git in an Educational Context

Åsmund Haugse[1] and Trond Aalberg[1,2][0000−0001−5593−0860]

[1] Norwegian University of Science and Technology, Norway
[2] Oslo Metropolitan University, Norway

Abstract. The version control system Git is commonly used in computer science education. Best practice use of Git is a relevant professional skill that students need to learn, but Git also is a natural choice for efficient collaboration on programming assignments. The learning context is however different from the professional context. In this paper we explore the implications Git has in an educational context using a questionnaire and result implies that Git influences some students' experience with learning and group work. However, it also verifies previous findings that using Git is beneficial for student collaboration. Data from the use of Git can potentially inform students and educators about aspects of group work. A prototype mirroring tool using GitLab data has been developed to investigate the data's potential and semi-structured interviews with students and teachers were conducted to evaluate and explore the tool. Results suggest that GitLab data is well-suited to provide students new insight into their work and educators with an efficient method for monitoring project work and student groups.

Keywords: Git · Collaborative Learning · Mirroring Tool.

1 Introduction

Version control systems, such as the de facto standard tool Git, are crucial in software development, offering functionality to collaborate and contribute asynchronously and distributed. Several studies have explored the use of Git in the classroom and its benefits, disadvantages, and challenges in an educational context. Feliciano, Storey, and Zagalsky [6] found that the use of Git and GitHub in software engineering courses was beneficial to and well-received by most of the students. Hsing and Gennarelli [11] found that students who used GitHub in the classroom felt a greater sense of belonging to the field, suggesting that Git can provide benefits beyond its technical capabilities.

Less research has examined the challenges Git may introduce for students in the process of exploring and learning programming and participating in group work. The tool is complex and adds to the challenge of learning. Furthermore, development with Git is transparent and contributions by individual students may expose differences in skills and competencies. Knowledge about how students experience Git is thus important e.g. to identify and compensate for the negative effects it may have on the learning experience. A questionnaire with respondents from two different courses has been used to provide insight into students' experiences using Git, and how Git affects the learning experience in group work.

When using Git, detailed data about commits, issues and other aspects are logged and potentially available for inspection. In an educational context, such data may describe and inform about the performance of the group as well as individual students. However, most students pay little attention to the history of their repository, and current features for inspecting the history is not particularly useful for educators that may need to inspect and compare numerous repositories. To explore the use of Git data in an education context, we present an experimental mirroring tool in the form of a dashboard. The purpose of the tool is to support students gaining insight into work habits and collaboration. For an educator, the dashboard informs about the performance of groups and individuals as well as assist in the identification of problems. Through interviews and demonstrations, the dashboard was presented to and tested by both students and educators. The research questions these contributions attempt to answer are:

RQ1: What are students' perceptions on and experiences with using Git?
RQ2: How can Git log data be visualized in a mirroring tool?
RQ3: What does a mirroring tool with Git data offer?

2 Background

2.1 Version Control Systems

Version control is a term describing a system that maintains records of changes to a set of files, allowing users to access specific versions at a later time [1]. Today, Git is the most common version control system (VCS). It is open-source, free to use, efficient and well-suited for handling large projects and non-linear development. Various repository hosting services with support for Git are offered. The most popular being GitHub which is subscription-based although educational and free subscriptions with some limitations are offered. GitLab is another popular service providing users with free hosting of repositories. GitLab can also be hosted locally, which makes it a natural choice for use in education where there is a need for local ownership. Git repository services typically offer additional functionality to navigate and perform actions on repositories such as overview of the commit history, branches and files, as well as processes to support the management and quality assurance of merging code from individual developers into the main branch. Repository services such as GitLab and GitHub also provide developers with the option to manage tasks using issues.

2.2 Git in Education

Git, and the features offered by repository services adds up to a complex tool that can be difficult to learn and understand. Lawrance, Jung, and Wiseman [15] examined the technical obstacles encountered when exposing students to version control as well as the perceived benefits. Students were often initially confused by Git's inner workings, but they appreciated what it offered and did not feel that learning Git was a waste of time. By using Git in a project with multiple contributors, students learned of the benefits of adopting good habits such as branching by topic and minor incrementing changes to prevent massive merge conflicts. Kelleher [12] also found that students enjoyed using Git knowing that they were using industry-standard technology. Similar to the study of Lawrance et al., students encountered technical issues when using Git, but these diminished as students became more experienced.

Incorporating Git into software education is not a trivial task and stepwise introduction may be a relevant approach. Haaranen and Lehtinen [8] suggested using GitHub as an LMS to distribute and collect assignments. They initially require students to clone assignments and gradually introduced new Git features such as asking students to make incremental changes, committing changes often and add tasks that requires using branches and merging.

To give students an authentic experiences with Git, assignments where students must work in groups are beneficial. Although merge conflicts can occur when working alone, they are more common in projects with multiple collaborators and even more common in projects where collaborators have little experience with Git. Feliciano [5] conducted a study on the use of GitHub in multiple software development courses looking at perceived benefits and challenges. Benefits noted by students was gaining experience with an industry-standard tool but also its transparency, in seeing when and how their team members work, helped keep each other accountable for the group project. Additionally, students found it helpful to use GitHub as a portfolio for their projects.

2.3 Project-based Learning

Software development courses frequently implement project-based learning in teams [21] and the use of Git also relates to the collaborative learning experience. Grouping students together and providing tools for efficient collaboration naturally increases the capability of what they can achieve, allowing for more open-ended assignments and improved learning [7]. Software development in teams, however, requires skills beyond that of technical competence. A group seldom consists of equally experienced and talented students. Ideally, this leads to the more experienced students explaining or giving pointers to the less experienced and some argue that group diversity is a necessary condition for collaborative learning [3] or that diversity creates increased learning opportunities [13]. On the other hand, research on collaboration and group dynamics also present several negative scenarios (e.g., internal conflicts, free-riders, competitive behaviors) that can occur in collaborative work [13]. One of the main concerns are slackers or free-riders, group members who do not contribute their part to the group project [18]. Colbeck, Campbell, and Bjorklund [2] found that the potential for slacking increased as team size increased. Having a slacker can be detrimental to the learning of all members of the group and students would try to avoid teaming up with slackers. Hall and Buzwell [9] concludes that many students' frustration with slackers stems from the slackers often receiving the same mark as those who contributed the most, due to the final product often being evaluated instead of the process leading up to it. In a study by Hendry, Ryan, and Harris [10], both educators and students rank the frequency of problems occurring when working in groups. The third most frequent problem was that of the dominant student, someone who talks a lot and tries to control the direction of the discussion, often preventing others from contributing. Students reported that the case of dominant students was difficult to solve and that problems arose because of how the group was working, not because of the task at hand.

2.4 Learning Analytics and Mirroring Tools

Data describing activities and individual contributions has a potential usage in learning analytics and mirroring tools. The Society for Learning Analytics Research (SoLAR) defines learning analytics as: *The measurement, collection, analysis, and reporting of data about learners and their contexts, for the purposes of understanding and optimizing learning and the environments in which it occurs* [20]. Research on learning analytics has generally focused on how information can support making improvements to the education. Robles and Gonzalez-Barahona [19] explored the mining of student software repositories in a learning analytics context. They present a semi-automated solution to gather data from students' Git usage in programming assignments. The data was used to assess code quality, plagiarism, automated feedback, and the creation of personal exams. Soller et al. [22] focuses on monitoring as a technique to provide educators and students with insights into student groups and detecting good and bad collaboration patterns.

Mirroring tools are orthogonal to learning analytics and describe systems that collect and aggregate data about students and reflect this information back to the user. Typically, students in a learning situation generate data, which an educator or the students can reflect on through visualizations. For students, these systems aim to enhance self-awareness of one's actions and behavior [22, 4] to improve upon. Educators use mirroring tools to gain insight into how students work and to identify those that need guidance. Dietsch et al. [4] found that a mirroring tool for student activity in a collaborative software development setting helped identify reoccurring roles in student groups. For example, they identified students in the role of free-riders as students who were responsible for less than 10% of the code.

Mirroring tools designed for educators are sometimes referred to as teacher dashboards. The usefulness of teacher dashboards comes down to how efficiently and effectively they convey information to a teacher [16]. Research on the usefulness of teacher dashboards has shown that they can be both helpful and insightful to teachers and that they can be used to give teachers more information on their students' activity [14]. However, research on mirroring dashboards also shows that they do not consistently improve the detection accuracy of a teacher concerning student groups [23, 17].

3 Method

3.1 Cases

Two courses at the NTNU Department of Computer Science has been used as cases for the research presented in this paper. TDT4140 Software Development teaches software project management and development processes. The students work on a single project and must complete a set of mandatory demonstrations, presentations, and other deliverables throughout the semester. Groups consist of 7 students and are put together at random across study programs. IT2810 Web Development teaches technologies and methods for developing web-based solutions. Students deliver one individual and 3 group-based projects. Groups consists of three members and students can choose to work with friends or be randomly assigned to a group. Deliverables are in the form of a repositories in GitLab and running prototypes. Another difference in the setup is that IT2810 also includes a peer review process where the code was inspected and reviewed by fellow random students.

Students enrolled in IT2810 have slightly more experience with development compared to those of TDT4140 given that students usually take IT2810 in the third or fourth year, whereas TDT4140 is scheduled for the second year. TDT4140 is a mandatory course in many study programs, whereas IT2810 is optional and students often enroll because they have a genuine interest in the topic. Thus, it is likely with differences in the overall experience of these student as well as in the perception and experience of using Git.

3.2 Questionnaire

An exploratory study in the form of a questionnaire has been used to provide insight into these students' perceptions of and experiences with Git (RQ1). The questions used can be divided into the following categories:

- Motivation and previous experience.
- Experiences and habits using Git, perceived usefulness and usability.
- Git's transparency and feedback on code.
- Effects on the social dynamics of group work.

Students were invited to answer the questionnaires via Blackboard and e-mail. IT2810 students were invited in January 2021 and TDT4140 students were invited in May 2021. In total 91 students answered the questionnaires, 48 from IT2810 (app. 25% of enrolled students) and 43 from TDT4140 (app. 10%).

The same set of questions were used for the two courses, but with different wording for certain questions. For instance, the questionnaire for IT2810 had questions about projects, whereas the questionnaire for TDT4140 had questions about sprints. Questions are worded as statements to which the student answers with how much they agree with the claim (Likert scale). For example, one statement is "I only want to share code that I know is good", to which the answer options are "Not relevant", "Strongly disagree", "Somewhat disagree", "Neither nor", "Somewhat agree" or "Strongly agree". The average time spent answering the questionnaire was 8 minutes and 6 seconds.

The questionnaire was online using the service Nettskjema hosted by the University of Oslo. Although none of the individual questions of the questionnaire ask about personal information, an application to conduct the project was sent to the Norwegian Centre for Research Data (NSD). This decision was made based on the assumption that someone could guess what student had answered the questionnaire if viewing answers from an individual.

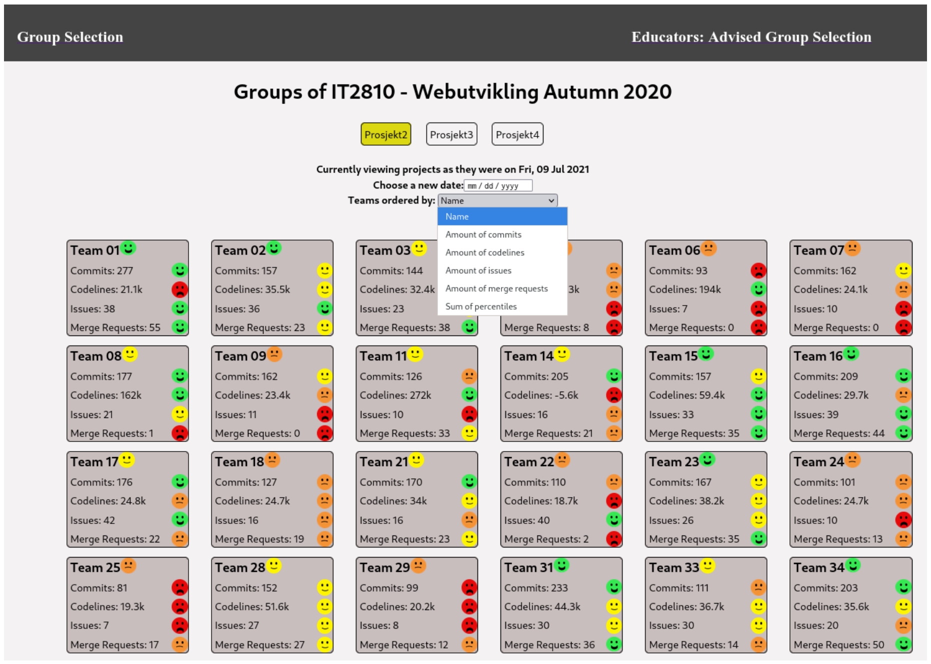
3.3 The Dashboard Mirroring Tool

To address RQ2 and RQ3, a web-based dashboard was implemented. As a research method this falls within design science where a developed artifact represents an approach to solve the problem and the evaluation of the artifact is used to gain insight into qualities and characteristics of the solution. The dashboard is proof of concept prototype intended to aid student groups in self-reflection and educators in gaining insight into student groups' performance.

The dashboard is simple in architecture, consisting of a browser-based client, a MongoDB database, and a Node backend for intermediate communication between the dashboard and GitLab. GitLab has a well-documented API listing how to query specific resources such as groups or projects. Groups logically group one or more projects (repositories) or sub groups. Codebases are hosted as projects and have accompanied issue trackers, repositories, merge requests and more. The dashboard queries the Group API for all subgroups of a course and all projects of each subgroup and data is stored in an intermediate database to reduce the number of requests sent to GitLab. The current implementation requires manually updating with new data when needed and makes some ad hoc assumptions on how groups and projects are named and organized.

The dashboard can be interacted with in different ways and offers various views for exploring the data. The Advised Group Selection is primarily intended to support educators to get an overview and easily identify specific groups according to the metrics (figure 1). Each group is presented with information about number of commits, codelines, issues and merge requests; each number accompanied by a color coded icon of varying colors to indicate what quartile the student group is in for the specific metric. From bad to good, the colors are red, orange, yellow, and green, red meaning the group is in the bottom 25% of all groups for that metric, orange 25-50%, yellow for the 50-75% , and green for 75-100%. Teams can be ordered by number or by a specific metric in the display. The presentation shows aggregated data up until the current date, but can also display data for specific dates.

The dashboard also offers a set of views to visualize the metrics for specific groups and projects. Individual groups can be selected either from the above mentioned Adviced Group Selection (for educators) or from a more plain view just showing group numbers (for educators and students). The metrics reflect the tasks and work in various ways. Merge requests and issues represent work tasks at a higher level than commits and can be useful metrics to inspect the progress



Group Selection

Educators: Advised Group Selection

Groups of IT2810 - Webutvikling Autumn 2020

Currently viewing projects as they were on Fri, 09 Jul 2021

Choose a new date: mm / dd / yyyy

Teams ordered by: Name

Team 01
Commits: 277
Codelines: 21.1k
Issues: 38
Merge Requests: 55

Team 02
Commits: 157
Codelines: 35.5k
Issues: 36
Merge Requests: 23

Team 03
Commits: 144
Codelines: 32.4k
Issues: 23
Merge Requests: 38

3k
Merge Requests: 8

Team 06
Commits: 93
Codelines: 194k
Issues: 7
Merge Requests: 0

Team 07
Commits: 162
Codelines: 24.1k
Issues: 10
Merge Requests: 0

Team 08
Commits: 177
Codelines: 162k
Issues: 21
Merge Requests: 1

Team 09
Commits: 162
Codelines: 23.4k
Issues: 11
Merge Requests: 0

Team 11
Commits: 126
Codelines: 272k
Issues: 10
Merge Requests: 33

Team 14
Commits: 205
Codelines: -5.6k
Issues: 16
Merge Requests: 21

Team 15
Commits: 157
Codelines: 59.4k
Issues: 33
Merge Requests: 35

Team 16
Commits: 209
Codelines: 29.7k
Issues: 39
Merge Requests: 44

Team 17
Commits: 176
Codelines: 24.8k
Issues: 42
Merge Requests: 22

Team 18
Commits: 127
Codelines: 24.7k
Issues: 16
Merge Requests: 19

Team 21
Commits: 170
Codelines: 34k
Issues: 16
Merge Requests: 23

Team 22
Commits: 110
Codelines: 18.7k
Issues: 40
Merge Requests: 2

Team 23
Commits: 167
Codelines: 38.2k
Issues: 26
Merge Requests: 35

Team 24
Commits: 101
Codelines: 24.7k
Issues: 10
Merge Requests: 13

Team 25
Commits: 81
Codelines: 19.3k
Issues: 7
Merge Requests: 17

Team 28
Commits: 152
Codelines: 51.6k
Issues: 27
Merge Requests: 27

Team 29
Commits: 99
Codelines: 20.2k
Issues: 8
Merge Requests: 12

Team 31
Commits: 233
Codelines: 44.3k
Issues: 30
Merge Requests: 36

Team 33
Commits: 111
Codelines: 36.7k
Issues: 30
Merge Requests: 14

Team 34
Commits: 203
Codelines: 35.6k
Issues: 20
Merge Requests: 50

Fig. 1. The Advised Group Selection view

of a project and how workload is distributed over time. Each visualization has been designed to highlight certain aspects. Line charts are suitable to show how data changes over time, bar charts are useful to show how a metric is distributed along some categorical value, doughnut charts convey values relative to each other, each slice representing their share of some total value.

Different visualizations that are implemented in the dashboard are:

- Merge requests and issues for a project implemented as a line chart component. Lines of different color are used to show a timeline of respectively open merge requests or issues, and the sum of merge requests or issues.
- Commits and lines and of code for a project distributed by dates, implemented as bar chart. Showing either metrics for the team members combined, or distributed by members.
- Commits and lines of code distributed by member, implemented as doughnut charts (figure 2).
- A list of commits with basic data such as commit message, author, date and number of changed codelines. The component is mainly intended to get more details about the commits and a possibility to inspect the commit more in detail in GitLab.

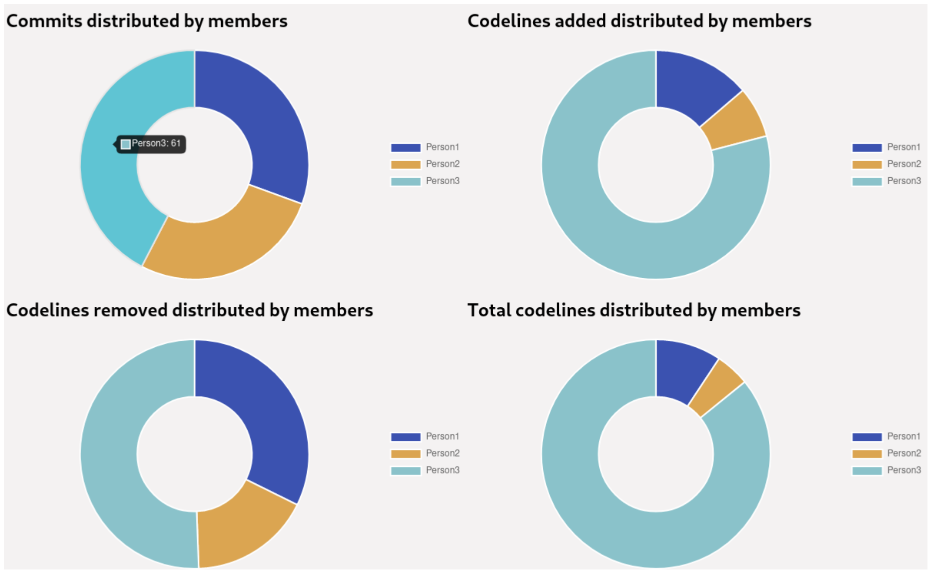


Fig. 2. Doughnut charts showing commit and code line distribution by members

3.4 Interviews

To evaluate the dashboard and to gain insight into the usefulness of the overall approach, semi-structured interviews were held with students and educators. Students received an invitation via email invite to participate in one-on-one interviews and 9 students from IT2810 and 13 students from TDT4140 participated. To collect feedback on the educator's view, educators who use version control (GitLab or GitHub) in their courses were invited and 11 educators were interviewed.

Interviews were semi-structured and following an interview guide, with follow-up questions and new questions as needed. The same interview guide was used to interview students from both courses. Interviews opened with some questions about the student's group to get to know the interviewee and to make them comfortable in the setting. Then, students received an introduction to the dashboard and its functionality were described to them. Finally, students are instructed to view the dashboard at their own pace and reflect out loud what the visualizations mean to them. Students were suggested to use sentences structured as "This [surprises, makes sense to, is interesting to] me because" and to express what they like and dislike about the dashboard's features. The educators were first presented with the tool through a demonstration, and the interview as a more open ended discussion of the different features and usage scenarios in the context of each interviews educational needs.

All recorded interviews have been transcribed using the software solutions Temi [56] and oTranscribe[57] and analyzed using NVivo, a qualitative data analysis software.

4 Results

4.1 Questionnaire results

In the following we discuss some of the findings from the questionnaire. The two courses represent different populations and for some of the questions it is interesting to identify and analyse the difference.

The students responding to the questionnaire report different level of experience with Git. For TDT4140, 72.1% answered they were somewhat or not experienced with Git, compared to IT2810 where 81.3% of students said they were quite or very experienced with Git. The difference is likely caused by the courses being at different level and having different student populations. Students of IT2810 are likely to have a different relationship with Git compared to TDT4140 students. Substantiating the claim that IT2810 students are more experienced we observe that 89.6% of IT2810 students use Git for private projects compared to 44.2% of TDT4140. These results may indicate that private projects is a main source for experience and skills in using Git in these student populations. 100% of respondents from both courses agree or strongly agree that Git simplifies working with others. Unsurprisingly, most perceive Git as valuable for work after studies, 97.9% and 86% of IT2810 and TDT4140 students respectively agree or strongly agree.

In terms of students perception and experience with git as a tool, we observe that for IT2810, only 50% agree that Git easy to learn, 72.9% agree it is easy to use, and 52.1% agree that it is easy to understand. In contrast, for TDT4140, only 32.6% agree Git is easy to learn, 53.5% agree it is easy to use, and 37.2%, agree it is easy to understand. The large number of students who find Git hard to learn may be a result of little or no formal training or that it is a complex tool to gain experience in. Student answers show that fewer TDT4140 students understand how Git works (65.1%) compared to IT2810 (87.5%). Although not all understand Git, most think it is necessary for project work and that it makes it easier to track progress in the codebase (90.7% of TDT4140 and 93.8% of IT2810). For IT2810, 6.3% students agree that Git has been demotivating to use, 50% agreed it has been frustrating to use at times. Results of TDT4140 are slightly worse, almost one out of five (18.6%) agree it has been demotivating to use, 7% said it had caused more trouble than it has been helpful, and 58.1% agree it has been frustrating to use at times. Furthermore, four students (9.3%) from TDT4140 agree that Git has prevented them from focusing on programming. No students of IT2810 reported the same. The more experienced IT2810 students mainly have few problems with Git, compared to the less experienced TDT4140 students, but also shows that Git may impact the learning experience for some.

Regarding Git's transparency, the two courses are similar in their answers. Of IT2810 and TDT4140 respectively, 85.4% and 81.4% agree that it is OK that staff can view their Git history, 79.2% and 67.4% are OK with strangers seeing their code, and 95.8% to 95.3% are OK with group members seeing their code. In a course context only educators and team members are likely to see students' code. Thus, the positive numbers are reassuring.

Offering quantifiable metrics of contribution in terms of completing issues, amount of commits, and similar metrics, 39.6% of IT2810 students agree that using GitLab makes group work more competitive, 37.5% disagree, and 20.8% neither agree nor disagree. On the other hand, only a smaller portion of TDT4140 agrees with the same statement. Most students agree with Git makes differences in skill level more apparent, namely 75% of IT280 and 81.4% of TDT4140. Furthermore, 50% and 48.8% respectively agree with Git works best if all team members are on the same skill level. Close to one out of four (27%) of IT2810 students and 41.9% of TDT4140 students agree with the statement Working with GitLab makes it worse to be on a lower skill level than the team, but we also see that 50% and 25.6% respectively agree with GitLab makes it less difficult to work with someone on a lower skill level than yourself. Whether students experience this a problem or not is likely to be related to skill level and group size. Supplementing the former statements, a third (33.3%) of IT2810 students and a fourth (27.9%) of TDT4140 students agree with the statement of Development with GitLab turn team members who accomplish less into scapegoats. Worth noting is that also 45.8% of IT2810 and 51.2% disagree with the same statement. The statement responses may reflect how less experienced students feel about their contribution, but it can also be more experienced students who feel this way. The slightly larger amount of IT2810 students who agree may correlate to team sizes, because team members who contribute less will be more detrimental to the project. For the statement of "Git results in the best members hijacking the development process", a concerning 45.8% of IT2810 students and 37.2% of TDT4140 students agree with the statement.

4.2 Dashboard evaluation

The dashboard was evaluated through demonstrations in combination with semi-structured interviews with both students and educators. The intended outcome of this evaluation is knowledge related to the specific visualizations and selection of data and the perceived usability of the overall dashboard approach.

The advised group selection screen is of most interest to educators, providing quick insight into the composition of all student groups of an entire course. The current implementation was by none perceived to be perfect, and most educators suggested changes that would fit better to their needs. Despite its shortcomings, the general feedback was that the overarching idea of the dashboard was helpful. The use of colors to quickly convey information visually, was well-received by all educators. Most educators, however, were indifferent to smiley faces, emphasizing the use of colors instead.

The selected metrics of commits, lines, merge requests, and issues, were perceived to be good choices to capture student group performance, although not equally valuable. For example, one educator suggested that for an exercise where the curriculum is on issues, that metric would be the only of interest. Moreover, he noted that for courses where students had yet to learn merge requests and issues, it would not make sense to use those metrics.

Educators were also pleased to see how issues and merge requests developed over time; some admitted not considering the data points as indicators beforehand. As metrics to provide insight, an educator said

> *The curves (of the two diagrams) are very useful because they provide insight into how the students experience the scope of the projects and how they work towards it.*

When asked if the dashboard could be helpful to them, all educators confirmed. Seven of the educators expressed that they would be interested in using the dashboard's Advised Group Selection view to gain insight into how student groups perform throughout the semester. They expressed that they would use it to spot struggling student groups, decide what student groups to give attention to or contact. One educator noted that having issues visualized could impact how he structured his course. Others indicate that more knowledge about how students use Git could impact how they teach and train students in the use of the tool. Several instructors said they already consult GitLab to gain insight into student groups and that the same insight is more efficiently available from using the dashboard. One educator is quoted below on the topic:

> *That is more useful than what GitLab offers out of the box. And I think it would pretty much cover the typical use cases that we currently use the data for. It's a bit easier having the dashboard because you don't have to do sort of a manual checking yourself.*

The anonymity of students was also a topic in the interview since user names can be displayed. Some expressed that the dashboard should hide names to ensure a fair assessment of students. One educator expressed that showing student names would improve their trust in the system and help validate the system's accuracy. Beyond this, the educator claimed that showing or hiding names would not change the value of the dashboard.

As a mirroring tool, the dashboard aims to aid students in self-reflection and provide new insights into their group's performance. Overall, most students expressed that the visualizations made sense to them based on their impression of their group's work. When asked what wrong assumptions an outsider, e.g., an educator, could make about their group, all students shared the same concern — it does not capture all aspects of group work. All students of TDT4140 expressed that the dashboard gave the impression that their group had significant differences in how much each member contributed because it does not account for pair programming or other project-related work (e.g., report writing, planning). Almost all IT2810 students expressed similar concerns.

When asked if they were made aware of anything new, a handful of students replied that they had become more aware of Git as a tool and the correlation (or lack of) correlation between commits and lines. Some students expressed that viewing their data visualized changed their impression of their own contribution to the project. One student, claiming to be the least experienced member of their group, expressed:

Now looking at project four, like I felt that I was contributing very little to the project sometimes, but looking at it now, it's like, I'm definitely had fewer commits or whatever than others, but it's not that uneven...

In contrast, several students expressed that the dashboard served to confirm notions they had of their group. All students were told to use the dashboard with their project experiences in mind. However, some students experienced the dashboard not to be useful. Although they agreed with the visualizations, they did not express any new insights, realizations or enjoyment.

Students were mainly of the impression that the dashboard should not hide students' names, although some also made arguments for anonymization. Concerning the harmful effects of having students' contributions on display, some students were concerned about how those who contributed less would experience this transparency. Although the same data is available in GitLab, differences are more apparent when data is aggregated.

5 Conclusion

The survey conducted to answer the research question on students' perceptions on and experiences with using Git, serves both to verify previous research and to probe into the relatively unexplored social implications of Git in educational group work. Previous finding on the challenges of learning and using Git is verified but also the positive perception of its usefulness and relevance. Results also indicate that using Git has effect on group dynamics and how students perceive their team members. Respondents have different opinions on statements such as Git makes skill differences more apparent, Git makes it worse to be on a lower skill level than other, Git makes team members that accomplish less into scapegoats, Git results in the best members hijacking the development process. These are all problems where educators need to listen to the voice of the few. As long as some have these experiences they need to be addressed when we make use of Git in education.

Respondents on the questionnaire where recruited from two courses at NTNU and the findings are more of value as qualitative insight rather than quantitative, given the low number or participants. It is also worth commenting that many study programs (even at NTNU) have a more systematic approach to the use of Git in the education and possible future work is to survey these courses to learn about the effect when Git is better integrated in the teaching.

To answer the research questions on how to visualize Git data and what these visualizations have to offer, a dashboard solution using GitLab data has been implemented and evaluated by both educators and students. The visualizations are designed to provide insight into student groups work and offers educators functionality to identifying groups that may be lagging behind or have anomalies in how team members contribute. The interviews support the relevance of such a tool. Feedback from interviewees shows that the selected visualizations can be helpful to gain insight into work and work patterns in retrospect. The visualizations allowing for comparing student contributions were well-received by both students and educators. Furthermore, the dashboard design was perceived to have high affordance and conveyed data to students and educators in a way that made sense.

Interviews with students and educators have solidified the need for better understanding of the data and critical thinking when viewing the visualizations. Although providing some insight into how students work, the data on its own do not do accurately capture all aspects of software development in groups. Findings also indicate that the visualizations can be helpful to students to improve self-reflection, planning in groups, and make students aware of work patterns. However, results show that individuals viewing the dashboard in retrospect made few new realizations.

The implemented prototype dashboard served well to demonstrate the use of mirroring tool using git data. In particular the educators had many opinions about what to show and in what way, indicating that such a dashboard solutions needs to be highly configurable to succeed as an attractive tool to use in education - which is a topic for future work.

References

1. Chacon, S., Straub, B.: Pro git. Springer Nature (2014)
2. Colbeck, C.L., Campbell, S.E., Bjorklund, S.A.: Grouping in the Dark. The Journal of Higher Education **71**(1), 60–83 (Jan 2000)
3. Curşeu, P.L., Pluut, H.: Student groups as learning entities: The effect of group diversity and teamwork quality on groups' cognitive complexity. Studies in Higher Education **38**(1), 87–103 (Feb 2013)
4. Dietsch, D., Podelski, A., Nam, J., Papadopoulos, P.M., Schäf, M.: Monitoring Student Activity in Collaborative Software Development. arXiv:1305.0787 [cs] (Jun 2013)
5. Feliciano, J.: Towards a Collaborative Learning Platform: The Use of GitHub in Computer Science and Software Engineering Courses. Thesis, University of Victoria (2015), accepted: 2015-08-31T21:15:20Z ISSN: 1906-1927
6. Feliciano, J., Storey, M.A., Zagalsky, A.: Student Experiences Using GitHub in Software Engineering Courses: A Case Study. In: 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C). pp. 422–431 (May 2016)
7. Freeman, K.A.: Attitudes toward Work in Project Groups as Predictors of Academic Performance. Small Group Research **27**(2), 265–282 (May 1996), publisher: SAGE Publications Inc
8. Haaranen, L., Lehtinen, T.: Teaching Git on the Side: Version Control System as a Course Platform. In: Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '15. pp. 87–92. ACM Press, Vilnius, Lithuania (2015)
9. Hall, D., Buzwell, S.: The problem of free-riding in group projects: Looking beyond social loafing as reason for non-contribution. Active Learning in Higher Education **14**(1), 37–49 (Mar 2013), publisher: SAGE Publications
10. Hendry, G.D., Ryan, G., Harris, J.: Group problems in problem-based learning. Medical Teacher **25**(6), 609–616 (Nov 2003), publisher: Taylor & Francis Ltd

11. Hsing, C., Gennarelli, V.: Using GitHub in the Classroom Predicts Student Learning Outcomes and Classroom Experiences: Findings from a Survey of Students and Teachers. In: Proceedings of the 50th ACM Technical Symposium on Computer Science Education. pp. 672–678. SIGCSE '19, Association for Computing Machinery, New York, NY, USA (Feb 2019)
12. Kelleher, J.: Employing git in the classroom. In: 2014 World Congress on Computer Applications and Information Systems (WCCAIS). pp. 1–4 (Jan 2014)
13. van Knippenberg, D., Schippers, M.C.: Work Group Diversity. Annual Review of Psychology **58**(1), 515–541 (2007)
14. Kosba, E., Dimitrova, V., Boyle, R.: Using Student and Group Models to Support Teachers in Web-Based Distance Education. In: Ardissono, L., Brna, P., Mitrovic, A. (eds.) User Modeling 2005. pp. 124–133. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2005)
15. Lawrance, J., Jung, S., Wiseman, C.: Git on the cloud in the classroom. In: Proceeding of the 44th ACM technical symposium on Computer science education. pp. 639–644. SIGCSE '13, Association for Computing Machinery, New York, NY, USA (Mar 2013)
16. van Leeuwen, A., Rummel, N.: Comparing teachers' use of mirroring and advising dashboards. In: Proceedings of the Tenth International Conference on Learning Analytics & Knowledge. pp. 26–34. LAK '20, Association for Computing Machinery, New York, NY, USA (Mar 2020)
17. Mazza, R., Dimitrova, V.: CourseVis: A graphical student monitoring tool for supporting instructors in web-based distance courses. International Journal of Human-Computer Studies **65**(2), 125–139 (Feb 2007)
18. Oakley, B., Felder, R.M., Brent, R., Elhajj, I.: Turning student groups into effective teams. Journal of Student Centered Learning **2**(1) (2004), publisher: Citeseer
19. Robles, G., Gonzalez-Barahona, J.M.: Mining student repositories to gain learning analytics. An experience report. In: 2013 IEEE Global Engineering Education Conference (EDUCON). pp. 1249–1254 (Mar 2013)
20. Siemens, G.: Learning Analytics: The Emergence of a Discipline. American Behavioral Scientist **57**(10), 1380–1400 (Oct 2013), publisher: SAGE Publications Inc
21. Sindre, G., Giannakos, M., Krogstie, B.R., Munkvold, R.I., Aalberg, T.: Project-Based Learning in IT Education: Definitions and Qualities. 147-163 (2018), accepted: 2018-06-13T13:17:13Z Publisher: Universitetsforlaget
22. Soller, A., Martínez, A., Jermann, P., Muehlenbrock, M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. International Journal of Artificial Intelligence in Education (IOS Press) **15**(4), 261–290 (Dec 2005)
23. Voyiatzaki, E., Avouris, N.: Support for the teacher in technology-enhanced collaborative classroom. Education and Information Technologies **19**(1), 129–154 (Mar 2014)