Integrating GitLab Metrics into Coursework Consultation Sessions in a Software Engineering Course

Integrating GitLab Metrics into Coursework Consultation Sessions in a Software Engineering Course

Sukru Eraslan*[1], Kamilla Kopec-Harding+, Caroline Jay*, Suzanne M. Embury*, Robert Haines+, Julio César Cortés Ríos*, Peter Crowther+

*Department of Computer Science, University of Manchester, Manchester, United Kingdom

+Research IT, University of Manchester, Manchester, United Kingdom

Abstract

Software developers use version control systems for collaborative coding. These systems are integrated into several software development platforms (including GitLab and GitHub) which support additional software engineering functionalities. Using these platforms in an educational context allows students to gain skills relevant to industry, whilst providing a means of keeping track of their activities. In this paper, we investigate the effect of presenting teams of students with GitLab metrics about their performance at coursework consultation sessions (*checkpoint* sessions), with a particular focus on the number of issues assigned and completed, and the number of commits made to the repository. A comparative analysis of project marks in two consecutive academic years indicates that these checkpoint sessions may lead to better student outcomes. An interview study with students and teaching assistants identified viewing the GitLab metrics in the checkpoints as an opportunity to see the relative contributions of team members and address resulting issues, and as a catalyst for improving engagement with the team project. The study also identified drawbacks of using the metrics too simplistically, and suggested that it was important to consider the quality and amount of written code, as well as the number of times someone committed to the repository.

Keywords: software engineering education, undergraduate education,

[1]Corresponding author: sukru.eraslan@manchester.ac.uk

collaborative software development, version control system, Git

1. Introduction

Software engineering courses aim to equip students with the skills required for diverse software engineering practices, including requirements analysis, design, implementation, testing, deployment, and maintenance. In these courses, students are usually asked to work collaboratively with other students on a particular project to allow them to experience real-life team working practices. They are also often required to use a version control system (VCS) that supports collaborative coding. These systems allow software developers to work simultaneously and record changes to the code base over time so that they can access specific versions of it later. A VCS records the activities of contributors, such as when they make revisions, when they initiate a line of development that they want to temporarily isolate from the rest of the team, when they integrate their work with the work of others and when they undo work. VCSs are also integrated into several software development platforms (such as GitLab[2] and GitHub[3]) which provide additional functionality, such as allowing software developers to create and resolve issues (items of work 'to do'). Using software development platforms in an educational context helps students gain the skills needed for working in industry. It also allows educators to collect data about the way the students interact with these platforms and the work they are doing. This data can then be used to identify problematic student behaviour such as low engagement and poor working practices. If this behaviour is identified early then remedial action can be taken to improve the learning outcomes of the students.

The activity metrics available from software development platforms, including VCSs and issue-tracking systems, have already been used for various purposes in an educational context such as describing student behaviours [1, 2, 3, 4,

[2] https://gitlab.com/
[3] https://github.com/

5], correlating these behaviours with outcomes [6, 7, 8], predicting grades with supervised machine learning approaches [9, 10, 11, 12, 13, 14], profiling student behaviours [15, 16], conformance checking [17, 18] and as a marking aid [19]. In particular, Glassy [2] presented a basic analysis of student repositories on a VCS. The analysis was based on the dates that revisions were made, gaining insights into how students work such as identifying groups of students who work steadily and those who make periodic revisions. In this analysis, the author suggested that VCSs allow the detection of problematic student work patterns and to provide feedback to them about how they can improve their development process. Mierle *et al.* [9] examined the relationship between a wide range of data from VCS repositories and attainment, and found that the strongest predictor was lines of code (LOC) written. Kay *et al.* [16] conducted a sequential analysis of student teamwork on a VCS and an external issue-tracking system. The teamwork of students was first represented in terms of sequential events. These sequences were then divided into multiple sessions/resources. For example, if a sequence in a session is represented as $< (2T1), (4S3) >$, it means that two ticket (or issue) actions performed by the same person were followed by four VCS actions performed by three people. These sequences were then analysed by using a frequent sequential pattern mining algorithm. The patterns generated for different groups with various levels of success were then compared. For example, it was found that the best performing group had more sessions with three or more of their six members. Finally, Baumstark and Orsega [18] examined whether or not introductory computer science students followed the "code-a-little, test-a-little" process while developing their programming assignments. They found that even though the students made small commits which focused on a single aspect of the program, not all of the students properly incorporated testing as part of their iterative development. All these studies show that the analysis of student repositories provide insights into how students work.

Gary & Xavier [20] presented a continuous feedback tool which visualises the metrics of specific activities of students in a software development platform to support them while working on a project within a team. The usefulness of

the tool was evaluated based on the ratings of the students and they mainly found it useful. Dietsch *et al.* [6] proposed a road-map for a software engineering course, which includes two coursework consultation sessions where the metrics from a VCS and an external issue-tracking system are used for feedback and discussion. The students were asked to rate different aspects of the road-map (not the consultation sessions directly), including its workload, the value of their contributions to the team by the end of this road-map, etc. Even though the students found the workload to be high, they tended to be satisfied with the workload when they considered the learning outcomes. They also saw themselves as a valuable part of their team. Similarly, Neyem *et al.* [21] also proposed a road-map for a software engineering course. The proposed road-map involves weekly coursework consultation sessions where a project tracking tool is used to keep track of the work of each student such as which requirements they completed and which requirements they are currently working on. This tool is also integrated with GitHub to view how the requirements are implemented and tested. Neyem *et al.* [21] stated that their road-map was found to be successful as most of the students delivered their projects with high client satisfaction, but they did not conduct a formal evaluation with students.

The wider educational research literature also reports software engineering courses that incorporate coursework consultation sessions without the integration of activity metrics [22, 23]. For example, a recent work by Marques *et al.* [22] found the use of reflexive weekly monitoring in software engineering courses to be beneficial for students.

In this paper, we use a qualitative research study with students and teaching assistants (TAs) to develop a richer understanding of their opinions of using VCS and issue-related metrics to provide feedback about team activity. Metrics about the activity of contributors were presented to students at two coursework consultation sessions (i.e. *checkpoint* sessions) in a software engineering course at the University of Manchester in the UK in the academic year 2018-2019. Students worked on a team project for 12 weeks, where the code was held in a GitLab repository. Each week they were given a set of tasks to be completed

by the end of the course. In the checkpoint sessions, run in Week 5 and Week 10, each team had a meeting with a TA about their progress, using metrics extracted from GitLab to focus the discussion. Interviews were conducted with 20 students and six TAs, with the aim of understanding their opinions about the strengths and drawbacks of using the metrics as a mechanism for feedback and progress monitoring.

To investigate whether the checkpoint sessions improved student outcomes, we also compared the proportion of students who passed the course in the academic year 2017/2018, where no checkpoints were held, with the academic year 2018/2019, where two checkpoints were conducted.

In the rest of the paper, we describe which metrics were used and how they were presented at the checkpoint sessions (Section 2), and outline our methodological approach (Section 3). After that, we present and discuss our results along with concluding remarks, respectively (Section 4 and Section 5).

2. GitLab Metrics in Coursework Consultation Sessions

In the software engineering course at the University of Manchester in the academic year 2018/2019, students were asked to build a web application to create, list, search for and manage events in a particular city. They were given a list of specific tasks each week, such as allowing an event manager to add, update and delete an event. Students were required to use GitLab to work collaboratively with other students within a team. GitLab is a web-based software development platform which is a Git repository manager. Git is one of the most popular distributed version control systems[4] allowing software developers to create a local copy of a shared team remote repository and work on their local repositories independently of the shared remote repository. When they complete their work, they can synchronise the local changes with the remote repository [24]. In Git, a repository may include a development branch

[4] `https://git-scm.com/`

that is sometimes used as the main line of development. When this convention is followed, developers can create other branches to have independent lines of development, and they can perform merge operations to integrate different branches. For example, developers can create a branch from the development branch to develop a new feature without affecting the main line of development. When they complete the development of the feature, they merge the feature branch into the development branch. An example of this process is illustrated in Figure 1 where *commits* (i.e. revisions) are illustrated with circles, the black ones on the development branch and the white ones on the feature branch.

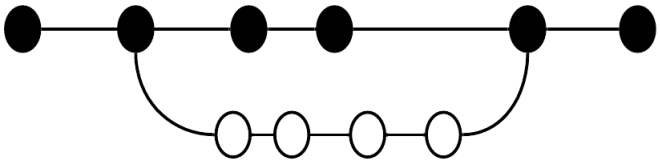

Figure 1: An example of a commit graph in Git (Direction: $\rightarrow$)

GitLab also includes additional features to support various software engineering practices including a Wiki, issue-tracking, and CI/CD (continuous integration/continuous delivery). It also allows developers to review the code written by other developers to find errors and problems in commits and through merge and pull requests (i.e. requests for merging two branches).

A report of specific GitLab metrics was presented to students online at each of two checkpoint sessions. Each team was shown their own reports only. In the first session, the number of commits and the number of issues assigned and closed by each student were included in the report. In the second checkpoint session, the number of commits by each student per week were also included. A summary of team attendance at the course sessions was also added to the report in the second checkpoint session. Team attendance was computed as the ratio of the number of individual attendances of team members to the maximum possible individual attendances of all members. For example, if a team has six members, then the maximum possible individual attendances of all members for ten weeks will be 60 ($6 \cdot 10 = 60$). If there were 54 individual attendances

of team members, the team attendance is calculated as 54 out of 60, or 90%. These metrics were presented in the form of interactive graphs and tables.

Figure 2 shows an example report from the second checkpoint session using mock data. Figure 2-A shows how the number of assigned and closed issues for each member were presented and Figure 2-B shows how the number of commits and the number of assigned issues for each member were visualised in the reports of both of the checkpoint sessions. Figure 2-C and Figure 2-D show additional metrics provided in the report of the second checkpoint session. Figure 2-C shows how the overall team attendance was visualised and Figure 2-D shows how the distribution of commits over different weeks for each member was visualised.

231 students enrolled on the course in the academic year 2018/2019. The teaching staff comprised four lecturers and six TAs. Experienced TAs were available to help new TAs resolve any conflicts in the marking process, ensuring consistency. The TAs were not allocated to specific teams, and the teams might have different TAs in the first and second checkpoint sessions. Four of the TAs participated in both of the sessions and two of them participated in the first session only.

3. Methods

To investigate the impact of presenting the metrics to students, we asked the following research questions:

1. *What are the opinions of students and TAs regarding the activity metrics presented in the checkpoint sessions?*

   To answer this research question, we conducted a semi-structured interview study with both students and TAs after the checkpoint sessions. We used the Standards for Reporting Qualitative Research (SRQR) guidelines to develop our interview study and report its results [25].

2. *Does this kind of checkpoint session help students perform better in their coursework?*

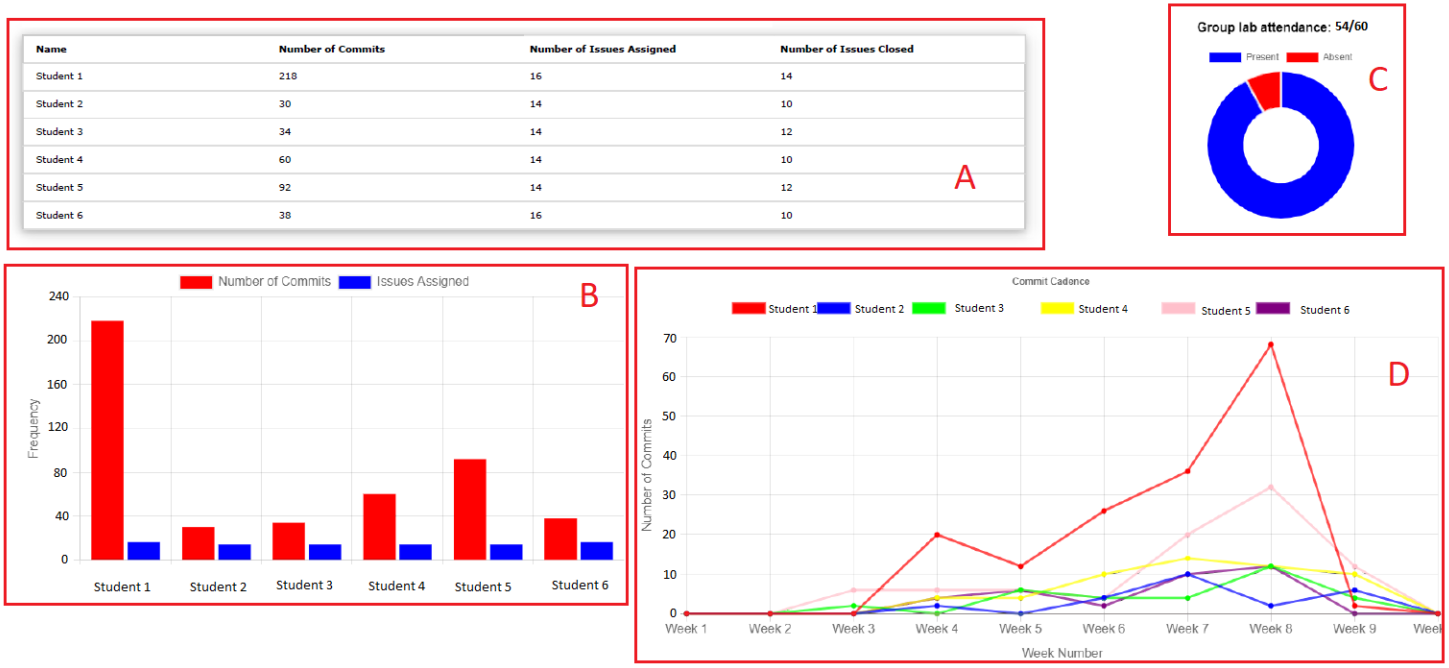

Figure 2: An example report from the second consultation session using mock data

To answer this research question, we compared the proportion of students achieving a pass mark in two consecutive academic years with and without the checkpoint sessions.

3.1. Interview Study

The interview study was reviewed and approved by the University of Manchester Proportionate University Research Ethics Committee (Ref: 2019-6195-9626). Recruitment and data collection were carried out by researchers who were not connected to the course as the participants might have been reluctant to give negative or critical feedback/responses to the course lecturers. The members of the research team directly involved in the course only had access to the anonymised data.

Participants: We invited all the students and TAs on the course to take part in the study. We interviewed 20 student volunteers and all of the six TAs on the course; therefore, 9% of the students (20/231) and 100% of the TAs (6/6) on the course participated in the interview study. The students were a self-selecting convenience sample. We did not record which team they belonged to for higher confidentiality, but there were student participants from both the same teams and different teams. All the TAs were recruited before the grading sessions (Week 12) and all the students were recruited after the grading sessions to ensure that the members of each group of participants were interviewed under the same conditions.

Questions: We used an initial set of questions to prompt discussion in our semi-structured interview as listed below.

An initial set of questions for the student interviews are as follows:

1. How do you rate your overall satisfaction with the consultation sessions (Very dissatisfied, Dissatisfied, Neither, Satisfied, Very satisfied)? Why?
2. How do you rate your satisfaction with the inclusion of the reports of GitLab metrics as a supportive document in the consultation sessions (Very dissatisfied, Dissatisfied, Neither, Satisfied, Very satisfied)? Why?

3. Do you consider the number of commits as an appropriate metric to show the level of the contributions of the team members (Strongly disagree, Disagree, Neither, Agree, Strongly agree)? Why?
4. Do you consider the number of issues as an appropriate metric to show the level of the contributions of the team members (Strongly disagree, Disagree, Neither, Agree, Strongly agree)? Why?
5. What did you learn from the consultation sessions?
6. How did the consultation sessions change your and your team's behaviour?
7. What other metrics can be included in the report of GitLab metrics?
8. How can the consultation sessions be improved?

An initial set of questions for the TA interviews are as follows:

1. Do you think the consultation sessions helped the students? If yes, how? If no, why not?
2. How do the reports of GitLab metrics affect the consultation sessions?
3. What do you think about student satisfaction about the consultation sessions/reports of GitLab metrics?
4. Did the consultation sessions and reports change the behaviour of the team and individuals?
5. Do you have any other comments or suggestions for improvement?

Materials: In the interview study, we provided example reports to remind our participants which metrics were included and how they were presented.

Procedure: Participants read the information sheet explaining the study and provided informed consent. The information sheet and the consent form are available from our external repository (see Open Data Section). We recorded all of the interviews on a recording device owned by the University.

Analysis: We transcribed the audio files and analysed them using inductive thematic analysis, following the methods described in [26], as we were interested

in identifying, analysing, and reporting patterns or themes within data, rather than developing a theory. The thematic analysis was conducted by two coders. The first coder extracted a subset of interview quotations and generated the themes for both the student and TA interviews. Both of the coders then matched the interview quotations with the themes. The coders resolved disagreement through discussion.

Researcher characteristics: Both of the coders have a Ph.D. degree. The first coder has a background in computer science and data engineering and has teaching experience in several programming courses. The first coder has also conducted different kinds of user studies for his previous research projects. The second coder has a background in data management and data science. Both coders also have experience on data handling, statistics and using the Git version control system. They did not know any of the participants personally.

3.2. Comparative Analysis of Marks

To investigate whether presenting GitLab metrics in checkpoint sessions helped students perform better, we compared the proportion of students achieving a pass mark ($< 40\%$) in two consecutive academic years, the first where students did not take part in checkpoint sessions, and the second where they did. As the checkpoint sessions were conducted in the academic year 2018/2019, we compared marks with the previous academic year (2017/2018). There were no major changes in course content, requirements, assignments or marking criteria during this period. The lecturers were the same. There were six TAs in 2018/2019 and five TAs in 2017/2018 and three of these TAs were involved in both of the academic years.

This analysis was conducted as part of a parallel learner analytics research study which looks at the relationship between student use of software development platforms and attainment. This study was reviewed and approved by the University of Manchester Proportionate University Ethics Committee (Ref: 2019-5108-10350).

Participants: 250 and 231 students registered and commenced the course in the academic years 2017/2018 and 2018/2019, respectively. A small number of students in each cohort chose not to take part in the learner analytics study: 2017/2018 - 4 non-participants, 2018/2019 - 3 non-participants. The distribution of team sizes for the two academic years is summarised in Table 1. The proportion of students placed in larger teams of 7 was lower in 2018/2019 (21/231, 9.1%) than in 2017/2018 (56/250, 22.4%).

Table 1: Distribution of student team size by cohort year

	Academic Year			
	2017/2018		2018/2019	
Team Size	N Teams (%)	N Students (%)	N Teams (%)	N Students (%)
5 Members	4 (9.8)	20 (8.0)	0 (0.0)	0 (0.0)
6 Members	29 (70.7)	174 (69.6)	35 (92.1)	210 (90.9)
7 Members	8 (19.5)	56 (22.4)	3 (7.9)	21 (9.1)
Total	41 (100)	250 (100)	38 (100)	231 (100)

Analysis: Anonymous pass/fail project marks for each cohort were obtained from the course team, excluding those for students who chose not to take part in the learner analytics study (2017/2018: 246 grades, 2018/2019: 228 grades). A Chi-square test of independence (complete case analysis) was carried out to examine the association between the proportion of students receiving a pass mark and cohort year.

By default, students were awarded a team grade unless marks were redistributed between members by negotiation with course tutors or deducted on an individual basis due to insufficient contributions. Consequently, marks are strongly correlated within teams. Chi-square statistics have been adjusted for the clustering of observations within teams using the approach of Donner (1989) [27]. Statistical analysis was carried out using R 3.5.2 (R Core Team, 2018) [28] with the aod library (v1.3.1; Lesnoff and Lancelot (2012) [29])).

4. Results

4.1. Overall Student Satisfaction with Checkpoint Sessions

During the interview study, we asked the students closed questions about whether they were satisfied with the checkpoint sessions and the use of the GitLab metric reports in the sessions. We also asked them whether or not they see the number of commits and the number of issues as appropriate indicators of the level of contributions of the team members.

Almost all the students were satisfied with the consultation sessions overall to some degree (Satisfied: 55% - 11/20, Very satisfied: 35% - 7/20) (see Figure 3). The responses also show that many students were satisfied with the use of the metric reports in the sessions to some degree (Satisfied: 45% - 9/20, Very satisfied: 30% - 6/20) (see Figure 4).

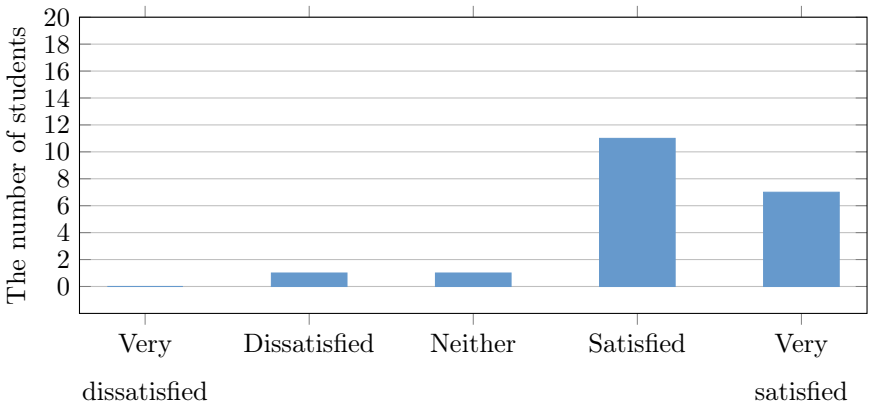

Figure 3: Overall satisfaction with the checkpoint sessions

Half of the students did not agree to some degree that the number of commits is an appropriate metric for showing the contributions of team members (Strongly disagree: 5% - 1/20, Disagree: 45% - 9/20, Neither: 25% - 5/20) (see Figure 5). More students agreed to some degree that the number of issues assigned/closed is an appropriate metric (Strongly agree: 10% - 2/20, Agree: 50% - 10/20), although this was not a consensus (Disagree: 20% - 4/20, Neither: 20% - 4/20) (see Figure 6).

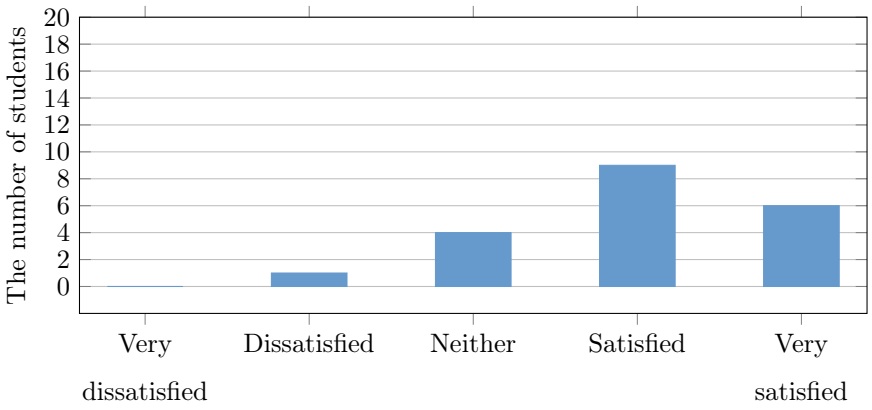

Figure 4: Satisfaction with the use of GitLab metrics in the sessions

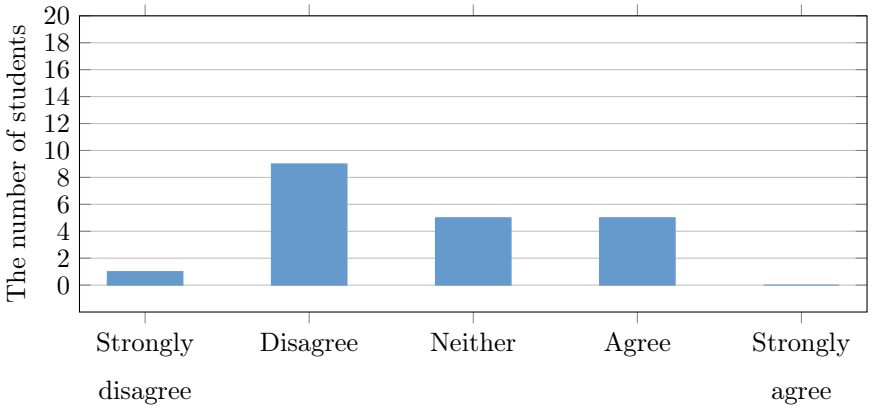

Figure 5: Agreement with the appropriateness of the number of commits

We also asked the students to explain the reasons for their answers. We analysed their answers thematically, and present the results in the following section.

4.2. Results of the Thematic Analysis

A high level of agreement was achieved between the first coder and the second coder in matching the quotations to the primary themes in both the student interviews (91.9%, n=346) and the TA interviews (93.6%, n=141), where an un-

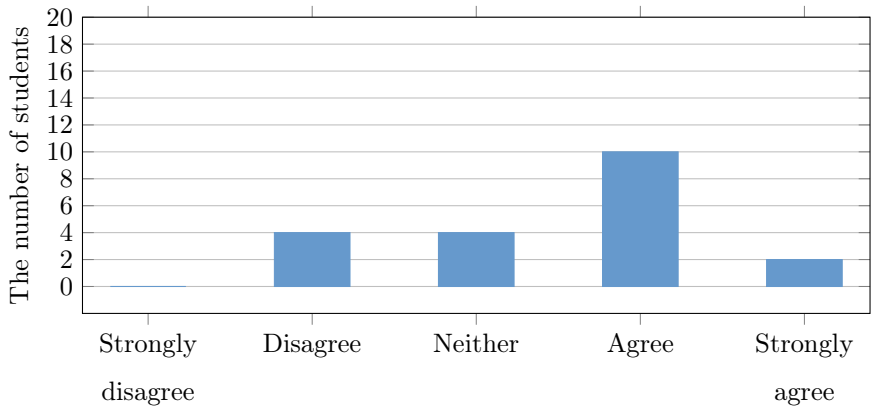

Figure 6: Agreement with the appropriateness of the number of issues assigned/closed

weighted Cohen's Kappa[5] shows almost perfect agreement for both the student interviews (kappa=0.91) and the TA interviews (kappa=0.93).

The mean duration of the student interviews was 9:16 (minutes:seconds) with a standard deviation of 4:41, whilst the mean duration of the TA interviews was 17:02 with a standard deviation of 13:45.

In this paper, we focus on the themes related to the benefits and drawbacks of using the GitLab metrics in the checkpoint sessions, as well as what other GitLab data would be useful to include. The list of relevant quotations along with their corresponding themes are available in our external repository (see Open Data Section).

4.2.1. Benefits

Our thematic analysis revealed three primary positive themes related to the benefits of using the GitLab metrics report in the checkpoint sessions, which were improvements in student practices, the benefits of seeing the GitLab metrics, and the benefits of the checkpoint sessions. Table 2 and Table 3 show these

[5] <0 Less than chance agreement, 0.01-0.20 Slight agreement, 0.21-0.40 Fair agreement, 0.41-0.60 Moderate agreement, 0.61-0.80 Substantial agreement, 0.81-0.99 Almost perfect agreement [30]

Table 2: The student themes with regard to the benefits

Primary Themes	**% of Students**	**Sub-themes**	**% of Students**
Improvements in student practices	85% (17/20)	Software engineering skills	35% (7/20)
		Engagement	35% (7/20)
Benefits of the GitLab metrics	90% (18/20)	Indication of the behaviour of students	90% (18/20)
Benefits of the checkpoint sessions	95% (19/20)	Clarification, feedback, Q&A	65% (13/20)
		Team progress check	60% (12/20)
		Opportunity for dealing with team problems	55% (11/20)
		Motivation to continue working in the same way	5% (1/20)

primary themes and their sub-themes along with the percentage of the participants who mentioned each theme for the students and TAs respectively, where the sub-themes are sorted according to the percentage of the participants who mentioned them.

Improvements in Student Practices: Most of the students (85% - 17/20) and the TAs (83% - 5/6) mentioned improvements in student practices including software engineering skills (Students: 35% - 7/20, TAs: 50% - 3/6) such as task allocation/execution, the use of Git/GitLab, and other skills related to design and implementation, and improved engagement (Students: 55% - 11/20, TAs: 67% - 4/6) including involvement in coursework, attendance, etc.

A TA describes how the checkpoints improved software engineering skills as follows:

> *"I think these sessions help the students and specifically, I think, they help the students to improve the way they plan and execute the various tasks in the team coursework."* (TA - P1)

Table 3: The TA themes with regard to the benefits

Primary Themes	**% of TAs**	**Sub-themes**	**% of TAs**
Improvements in student practices	83% (5/6)	Engagement	67% (4/6)
		Software engineering skills	50% (3/6)
Benefits of the GitLab metrics & attendance reporting	100% (6/6)	Prompt for discussion and feedback	100% (6/6)
		Opportunity for correcting Git profile configuration	50% (3/6)
		Indication of the behaviour of students	33% (2/6)
		Reflection on the Git/GitLab best practices	17% (1/6)
Benefits of the checkpoint sessions	100% (6/6)	Opportunity for dealing with team problems	67% (4/6)
		Clarification for coursework/session, feedback on coursework	50% (3/6)
		Team progress check	33% (2/6)
		Motivation to continue working in the same way	17% (1/6)

Another TA describes the improvement in engagement:

> *"People who were not participating well in their teams, they are now, you know. They realise they are seen from an outsider not just from their colleagues who can send them emails and they can ignore it. They know we also see the problem and we will take actions even if their colleagues didn't."* (TA - P5)

Benefits of the GitLab metrics: Almost all the students (90% - 18/20) believed that the reports of GitLab metrics were beneficial as they gave an early warning sign of problematic behaviours (such as low engagement and poor working practices):

> *"Interesting to be honest. I think it's a good thing to do because again if you do have a team member that's not doing any work, it can be really useful formative evidence to show that they haven't actually been doing anything so their commits might be really low and so on. It can actually be really good evidence for it."* (Student - P1)

Whilst some TAs (33% - 2/6) also mentioned that the GitLab metrics reflected the behaviour of students, all of them stated that these reports were used by both TAs and students as a prompt for discussion and feedback:

> *"It triggered important discussions on the issue creation, assignment and closing. Members with no issues, or a small number of issues compared to others, would reflect on the need and discuss the strategy to mitigate its effect in the future."* (TA - P1)

Half the TAs also mentioned that the reports of GitLab metrics allowed them to recognise problems with the Git configuration, so they could advise students on how to correct them:

> *"Some people were using their own IDs, so the GitLab system couldn't recognise, instead of the university email address while making the commit. So we could correct that."* (TA - P2)

One TA (17% - 1/6) also mentioned that the metrics prompted students to reflect on Git/GitLab best practices, in particular dividing the tasks into sub-tasks and creating an issue for each of these sub-tasks, and committing changes more frequently.

Benefits of the Checkpoint Sessions: Almost all the students (95% - 19/20) and all the TAs mentioned the benefits of conducting the checkpoint sessions. 65% (13/20) of the students and 33% (2/6) of the TAs stated these sessions were beneficial in terms of being able to clarify, provide feedback and answer questions:

> *"It was helpful to have some feedback on our project because it's not like the other course units where you get feedback after every exercise, here we just have a big project [...] if we have any problems, it's a lot easier to fix them sooner rather than later."* (Student - P10)

60% (12/20) of the students and 33% (2/6) of the TAs stated that these sessions allow TAs to check how students work overall within a team:

> *"I learnt about the statistics and how well my team has been working on. You know, it's like a bit of a check up to see that the team is on point [...]."* (Student - P5)

55% (11/20) of the students and 67% (4/6) of the TAs stated that these sessions allowed them to confront team problems. A TA describes this opportunity as follows:

> *"The sessions served as an opportunity for staff and TAs to detect and resolve team problems before it was too late. For example, issues to do with the inactive team members were reported in the sessions and discussed and they were resolved in these sessions."* (TA - P1)

A student had a similar view:

> *"If any particular team member say, is not pulling their weight and doing their work properly, [...] we could consult with that team member to check why they have been not working as they should have. So, yes, it was useful."* (Student - P5)

One student and one TA also stated that these sessions motivated those who had been working productively to continue doing so (Students: 5% - 1/20, TAs: 17% - 1/6).

4.2.2. Drawbacks

Our thematic analysis also revealed some drawbacks of using the GitLab metrics in the checkpoint sessions and of the checkpoint sessions themselves. Table 4 and Table 5 show the primary and sub-themes along with the percentage of the participants who mentioned each theme for the students and TAs respectively. The sub-themes are sorted according to the percentage of the participants who mentioned them.

Drawbacks of the specific GitLab metrics: All the students and 67% of the TAs mentioned the drawbacks of the particular GitLab metrics used. 90% (18/20) of the students thought that they were not reflective of the work distribution of team members, especially the number of commits, as commit styles can differ from one member to another. One student commented:

> *"A metric is the number of commits, but in our team, there are people who would work for an entire day, make sure everything works and then commit and basically would have only one commit. Whilst others are working gradually, and like commit today and then a few hours after that and a few hours after that. So, the same amount of work could equal, for one person one commit and for another even 20."* (Student - P4)

Others felt that the number of issues was also not reflective of the work distribution of team members:

Table 4: The student themes with regard to the drawbacks

Primary Themes	**% of Students**	**Sub-themes**	**% of Students**
Drawbacks of the specific GitLab metrics	100% (20/20)	Not reflective of work distribution	90% (18/20)
		Technical/configuration issues	20% (4/20)
		Not reflective of real-life practices	20% (4/20)
		Manipulable	15% (3/20)
		Limited availability to students during/before sessions	10% (2/20)
Drawbacks of the checkpoint sessions	95% (19/20)	Limitations of resources	60% (12/20)
		Not comprehensive	50% (10/20)
		Lack of strict checks on individual progress	10% (2/20)
		Limited ways to report inactive members	5% (1/20)
		Timing of the second session	5% (1/20)
		Different TA for each session	5% (1/20)

> *"While issues are important to structure who does what, it might be that some issues are not as extensive or as hard as compared to other issues. So, I don't think in that sense it's an appropriate metric to show the level of contributions."* (Student - P5)

One TA also agreed with the students that the metrics were not reflective of the work distribution of team members. The students also mentioned that these metrics could be manipulated (15% - 3/20), they were easily affected by technical/configuration issues (20% - 4/20), they were not reflective of real-life practices (20% - 4/20) and their availability to students was limited (for example, they were not available outside of the sessions in this form) (10% - 2/20).

Table 5: The TA themes with regard to the drawbacks

Primary Themes	**% of TAs**	**Sub-themes**	**% of TAs**
Drawbacks of the specific GitLab metrics	67% (4/6)	Not very representative of team performance	50% (3/6)
		Not reflective of distribution of work	17% (1/6)
		Unintended consequences	17% (1/6)
		No detailed definition of metrics	17% (1/6)
Drawbacks of the checkpoint sessions	83% (5/6)	Not beneficial from the technical side	33% (2/6)
		Assessment arrangement	33% (2/6)
		Similar to regular sessions	17% (1/6)
		More obvious benefits for course leaders than students	17% (1/6)

The TAs mentioned that these metrics could cause unintended consequences, such as focusing on only the metrics instead of actual work (17% - 1/6), that they were not necessarily representative of team progress (50% - 3/6) and that there was no detailed documentation about the metrics available (17% - 1/6).

In spite of the drawbacks, 45% (9/20) of the students stated that the GitLab metrics were appropriate in certain circumstances. For example, when students all have issues of comparable size and complexity, these metrics can reflect work distribution accurately (30% - 6/20), when there are significant differences in the metrics between students, this can highlight underlying differences in behaviour for the team to discuss (10% - 2/20) and when different kinds of metrics are considered together, they provide a better reflection of students' behaviours (5% - 1/20).

Drawbacks of the Consultation Sessions: Although 95% (19/20) of the stu-

dents and 83% (5/6) of the TAs mentioned at least one drawback of the way the checkpoint sessions were conducted, they focused on different aspects. Students mentioned the limitations of resources. For example, they stated that the consultation sessions should have been longer (60% - 12/20):

> *"It was rather hectic [...] There were loads of groups to go round in quite a short amount of time, so the TA doesn't really spend too much time with each group and it was quite a short thing. So if they made it maybe more sessions and more TAs can spend a longer time with each group, then maybe they can go into more depth about what to do and stuff like that."* (Student - P15)

The students also did not find the consultation session comprehensive enough in terms of technical support, clarification on marking scheme details, and Q&A (50% - 10/20):

> *"I guess it could be more improved [...] because the way I see it there's a heavy reliance on the GitLab statistics by the TAs. So, in my opinion, I think they should also assess the group dynamic as well. I understand that that might be difficult too because no group is designated one TA, so that task might be difficult. But if the TA would spend more time as opposed to just having the GitLab statistics to ask questions, to ask more in-depth questions outside whatever report they have on hand, I think that would be more useful to make the group think better on how they work."* (Student - P5)

Other drawbacks mentioned by students included the late semester scheduling of the second session (5% - 1/20), not having the same TA for both of the sessions (5% - 1/20), no anonymous way to report inactive team members (5% - 1/20), and a lack of strict check of the progress of individuals (10% - 2/20). Even though 33% (2/6) of the TAs also thought that the consultation sessions were not beneficial from the technical side, they also thought that these sessions were similar to regular weekly sessions (17% - 1/6) where the students could also

ask their technical questions and report team issues to the TAs and lecturers. The TAs also believed that these sessions could have been better designed, with different assessment arrangements such as assigning some marks to these sessions (33% - 2/6).

4.2.3. Suggestions for GitLab Metrics

Table 6 and Table 7 show the metrics suggested by the students and the TAs respectively where these are sorted according to the percentage of the participants who mentioned them. 95% (19/20) of the students and 50% (3/6) of the TAs suggested including other GitLab metrics. Students were specifically asked about this, and it was thus expected that a high percentage would have suggestions.

Table 6: GitLab metrics suggested by the students

Suggested Metrics	**% of Students**
LOC per commit/per member	65% (13/20)
Time spent per issue or overall	25% (5/20)
Longitudinal metrics with and without milestones	20% (4/20)
Metrics from qualitative data	15% (3/20)
Test coverage	10% (2/20)
Branches	10% (2/20)
Consideration of multiple contributors for issues	10% (2/20)
Pull requests	5% (1/20)
Commits per issue	5% (1/20)
Number of changed files	5% (1/20)

As illustrated in Table 6, the most popular suggestion from students was lines of code (LOC) added or deleted in each commit, or overall for each member. Although LOC was suggested by 65% (13/20) of the students, 30% (6/20) noted limitations of this. For example:

> *"Although percent just shows how many lines someone generated rather than how difficult or complex those lines were. So that's also not the best way to report statistics but that will be something useful."*

(Student - P16)

Time spent per issue or overall was also suggested by 25% (5/20) of the students, but 15% (3/20) of the students mentioned the difficulty of measuring the spent time:

> *"GitLab has \spend and \estimate function on it. So, you can track how long you think an issue is going to take versus how long it actually takes. [...] Useful but then again that's probably hard to track because they may not work on it the entire time, or they don't use the Git \spend functionality properly."* (Student - P20)

Other metrics suggested by at least two students were as follows: longitudinal metrics with and without milestones (such as issue-related operations over time including opening and closing issues), metrics from qualitative data (such as the informativeness of commit messages and the role assignment of different members for a particular issue based on the comments made for the issue), test coverage (i.e. the portion of the code that was tested), branch activities (such as when a branch is created, when a branch is merged to another branch etc.), and also the consideration of multiple contributors rather than just the assigned team member for an issue. Metrics suggested by only one student were as follows: pull requests, the number of commits per issue, and the number of changed files.

As illustrated in Table 7, only the time spent per issue was suggested by two TAs. Other metrics were suggested by only one TA, including merge operations (such as merge or pull requests, merge reviews), the number of code reviews per member (not specific to only merge operations), LOC per commit/per member, the quality of commit messages, test coverage, comparison of metrics between sessions (such as the comparison of the number of issues between week 5 and week 10), attendance of individual members instead of overall team attendance.

Some of these new metrics were suggested by both groups of participants while others were only mentioned by one of the groups. For example, LOC added

Table 7: GitLab metrics suggested by the TAs

Suggested Metrics	**% of TAs**
Time spent per issue	33% (2/6)
Merge operations	17% (1/6)
Number of code reviews per member	17% (1/6)
LOC per commit/per member	17% (1/6)
Quality of commit messages	17% (1/6)
Test coverage	17% (1/6)
Comparison of metrics between sessions	17% (1/6)
Attendance of individual members	17% (1/6)

or deleted in each commit or overall, time spent and test coverage were suggested by both TAs and students. However, consideration of multiple contributors for issue-related metrics was suggested only by the student group (10% - 2/20), and consideration of merge operations was suggested only by the TA group (17% - 1/6). Table 8 shows the closest matching between the metrics suggested by both the students and the TAs. The majority of the TAs who suggested new metrics also indicated that the report should be kept as simple as possible (66% - 2/3).

Table 8: The closest matching metrics suggested by both groups

Students	**TAs**
LOC per commit/per member	LOC per commit/per member
Time spent	Time spent per issue
Test coverage	Test coverage
Branches	Merge operations
Pull requests	
Metrics from qualitative data	Quality of commit messages

4.3. Comparative Analysis of Marks

The comparative analysis of the proportion of students receiving a pass mark in 2017/2018 and 2018/2019 shows that the association between cohort year and receiving a pass mark is statistically significant, with students in the

2018/2019 cohort more likely to receive a pass mark (219/228, 96.1%) than those in 2017/2018 (206/246, 83.7%), adjusted χ^2 (1, N = 474) = 4.0, p=0.045. This suggests that such consultation sessions may help improve student performance.

5. Discussion and Conclusion

5.1. Comparison with Related Studies

Existing studies suggest that activity metrics from software development platforms can allow us to understand how students work and help with the detection of problematic working behaviours. There are a limited number of studies investigating metrics for feedback purposes, which show that students were mostly satisfied with their use. As summarised in Table 9, these studies relied predominantly on quantitative student feedback, or an evaluation of the road-map of the course, rather than the use of consultation sessions and/or activity metrics directly.

We add to the literature via a qualitative research study with both students and TAs to develop a richer understanding of their opinions on the use of the activity metrics, specifically at consultation sessions. This was supplemented with student feedback, in which the students ranked their overall satisfaction with the process. Our study allowed us to ascertain that both students and TAs are primarily positive about the use of the activity metrics at consultation sessions, and to determine the benefits and drawbacks of the approach. In contrast to other studies, we also investigated whether the consultation sessions lead to better student outcomes by comparing the proportion of students who passed the course in two consecutive academic years, where one cohort had the consultation sessions and the other did not.

5.2. Discussion of Results

Our study shows that the students thought that the metrics provided an evidence-based mechanism for identifying team problems in a timely manner. They recognised the value of the metrics even if they did not directly experience

Table 9: Comparison of our study and other related studies

Comparison Category	**Comparison Items**	**Gary & Xavier [20]**	**Dietsch *et al.* [6]**	**Neyem *et al.* [21]**	**This Study**
Availability of Metrics	Continuous	✓			
	At Consultation Sessions		✓	✓	✓
Focus of Evaluation	Road-map		✓	✓	
	Consultation Sessions				✓
	Use of Metrics	✓			✓
Type of Evaluation	Quantitative Feedback	✓	✓		✓
	Qualitative Feedback				✓
	Comparative Analysis of Marks				✓
Participants in Evaluation	Students	✓	✓		✓
	TAs				✓

problems within their own teams. The TAs found the metrics to be useful for structuring their feedback to students. Both the students and the TAs believed that the checkpoint sessions helped students to improve their working practices, especially engagement. The importance of engagement was illustrated in a study conducted by Kay et al. [16], who found that the best performing group in their dataset had more sessions where at least half of the group members worked together.

The study also identified drawbacks to the checkpoints in their current form. For example, both students and TAs mentioned that the available metrics were not always reflective of the work distribution amongst team members. The number of commits in particular was not considered an appropriate indicator of the level of contributions of a team member by the majority of the students. The reason they gave for this was that the amount of work contained in any given commit can vary enormously.

Even though the activity metrics may have drawbacks, they provide insights

into student working behaviours and differences between team members and therefore trigger discussions between TAs and students. In particular, when students have only a few commits, they have the opportunity to justify their contribution, or commit to improving in the future. Hence, the discussions triggered by the activity metrics may lead to improvements in student working practices.

The students felt the sessions could be improved if they were longer in duration, although it should be noted that the workload of a course with consultation sessions was found to be high in a study conducted by Dietsch et al. [6]. These drawbacks should be taken into consideration by those planning to use contributor activity metrics as part of student feedback. Alternative metrics were suggested by both the students and the TAs. LOC, the metric suggested by the highest proportion of students (65% - 13/20), has also been found to be the strongest predictor of student attainment by Mierle et al. [9]. Those suggested by the students focused on achieving a more accurate assessment of individual contributions. This indicates that accurate assessment of their contribution is important to students and may influence their satisfaction with checkpoint sessions.

To investigate whether the consultation sessions helped students to perform better, we compared the proportion of students achieving a pass mark (a mark of $> 40\%$) in 2017/18, where there were no checkpoint sessions, and 2018/19, where there were two. Our analysis shows that the proportion of students passing was significantly higher in the academic year with the checkpoint sessions. The use of the metrics may lead to an improvement in performance – predominantly by providing an opportunity to identify team problems (including poor engagement) early, and potentially by improving software engineering skills. However, it is not possible to verify this with the current data, and further study is thus required. Furthermore, the use of checkpoints themselves may also increase the proportion of students achieving a pass mark, in accordance with previous work that found the use of weekly monitoring in software engineering courses was beneficial to students [22]. However, we did not evaluate

the use of checkpoints with and without GitLab metrics, so cannot isolate their effects. Possible future work could investigate how students' marks are affected by presenting the GitLab metrics continuously and allowing them to track their progress.

While we were interviewing our participants, we discovered that some of the students did not fully understand the metrics used during the consultation sessions. The provision of contributor activity metrics within software engineering courses should be accompanied by definitions and suggested interpretations.

5.3. Limitations and Future Work

The study has a number of limitations. It was conducted for a software engineering course at a particular university. Students were self-selected for the study, and may not have been representative. Although the academic years were equivalent in terms of course content, the cohorts consisted of different students, and potentially other latent differences, which may also have affected marks. Besides, as the proportion of students in larger teams (7 rather than 5 or 6 members) declined from 2017/2018 to 2018/2019, the change in the proportion of students receiving a pass mark from 2017/2018 to 2018/2019 may also be attributable to the change in the distribution of team sizes between the two years. Nevertheless, our qualitative analysis adds to a growing body of work examining the use of software platform metrics in learner analytics.

The confounding effects caused by differences between academic years on the evaluation of teaching innovations are a common issue in computer science education research [31]. However, it is ethically problematic to test teaching techniques by splitting students on the same course into different groups, as it risks one group being disadvantaged [31], [32]. Another problem with evaluating teaching innovations in practice is that they may introduce inefficiency to the teaching process [33]. There are a number of previous studies that show that consultation sessions and/or the use of the activity metrics for feedback purposes tends to be helpful to students [20], [6], [21], and therefore the risk of introducing inefficiency using our proposed approach was low.

One of the possible future directions is to use the report of the GitLab metrics at two checkpoint sessions in two consecutive academic years and cross-check the findings. In addition, student repositories can also be analysed in the future to understand how these sessions affect their behaviours over time while working on a project within a team, such as how their commits patterns are affected by these sessions. Future analysis will also look at the effect of checkpoint sessions by examining student repositories, and considering, for example, whether they affect the way work is distributed. It would also be interesting to examine the effects of additional GitLab metrics.

Open Data

All materials used in our interview study including the information sheets for both the students and TAs and the consent form are available in our external repository at Zenodo [34]. The list of the relevant quotations along with their corresponding themes are also available in our external repository.

Acknowledgements

This work was supported by the Institute of Coding which received £20m of funding from the Office for Students (OfS), as well as support from the Higher Education Funding Council for Wales (HEFCW). We would also like to thank our participants for their time in participating in this study.

Conflict of Interest

No competing interests exist.

References

[1] Y. Liu, E. Stroulia, K. Wong, D. German, Using cvs historical information to understand how students develop software, in: Proceedings of International Workshop on Mining Repositories (MSR 2004) at the 26th International Conference on Software Engineering, 2004, pp. 32 – 36.

[2] L. Glassy, Using version control to observe student software development processes, J. Comput. Sci. Coll. 21 (3) (2006) 99–106.

[3] M. Mittal, A. Sureka, Process mining software repositories from student projects in an undergraduate software engineering course, in: Companion Proceedings of the 36th International Conference on Software Engineering, ICSE Companion 2014, ACM, New York, NY, USA, 2014, pp. 344–353. doi:10.1145/2591062.2591152.

[4] C. Matthies, R. Teusner, G. Hesse, Beyond surveys: Analyzing software development artifacts to assess teaching efforts, in: 2018 IEEE Frontiers in Education Conference (FIE), 2018, pp. 1–9. doi:10.1109/FIE.2018.8659205.

[5] L. Kindberg, N. Jagelid, Examination of the effects of student-teacher interactions on student commit patterns - in the github environment, Bachelor's thesis, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology (2018).

[6] D. Dietsch, A. Podelski, J. Nam, P. M. Papadopoulos, M. Schäf, Monitoring student activity in collaborative software development, arXiv preprint arXiv:1305.0787.

[7] F. Putra, H. Santoso, R. Aji, Evaluation of learning analytics metrics and dashboard in a software engineering project course, Global Journal of Engineering Education 20 (3) (2018) 171–180.

[8] F. Koetter., M. Kochanowski., M. Kintz., B. Kersjes., I. Bogicevic., S. Wagner., Assessing software quality of agile student projects by data-mining software repositories, in: Proceedings of the 11th International Conference on Computer Supported Education - Volume 2: CSEDU,, INSTICC, SciTePress, 2019, pp. 244–251. doi:10.5220/0007688602440251.

[9] K. Mierle, K. Laven, S. Roweis, G. Wilson, Mining student cvs repositories

for performance indicators, SIGSOFT Softw. Eng. Notes 30 (4) (2005) 1–5. doi:10.1145/1082983.1083150.

[10] D. Petkovic, M. Sosnick-Pérez, S. Huang, R. Todtenhoefer, K. Okada, S. Arora, R. Sreenivasen, L. Flores, S. Dubey, Setap: Software engineering teamwork assessment and prediction using machine learning, in: 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, 2014, pp. 1–8. doi:10.1109/FIE.2014.7044199.

[11] D. Petkovic, M. Sosnick-Pérez, K. Okada, R. Todtenhoefer, Shihong Huang, N. Miglani, A. Vigil, Using the random forest classifier to assess and predict student learning of software engineering teamwork, in: 2016 IEEE Frontiers in Education Conference (FIE), 2016, pp. 1–7. doi:10.1109/FIE.2016.7757406.

[12] D. Petkovic, Using learning analytics to assess capstone project teams, Computer 49 (1) (2016) 80–83. doi:10.1109/MC.2016.3.

[13] D. Petkovic, S. H. Barlaskar, J. Yang, R. Todtenhoefer, From explaining how random forest classifier predicts learning of software engineering teamwork to guidance for educators, in: 2018 IEEE Frontiers in Education Conference (FIE), 2018, pp. 1–7. doi:10.1109/FIE.2018.8659102.

[14] A. M. Guerrero-Higueras, V. Matellán-Olivera, G. Esteban-Costales, C. Fernández-Llamas, F. J. Rodríguez-Sedano, M. A. Conde, Model for evaluating student performance through their interaction with version control systems, in: Proceedings of the Learning Analytics Summer Institute Spain 2018, León, Spain, 2018, pp. 104–112.

[15] S. Xavier, Continuous assessment in agile learning using visualizations and clustering of activity data to analyze student behavior, Master's thesis, Arizone State University (2016).

[16] J. Kay, N. Maisonneuve, K. Yacef, O. Zaïane, Mining patterns of events in students' teamwork data, in: Proceedings of the Workshop on Educational

Data Mining at the 8th International Conference on Intelligent Tutoring Systems (ITS 2006), 2006, pp. 45–52.

[17] W. Poncin, A. Serebrenik, M. van den Brand, Mining student capstone projects with frasr and prom, in: Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion, OOPSLA '11, ACM, New York, NY, USA, 2011, pp. 87–96. doi:10.1145/2048147.2048181.

[18] L. Baumstark, Jr., M. Orsega, Quantifying introductory cs students' iterative software process by mining version control system repositories, J. Comput. Sci. Coll. 31 (6) (2016) 97–104.

[19] C. Jones, Using subversion as an aid in evaluating individuals working on a group coding project, J. Comput. Sci. Coll. 25 (3) (2010) 18–23.
URL http://dl.acm.org/citation.cfm?id=1629116.1629122

[20] K. A. Gary, S. Xavier, Agile learning through continuous assessment, in: 2015 IEEE Frontiers in Education Conference (FIE), 2015, pp. 1–4. doi:10.1109/FIE.2015.7344278.

[21] A. Neyem, J. I. Benedetto, A. F. Chacon, Improving software engineering education through an empirical approach: Lessons learned from capstone teaching experiences, in: Proceedings of the 45th ACM Technical Symposium on Computer Science Education, SIGCSE '14, ACM, New York, NY, USA, 2014, pp. 391–396. doi:10.1145/2538862.2538920.
URL http://doi.acm.org/10.1145/2538862.2538920

[22] M. Marques, S. F. Ochoa, M. C. Bastarrica, F. J. Gutierrez, Enhancing the student learning experience in software engineering project courses, IEEE Transactions on Education 61 (1) (2018) 63–73. doi:10.1109/TE.2017.2742989.

[23] S. Jarzabek, Teaching advanced software design in team-based project course, in: 2013 26th International Conference on Software Engineering

Education and Training (CSEE T), 2013, pp. 31–40. doi:10.1109/CSEET.2013.6595234.

[24] S. Chacon, B. Straub, Pro Git, 2nd Edition, Apress, Berkely, CA, USA, 2014.

[25] B. O'Brien, I. Harris, T. Beckman, D. Reed, D. Cook, Standards for reporting qualitative research: A synthesis of recommendations, Academic Medicine 89 (9) (2014) 1245–51. doi:10.1097/ACM.0000000000000388.

[26] V. Braun, V. Clarke, Using thematic analysis in psychology, Qualitative Research in Psychology 3 (2) (2006) 77–101. doi:10.1191/1478088706qp063oa.

[27] A. Donner, Statistical methods in ophthalmology: An adjusted chi-square approach, Biometrics 45 (2) (1989) 605–611.
URL http://www.jstor.org/stable/2531501

[28] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2018).
URL https://www.R-project.org/

[29] Lesnoff, M., Lancelot, R., aod: Analysis of Overdispersed Data, r package version 1.3.1 (2012).
URL https://cran.r-project.org/package=aod

[30] A. J. Viera, J. M. Garrett, et al., Understanding interobserver agreement: the kappa statistic, Fam med 37 (5) (2005) 360–363.

[31] C. Holmboe, L. McIver, C. E. George, Research agenda for computer science education, in: Proceedings of the 13th Workshop of the Psychology of Programming Interest Group, PPIG 2001, Bournemouth, UK, 2001, pp. 207–223.

[32] D. G. Feitelson, Using students as experimental subjects in software engineering research - a review and discussion of the evidence, ArXiv abs/1512.08409.

[33] J. Carver, L. Jaccheri, S. Morasca, F. Shull, Issues in using students in empirical studies in software engineering education, in: Proceedings. 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (IEEE Cat. No.03EX717), 2003, pp. 239–249.

[34] S. Eraslan, K. Kopec-Harding, C. Jay, Interview Study About Coursework Consultation Sessions with GitLab Metrics in a Software Engineering Course (Oct. 2019). doi:10.5281/zenodo.3521755.
URL https://doi.org/10.5281/zenodo.3521755