Computer Science Education

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ncse20

Assessing individual contributions to software engineering projects: a replication study

C.D. Hundhausen, P.T. Conrad, A.S. Carter & O. Adesope

To cite this article: C.D. Hundhausen, P.T. Conrad, A.S. Carter & O. Adesope (2022) Assessing individual contributions to software engineering projects: a replication study, Computer Science Education, 32:3, 335-354, DOI: 10.1080/08993408.2022.2071543

To link to this article: https://doi.org/10.1080/08993408.2022.2071543

Published online: 17 May 2022.

COMPUTER SCIENCE EDUCATION
2022, VOL. 32, NO. 3, 335–354
https://doi.org/10.1080/08993408.2022.2071543

Assessing individual contributions to software engineering projects: a replication study

C.D. Hundhausen[a], P.T. Conrad[b], A.S. Carter[c] and O. Adesope[d]

[a]School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, USA; [b]Department of Computer Science, University of California Santa Barbara, Santa Barbara, CA, USA; [c]Department of Computer Science, Humboldt State University, Arcata, CA, USA; [d]Educational Psychology Program, College of Education, Washington State University, Pullman, WA, USA

ABSTRACT
Background and Context: Assessing team members' indivdiual contributions to software development projects poses a key problem for computing instructors. While instructors typically rely on subjective assessments, objective assessments could provide a more robust picture. To explore this possibility, In a 2020 paper, Buffardi presented a correlational analysis of objective metrics and subjective metrics in an advanced software engineering project course (n= 41 students and 10 teams), finding only two significant correlations.
Objective: To explore the robustness of Buffardi's findings and gain further insight, we conducted a larger scale replication of the Buffardi study (n = 118 students and 25 teams) in three courses at three institutions.
Method: We collected the same data as in the Buffardi study and computed the same measures from those data. We replicated Buffardi's exploratory, correlational and regression analyses of objective and subjective measures.
Findings: While replicating four of Buffardi's five significant correlational findings and partially replicating the findings of Buffardi's regression analyses, our results go beyond those of Buffardi by identifying eight additional significant correlations.
Implications: In contrast to Buffardi's study, our larger scale study suggests that subjective and objective measures of individual performance in team software development projects can be fruitfully combined to provide consistent and complementary assessments of individual performance.

ARTICLE HISTORY
Received 12 November 2020
Accepted 26 April 2022

KEYWORDS
Team software development projects; individual contributions; GitHub log data analysis; peer evaluation; issues; commits; user stories

Introduction

Team software development projects are common in undergraduate computing education. While team projects are most often featured in capstone design courses (e.g. Herbert, 2018; Murphy, Sheth et al., 2017), they also appear in a variety of upper and lower division computing courses that involve software development (e.g. Saltz & Heckman, 2018). Such projects are seen as a crucial component of undergraduate

CONTACT C.D. Hundhausen hundhaus@wsu.edu Human-centered Environments for Learning and Programming (HELP) Lab, School of Electrical Engineering and Computer Science, Washington State University, P.O. Box 642752 Pullman, WA 99164,

computing education. Studies of software developers in industry underscore the need for undergraduates to acquire *authentic* software development experiences in which they develop solutions to realistic problems within collaborative software development contexts (see, e.g. Begel & Simon, 2008; Craig et al., 2018; Exter, 2014; Hewner & Guzdial, 2010; Sherriff & Heckman, 2018).

Despite their important educational value, team software development projects prove challenging for computing educators to implement. Among the many barriers to implementation, the assessment of individual contributions within team projects proves particularly challenging. While some instructors choose to assign all team members the same grades, this approach may be seen as unfair since it fails to acknowledge the reality that a team's work products are the result of varying contributions by individual team members (Clear, 2009; Farrell et al., 2012; Tafliovich et al., 2015).

Acknowledging the need to be sensitive to individual team members' contributions within team projects, computing educators have proposed a variety of assessment approaches. Most of these are *subjective*, integrating peer and instructor assessment of team member contributions, and instructor assessment of individual and team deliverables (e.g. Domínguez et al., 2020; Farrell et al., 2012; Tafliovich et al., 2015; Von Konsky & Ivins, 2008). With the advent of online collaborative software development platforms such as GitHub (Build software better, together, n.d.), there is an opportunity to augment these subjective assessment approaches with objective data on students' contributions to team projects (Kalliamvakou et al., 2014). These objective data include *code commits* (instances of adding new code to a code repository), *line contributions* (counts of lines added and removed from a code repository), and *issue tracking contributions* (instances of creating, tracking, and closing work items).

Citing the "halo effect" (i.e. the idea that one's overall impression of a person creates an assessment bias) observed in instructors' and students' assessments of individual student work, Buffardi (2020) performed an exploratory analysis that compared a set of standard subjective assessments of individual student work against a set of objective metrics derived from the above activities on GitHub. While his study found strong correlations between the subjective assessments and between the objective metrics, the correlation between subjective assessments and objective metrics was weak. Buffardi's study lays the groundwork for future work by proposing a methodology for comparing subjective assessments and objective metrics, and by providing evidence that current subjective assessment approaches may be biased and could benefit from triangulation with objective assessment approaches.

We are particularly interested in two research questions raised by the Buffardi study:

RQ 1. Do Buffardi's results generalize to other software development courses at different institutions?

RQ 2. What additional findings beyond Buffardi's can be gleaned from a larger scale study conducted at multiple institutions?

In this paper, we explore these questions through a replication of the Buffardi study. Our study focuses on team software development projects in three courses at three institutions:

(1) An upper-division web development course (n = 57 students and 12 teams) at a large research university;
(2) An upper-division mobile application development course (n = 5 students and 2 teams) at a mid-sized undergraduate university; and
(3) An upper-division software projects course (n = 57 students and 11 teams) at a large research university.

To address our research questions, we collected and analyzed a set of data in our courses equivalent to the data collected in Buffardi's study. While replicating all but one of Buffardi's major findings, our results identify eight correlations between objective and subjective measures that Buffardi's work did not find. Thus, our larger scale study contributes not only a near complete replication of Buffardi's study, but also new evidence that subjective and objective metrics of individual performance in team software development projects can furnish consistent and complementary assessments.

Related work

The related work cited in the original study (Buffardi, 2020) covers the general context of incorporating industrial software development practices into software engineering courses, and the more specific task of assessing the individual contributions of students who are members of teams. Buffardi opens the argument for including project-based software engineering work in the computing curriculum by noting its inclusion in the (ACM/IEEE Joint Task Force on Computing Curricula, 2013) CS curriculum guidelines (ACM/IEEE Joint Task Force on Computing Curricula, 2013), and noting several models for such courses, including working on open source projects in general (G. G. Pinto et al., 2019; G. H. L. Pinto et al., 2017; Smith et al., 2014); working specifically on humanitarian open source projects (Braught et al., 2018; Ellis & Hislop, 2016; Murphy, Buffardi et al., 2017); and working with industry and entrepreneurial partners (Bruegge et al., 2015; Buffardi et al., 2017; Clark et al., 2005; Devadiga, 2017; Garousi et al., 2016; Mead, 2015; Parker & Holcombe, 1999). Grading a team as a whole avoids the problem of assessing individual contributions. As reported by Tafliovich et al. (2015), as students progress in their study and within a course, their preference shifts from being evaluated as individuals to being evaluated as a team.However, as Buffardi notes, the problem of "social loafing", as described by Latané et al. (1979), presents a challenge to fairness if team members receive the same grade for widely disparate contributions (or even for no contributions). Gauging student performance is not the only reason to want to assess individual contributions; we are also interested in assessing the *effectiveness* of teaching about teamwork. If a single individual does the lion's share of the work, then group work not only fails to teach teamwork skills that are vital for students to acquire and use in the industry; it may even be counterproductive.

Peer assessment provides one source of data for assessing individual contributions (Fagerholm & Vihavainen, 2013). Like Buffardi, we used CATME, a standard peer evaluation instrument (Layton et al., 2007; Ohland et al., 2005, 2012), to elicit peer evaluations in our study. However, using peer assessment alone may be unwise, as it is subject to various biases. Herbert (2007) found that peer assessments can be inaccurate, and are subject to biases against women and students from underrepresented groups. Another bias in peer assessments is the so-called "halo effect", whereby raters are influenced to give higher ratings to individuals with positive characteristics that may be unrelated to the characteristics being assessed (Abikoff et al., 1993; Schneider et al., 2011). Instructor assessment is also subject to such biases (Van den Bergh et al., 2010).

To supplement peer evaluations and provide a cross check, Buffardi collected GitHub log data and analyzed four *objective* artifacts of the software engineering lifecycle captured through GitHub log data:

- *Commits*, in which the team's code repository is updated with code changes
- *Line changes*, which quantify the number of lines of code that are changed
- *Issues*, which teams use to describe, and track completion of, "user stories" (Rehkopf, n.d.) and other software development tasks. Notably, GitHub issues (About issues, n.d.) can be explicitly assigned to the team members responsible for completing them.
- *Story points* (Radigan, n.d.), an integer representing a team's assessment of the magnitude or difficulty of completing an issue or user story.

Our study largely replicates Buffardi's approach to collecting and analyzing these artifacts.

As a potential threat to the validity of assessing individual student work based on GitHub data, Buffardi cites *Campbell's Law*: the principle that assessing quantitative measures can cause students to modify their behavior to optimize it for the measurement, even if that behavior is contrary to the overall project or learning goals (Campbell, 1979). Buffardi also notes that negative effects of performance appraisals in Agile software development organizations have been reported by industry practitioners of Agile (e.g. Sutherland, 2018). Like Buffardi, we are mindful of this pitfall in our interpretation of the study's results.

Another potential threat to the validity of this approach not mentioned by Buffardi is that students could misrepresent their individual contributions to the GitHub issues completed by their team. To credit individual students for their work, Buffardi looked at the students to whom the issue was *assigned* in GitHub. However, the team members assigned to a given issue may or may not correspond to the team members who contributed to completing the issue. Indeed, team members are free to assign other team members to an issue who did not contribute to it. To replicate Buffardi's study, our study also credits student contributions based on who is assigned to issues in GitHub, while acknowledging that this approach may not accurately credit the students who did the work.

Methods

Courses and Team Projects

Whereas Buffardi collected data from a single course at a single institution, our replication study collected data from three courses taught by three instructors at three universities. Table 1 presents an overview of the courses and team projects considered in this study, including the original course studied by Buffardi. Henceforth, we will use the university abbreviations shown in this table (Chico, WSU, UCSB, and HSU) to refer to the courses at these institutions included in this study.

The team projects had differing weights in each course's overall grading scheme. In the Chico course, the project was weighted 60%, with each of six sprints weighted equally. Likewise, in the WSU course, the project was weighted 20%, with each of four sprints weighted equally. The UCSB and HSU course projects were both weighted 40% of the overall grade. At UCSB, students could fulfill the project requirement by completing 100 total story points in any combination of the three sprints. The four HSU sprints were weighted 10% each.

Another key difference across the course projects lies in whether they were graded individually or as a team. In the Chico and UCSB projects, all members of a given team were assigned the same grade for each sprint. In the WSU projects, all members of a given team were assigned the same grade for each sprint, but an individual multiplier derived from the CATME peer evaluation surveys was applied to team grades to obtain individual grades. In the HSU projects, the instructor assigned an individual grade to each student based on the perceived quantity and quality of their contribution to the project.

A third difference among the course projects can be found in their grading schemes, which are summarized in Table 2. As the table suggests, the Chico and WSU courses had one grading criterion (*software testing*) in common, while the HSU and WSU courses shared five common grading criteria. The UCSB project grading scheme was unique.

Participants

Table 3 presents counts of, and demographic data on, the participants in the study on a course-by-course basis. The counts include both the total number of students and project teams in each course, and the total number of students and project teams who consented to share their data for research purposes. Note that because peer evaluations constituted one of the forms of data, we could not use any data for a team unless *all* of its members gave us informed consent; a single non-consenting student meant that the data for that entire team had to be excluded from the study. This resulted in a low rate of consent at the team level for the UCSB offerings. Given the relatively low number of consenting students in the UCSB courses, and the fact that the courses were run in a consistent manner across the three offerings, we collapsed the data from the three UCSB courses into a single "UCSB" course for data analysis purposes.

Table 1. Overview of courses involved in the original and replication study.

University	Type	Course Number and Name	Project Length (weeks)	Number of Sprints
Original: Cal State University Chico (Chico)	Medium Undergrad	CSCI 430 ("Software Engineering")	12	6
Washington State University (WSU)	Large Research	CptS 489 ("Web Development")	5	4
University of California Santa Barbara (UCSB)	Large Research	CMPSC 156 ("Advanced Application Development")	3–4	3
Humboldt State University (HSU)	Medium Undergrad	CS 480 ("Modern Software Development")	8	4

Table 2. Comparison of project grading schemes in courses studied.

	Weight (%) in Project Grading Scheme			
	Original		Replication	
Grading Criteria	Chico	WSU	UCSB	HSU
Software usefulness	30			
Software design	35			
Software testing	35	10		
Software quality		20		
Code quality		10		25
Development progress (in sprint)		20		25
Development process: Branches, Commits, PRs, Code Review		15		20
Development process: Tracking, communication, coordination		15		10
Development process: Team retrospective		10		20
Completion of 100 story points			100	
Total:	**100**	**100**	**100**	**100**

Table 3. Number of students and teams in each course offering.

				# Students consenting					
Course	Term	# Students enrolled	# Teams enrolled	Total	M	F	U	M Age	# Teamsconsenting
Original: Chico	F18	2	10	42	–	–	–	–	10
Replication: HSU	S21	12	2	5	4	1	0	23.2	2
WSU	F20	57	12	57	48	4	5	23.3	12
UCSB	F20	66	12	37	29	8	0	19.9	7
	W21	54	10	9	9	0	0	20.7	2
	S21	64	12	11	8	3	0	19.7	2
	UCSB Total	186	34	57	46	11	0	20.0	11
Replication Totals		**253**	**48**	**119**	**98**	**16**	**5**	**21.7**	**25**

Note: M = Males, F = Females, U = Unknown or did not report, M Age = Mean Age

Data collection and measures

Buffardi's (2020) study explored correlations among individuals' CATME peer evaluations and course grades (what Buffardi termed "subjective" measures), and "objective" measures derived from data gathered through Github. Our replication study considered the same data and associated measures, but differed in notable ways, as summarized in Table 4. Below, we describe the data and measures in further detail.

Demographic data. To obtain a fuller picture of our participants, we collected demographic data through a presurvey, including participants' gender, ethnicity, and age. Buffardi did not collect these data in his study.

CATME Peer Evaluation Survey. Buffardi administered the CATME peer evaluation survey (catme.org, 2021) in weeks 5, 10, and 15 of his team project. The averages of the *Peer Contribution* and *Peer Interaction* ratings in the survey responses were used in his correlational data analysis. Like Buffardi, our correlational analysis used the average of the *Peer Interaction* and *Peer Interaction* ratings (catme.org, n.d.) provided in multiple survey responses. However, the number of surveys administered, and the time intervals at which we administered the surveys, differed by course. In

Table 4. Data sources and associated measures collected in study.

		Original	Replication		
Data Source	Description and Associated Measures	Chico	WSU	UCSB	HSU
Demographic Survey	Online pre-survey eliciting student demographic information: • Gender • Ethnicity • Age	◎	●	●	●
CATME Peer Evaluation Survey	Peer evaluation survey administered periodically during the project. Two ratings were considered:				
	• *Peer Contribution*: Average peer rating of a team member's contributions to the team's work on a five-point Likert-style scale.	●	●	●	●
	• *Peer Interaction*: Average peer rating of the quality and quantity of a team member's interactions on a five point Likert scale.	●	●	●	●
Course Grades	• *Project grades*: The grade received by the team on the team project (see, Table 2 for grading schemes)	◐	◐	◎	◐
	• *Individual grades*: Grades received by students on individual course assignments outside team project.	○	○	○	◎
GitHub	Log data from teams' GitHub repositories were used to derive the following measures for each team member:				
	• *Commit Counts*: Number of individual commits merged into the main branch of the team's repository.	●	●	●	●
	• *Relative Commit Shares*: Number of individual merged commits relative to *expected commits* (i.e. total team commits ÷ # team members)	●	●	●	●
	• *Change Counts*: Number of lines of code changed by each individual in merged commits	●	●	●	●
	• *Relative Change Shares*: Number of individual line changes relative to *expected line changes* (i.e. total team line changes ÷ # team members)	●	●	●	●
	• *Story Counts*: Number of completed GitHub issues (i.e. "user stories") assigned to each team member	◐	●	●	●
	• *Relative Story Shares*: Number of individual issues relative to *expected issues* (i.e. total team issues completed ÷ # team members).	◐	●	●	●
	• *Point Counts*: Sum of each team member's portion of story points assigned to each completed issue	○	◎	○	◎
	• *Relative Point Shares*: Number of individual story points relative to expected story points (i.e. total team story points ÷ # team members)	○	◎	○	◎

table legend:

●	*Data measure exactly matches corresponding data measure in at least one other course*
◐	*Data measure partially matches corresponding data measure in at least one other course*
○	*Data measure does not match corresponding data measure in any other courses*
◎	*Data measure not collected or could not be used in the analysis*

the WSU and HSU courses, four surveys were administered, while the UCSB course included three surveys. In all courses, the surveys were administered after each sprint cycle.

Individual grades. Buffardi used the average of four individual quizzes as a basis for gauging individual student knowledge and skills. We could not replicate this exactly, since none of our courses included quizzes. Instead, as available, we used the average of individual assignments given in each course: eight individual web programming assignments in the WSU course and the best six of eight individual homework assignments in the UCSB course. The HSU course did not include any individual assignments – only the course project.

Project grades. As indicated in Table 2, the way in which project grades were calculated varied considerably across courses. As noted in Table 4, there was a partial match between the Chico and WSU courses, and a more complete match between the HSU and WSU courses. However, the project grading scheme for the UCSB course was unique: all students who completed a set number of story points received 100%. In fact, just about all students met this threshold, making the UCSB project grade metric ineffective in discriminating student performance. For this reason, we did not include UCSB project grades in our correlational analyses.

Commit and line change counts. Like Buffardi, we counted commits and line changes that were ultimately merged into a software repository's default branch (e.g. main or master). Like Buffardi, we excluded from our counts commits and line changes associated with initializing a team's source repository with base code drawn from another source, as these tended to be outliers that, if not excluded, would artificially inflate the contribution of the student making such commits. We also excluded object files that were checked into the version control system. In our context, the excluded files were package-lock.json, server.compiled.js, and all files under client/build. We mention this detail to alert researchers seeking to replicate our study that they may need to devise similar exclusion criteria for object files appropriate to the framework(s) used in their courses.

Story counts. To receive credit for completing issues (stories) in Buffardi's course, students were asked to move completed GitHub *issues* (GitHub.com, 2021a) into the "Done" column of their project (Kanban) boards (About issues, Github.com, 2021b). In contrast, students in our courses received credit for completing issues when they linked the issues to *pull requests* (GitHub.com, 2021b and 2021c) that merged commits into the team repository's main branch. Given that our approach to crediting teams for completed stories differed from Buffardi's, our method for counting stories necessarily differed as well: Whereas Buffardi counted a story as "complete" only if it appeared in the "Done" column, we counted a story as complete only if it could be associated with code changes merged into a project's main branch. For replication purposes, the important point is that the story counting method was aligned with the definition of "done" established in each course.

Story point counts. Buffardi asked student teams to assign *story points* (Radigan, n.d.) to each completed issue. Those story points were then divided equally among all team members assigned to the issue. Unfortunately, as Buffardi found in his study, student teams do not reliably assign story points to issues; he had to exclude two teams from his story point analysis because they failed to assign story points. In the WSU and HSU courses, student teams proved even less reliable: Not one team assigned story points to all completed issues. For this reason, we did not include WSU and HSU story point shares in our correlational analyses.

In contrast, in the UCSB courses, the responsibility for assigning story points was delegated to course teaching personnel, who assigned points to *every* completed issue as it was code reviewed and merged by course staff (instructors and TAs). This approach provided a more consistent and reliable means of assigning story points, allowing us to include UCSB story point shares in our correlational analysis.

Preregistration

This study was preregistered with the journal. A preliminary version of the study methods was reviewed before the study was carried out. Subsequent to this review, we made the following changes to the study methods.

- We originally estimated the number of participants to be 221 students and 42 teams. Because not all students in the courses included in the study consented to participate, the number of participants fell to 119 students and 25 teams.
- We originally proposed to develop a rubric to assess the quality of team's software, issues, and pull requests. The idea was to provide a richer mix of qualitative and quantitative measures for correlational analysis. This proposal proved to be beyond the scope of what we could do in the time allotted for the study, so we dropped it.
- We originally proposed to collect and analyze responses to a pre-/post-survey of student attitudes, and to incorporate these results into our correlational analysis. While we collected these data, we did not include them in the analysis presented in this paper because we wanted to keep the focus on replicating Buffardi's analyses.
- We originally identified three research questions related to the broader scope we envisioned for the study (see the previous two points). Because we opted to narrow the focus of this paper, we retained the first research question, but replaced the final two research questions with a single research question related to possible new and expanded findings from a larger replication study.

Results

Buffardi presented an exploratory analysis of his results, beginning with a visual inspection of trends with respect to team and individual GitHub measures, and then shifting to correlational analyses of objective (GitHub) and subjective (grades and CATME survey) measures. The analysis presented below mirrors Buffardi's analysis.

Summary of team activities in GitHub

Table 5 presents summary statistics on student teams' activities in GitHub for the original (Chico) course and the replication courses. As this table indicates, student teams' commits and stories completed were similar across all courses. However, we observed three notable differences:

- In the Chico course, which spanned an entire semester, there were substantially more line changes.
- In the UCSB and HSU courses, the number of stories completed was lower than in the other two courses.
- The number of commits, line changes, and stories completed was generally lower in the HSU course than in the other courses.

Table 5. Summary of team GitHub activity by course.

Course	# Students	# Teams	# Commits		# Line Changes		# Stories Completed	
			M	SD	M	SD	M	SD
Chico	42	10	90.1	40.7	95,402.9	141,945.4	14.0	6.5
WSU	57	12	78.7	41.7	3127.3	2051.6	16.5	7.9
UCSB	57	11	78.0	36.5	11,903.7	8280.4	5.7	3.1
HSU	5	2	27.6	29.1	895.6	818.1	1.6	1.3

Summary of GitHub, peer evaluation, and grade measures

We collected the same objective (GitHub) and subjective (CATME peer evaluation and grade) measures as Buffardi. To support visual exploration of our results, Figure 1 presents bar charts showing these measures for all 25 teams included in our study. In this figure, the 12 WSU teams are labeled W1–W12; the 11 UCSB teams are labeled Sf1–Sf7 (seven teams from the Fall 2020 course offering), Sw1–Sw2 (two teams from

the Winter 2021 course offering), and Ss1–Ss2 (two teams from Spring 2021 course offering); and the two HSU teams are labeled H1 and H2. Each team member's share or rating is represented by a bar whose height indicates its magnitude. Hollow dots represent team members whose shares were 0. Team member bars line up vertically to facilitate visual comparisons of teams and team members across the metrics. For example, as can be seen from the first group in Figure 1a, team W1 had five team members, three of whom contributed commits. Of the three who made commits, one team member made about twice as many commits as the other team members.

As discussed in the Methods section, we were unable to collect a complete set of data in all courses. This explains the missing values for the Relative Story Point Share (Figure 1d), Individual Grade (Figure 1g), and Project Grade (Figure 1h) measures.

Visual inspection of the bar charts in Figure 1 yields some notable observations:

- Nearly half of the teams (11 out of 25, or 44%) had at least one member who did not contribute any code at all. Three of the teams had *two* members who did not contribute any code.
- Over half of the teams (14 of 25, or 56%) had at least one *dominant* member who contributed over twice the expected number of commits.
- Roughly one third of the teams (9 of 25, or 36%) stood out for achieving some measure of equality when it came to contributing code, with all team members having commit and line change shares clustered below 2.0.
- Team W9 had a dominant team member who stood out for contributing three times the expected number of commits and line changes, and for receiving credit for *all* of the team's completed stories, despite the fact that other members of the W9 team also contributed commits and line changes.
- Commit and line share measures suggest that team labor tended to be unbalanced.

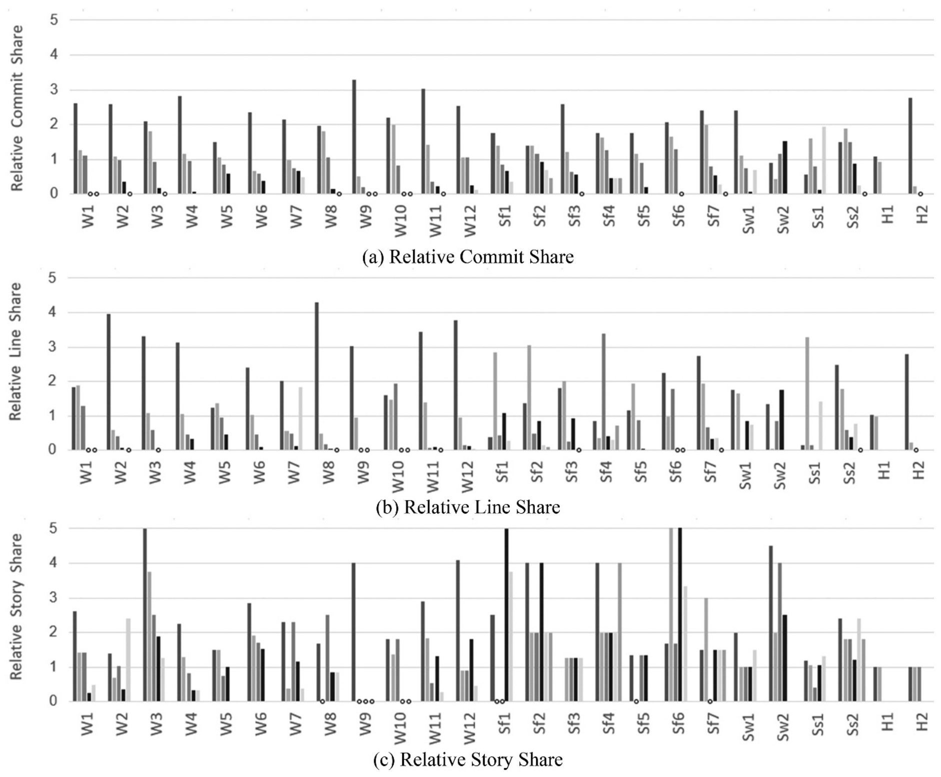


Figure 1. Graphical summary of objective (a-d) and subjective (e-h) measures by team and individual.

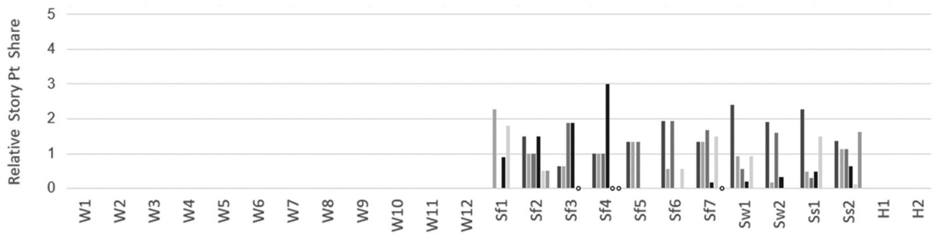
Correlational analysis

Mirroring Buffardi's study, we ran a correlational analysis of all subjective and objective measures for all courses using Pearson's correlation coefficient (r_p). Table 6 summarizes the results; the shaded section highlights the 16 correlations between objective and subjective measures. In this analysis, we set the threshold value for significance at $p < 0.05$. In addition, per the advice of Akoglu (2018), we interpret r_p

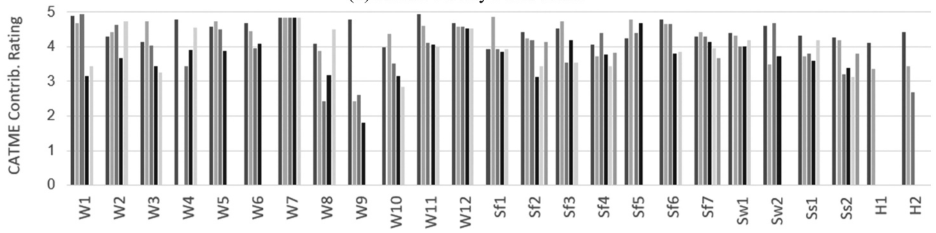
values of below 0.3 as *weak* correlations, r_p values between 0.3 and 0.6 as *moderate* correlations, and r_p values above 0.6 as *strong* correlations.

Mirroring Buffardi's analysis, we next conducted a multiple regression analysis to determine whether the (objective) GitHub metrics (Relative Commit Share, Relative Line Change Share, Relative Story Share, and Relative Story Point Share) predict the (subjective) CATME Peer Contribution Rating. In line with Buffardi's findings, our regression model was significant (*$F(2, 54) = 12.45$, $p < .001$, $R^2 = .32$, $R^2_{Adjusted} = .29$*). However, whereas Buffardi's analysis found that Relative Commit Share was the significant predictor in the model, we found that both Relative Line Change Share *(Beta = .51, $t(56) = 4.43$, $p < .001$)* and Relative Story Point Share *(Beta = .24, $t(56) = 2.03$, $p < .05$)* were the significant predictors.

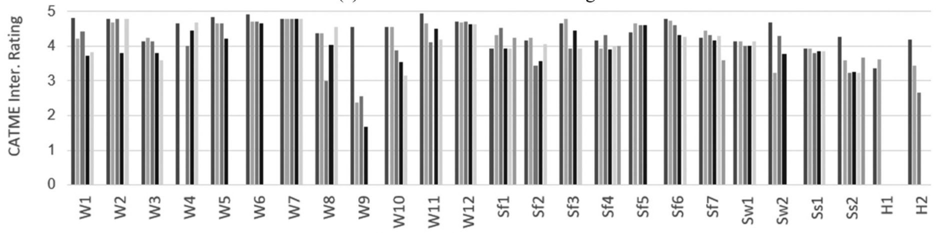
To complete our replication of Buffardi's analyses, we conducted a multiple regression analysis to examine if Relative Commit Share, Relative Line Change Share, Relative Story Share, CATME Peer Contribution Rating and CATME Interaction Rating predict Project



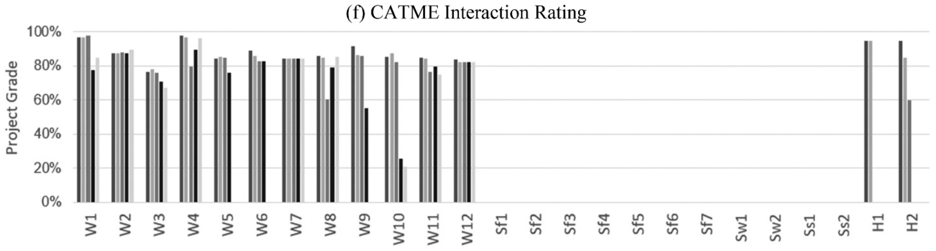
(d) Relative Story Point Share



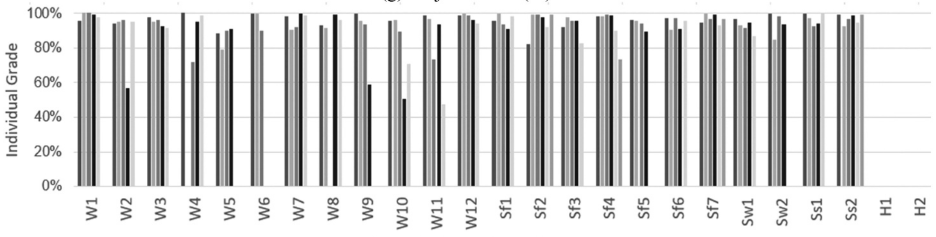
(e) CATME Contribution Rating



(f) CATME Interaction Rating



(g) Project Grade (%)



(h) Individual Grade (%)

Figure 1. Continued.

Grade. Again, as in the Buffardi study, our model was significant *($F(1, 60) = 15.94$, $p < .001$, $R^2 = .21$, $R^2_{Adjusted} = .20$)*. However, whereas Buffardi identified the (subjective) CATME Peer Contribution Rating as the significant predictor, we found that the (objective) Relative Commit Share significantly predicted project grade *($Beta = .46$, $t(61) = 3.99$, $p < .001$)*.

To address RQ 1, Table 7 summarizes the results of our correlational and regression analyses vis-a-vis Buffardi's findings. As the table illustrates, our study replicated all but one of Buffardi's five significant findings.

To address RQ2, we observe that Table 6 reveals several additional significant correlations that were not found in Buffardi's study. Whereas Buffardi identified just two significant correlations between the four (objective) GitHub metrics and the four subjective measures, our results identify statistically reliable, weak to moderate strength associations ($.19 \le r_p \le .46$) in 10 of the 16 possible objective-subjective variable pairs (see shaded area of Table 6)

- Relative Commit Share, Relative Line Change Share, and Relative Story Share are all significantly correlated with Peer Contribution Rating, Peer Interaction Rating, and Project Grade.
- Relative Story Point Share is significantly correlated with Individual grade.

Discussion

Our study replicated most of Buffardi's key results. It also detected more significant associations between subjective and objective metrics than Buffardi's study, identifying 10 of 16 (63%) of the possible associations. This was likely due to having a sample size nearly three times larger than Buffardi's, thus providing the statistical power necessary to be sensitive to these correlations. Prior to running our study, we conducted a power analysis using G*Power (Faul et al., 2007). Using a near moderate target effect size of 0.25, an alpha value of 0.05 and a power of 0.80, the analysis indicated that a sample size of 97 students would be needed to attain this effect size. As predicted, our sample size of 118 students proved adequate to detect statistically significant correlations of weak to moderate strength. It was likely the case that Buffardi's original sample size of 41 was simply not large enough to reliably detect these correlations. With a larger sample size, we suspect Buffardi would have detected more significant relationships among these variables.

When attempting to replicate a pedagogical approach such as team software development projects, educators need to be sensitive to *implementation fidelity* (Meyers & Brandt, 2015): the extent to which the pedagogical approach implemented in the replication course matches the approach implemented in the original course. Research shows

Table 6. Descriptive statistics and correlations for study variables.

Variable	M	SD	RCS	RLCS	RSS	RSPS	PCR	PIR	PG	IG
RCS	1.00	.813	–							
RLCS	1.00	1.054	.79**	–						
RSS	1.82	1.394	.37**	.42**	–					
RSPS	1.00	.742	.33*	.31*	.59**	–				
PCR	4.05	.724	.46**	.41**	.31**	.38	–			
PIR	4.13	.688	.33**	.29**	.19*	−.08	.92**	–		
PG	82.00	13.792	.46**	.37**	.27*	–	.38**	.29*	–	
IG	88.96	21.350	.09	.03	.14	.46**	.29**	.23*	.30*	–

Note: *M* and *SD* represent means and standard deviations. RCS = Relative Commit Share; RLCS = Relative Line Change Share; RSS = Relative Story Share; RSPS = Relative Story Point Share; PCR = CATME Peer Contribution Rating; PIR = CATME Peer Interaction Rating; PG = Project Grade; IG = Individual Grade. * indicates $p < .05$. ** indicates $p < .01$. Shaded region highlights correlations between objective and subjective variables.

Table 7. Summary of our replication of buffardi's main findings.

Buffardi Significant Result	Our Result	Replicated?
Relative Commit Share was significantly correlated with contribution rating	Relative Commit Share was significantly correlated with peer contribution rating ($p < .001$). The strength of the correlation was moderate ($r_p = .46$).	✓
Relative Commit Share was a significant predictor of Peer Contribution Rating	Relative Commit Share was **not** a significant predictor of Peer Contribution Rating (*Beta* = .20, *t*(56) = 1.29, *p* = .201). Instead, we found that Relative Line Change Share and Relative Story Point Share were significant predictors.	✕
Relative Story Share was significantly correlated with Project Grade	Relative Story Share was significantly correlated with Project Grade ($p = .002$). The strength of the correlation was moderate ($r_p = .39$).	✓
Peer Contribution Rating and Peer Interaction Rating were significantly correlated	Peer Contribution Rating was significantly correlated with Peer Interaction Rating ($p < .001$). The strength of the correlation was high ($r_p = .92$).	✓
Individual Grade and Project Grade were significantly correlated	Individual Grade was significantly correlated with project Grade ($p = .017$). The strength of the correlation was moderate ($r_p = .32$).	✓

that poor implementation fidelity can lead to inconsistent results when evaluating a pedagogical approach. While Buffardi's original course and the courses included in this replication study had in common a multi-week team software development project, there were many notable differences in the details of implementing the project in these courses. For example, Buffardi's project spanned an entire semester, whereas our replication projects spanned just a few weeks. Likewise, the evaluation criteria for the projects varied widely across implementations (see, Table 2), as did the implementation tasks in the projects themselves.

Despite the poor implementation fidelity of the software development projects considered in this study, we were still able to replicate most of Buffardi's significant results, as well as to identify an even larger set of significant correlations within three different courses at three different universities. The fact that these correlations were present in a sample that included different courses, universities, and project implementations is encouraging; it suggests that they may generalize to a broader population.

In addition, the fact that we were able to identify 10 of 16 possible significant correlations between subjective and objective measures runs counter to Buffardi's hunch that the "halo effect" (Schneider et al., 2011) may have played a role in teammates' ratings of each other. In our study, students' peer evaluations, as well as instructors' evaluations of students' project work, were well aligned with objective measures of students' software development activities. Given this, we wonder whether students might base their peer evaluations in part on an awareness of their teammates' GitHub activities. Indeed, further exploration of the basis of student perceptions of their teammates' contributions would be an interesting direction for future work.

Along these same lines, we wonder whether Buffardi's GitHub metrics provide *valid* measurements of a team member's contribution and value to a software team. One possibility that Buffardi did not discuss, but that seems relevant to this investigation, is that individuals could provide valuable contributions to a software project through activities that are not registered through GitHub. For instance, perhaps a team member takes a leadership role by coordinating daily check-in meetings, summarizing the results, and following up with team members regarding their progress. While these activities have value for the team, they may not take place in GitHub. This observation underscores the inherent limitations of using coding activities in GitHub as a means of measuring individual contributions. Clearly, we need to consider additional data, both inside of GitHub (e.g. issue writing and Kanban board activities), and outside of it (e.g. team communication and coordination).

Confounds and threats to validity

Our study is subject to several confounds and threats to validity:

- *Instructor bias*. The instructors of the courses involved in the study were also the authors of this paper. Since they were privy to the design of this research, their teaching of the course may have biased the results. Clearly, a better approach would be to study courses whose instructors are not involved in the design of the research.
- *Variations in pedagogy implementation*. As mentioned in the above discussion, our study considered different courses taught by different instructors at different institutions. There were significant variations in the ways in which the collaborative software development project was implemented across the courses. These variations contributed to differences in students' team project performance and collaboration, as well as to differences in the data we were able to collect in each course. One notable difference was that, in the UCSB course, the *teaching personnel* assigned story points to each issue, whereas the students themselves assigned points to issues in the WSU and HSU courses. As was the case in the original Buffardi study, we found that students often forgot to assign story points, leading to our inability to analyze the story point data in the WSU and HSU courses
- *Variations in projects*. Our study focused on team software development projects in which each student team works on a different project or a different piece of a project. Projects necessarily varied with respect to their level of difficulty, the opportunities they afforded for collaboration, and other dimensions. While improving the external validity of our study, these differences across software projects may have influenced the extent to which student perceptions captured through the CATME survey aligned with performance metrics derived through GitHub data.
- *Variations in teams*. In the WSU and HSU courses, teams were formed based on individuals' common interests in the available projects. In contrast, in the UCSB course, students were assigned to teams based on a team formation process that prioritized (a) ensuring women were on teams with at least one other woman, and (b) ensuring students were on teams with others in the same time zone. Variations in team composition and aptitude could have certainly influenced team performance and collaboration. By studying a broad range of teams and projects, this study aimed to mitigate this potential confound.

Summary, conclusions and future work

Team software development projects play a valuable role in computing education by providing computing students with authentic learning experiences. However, assessing individual performance in such projects is challenging, and often relies exclusively on *subjective* measures such as peer and instructor evaluation. Buffardi (2020) explored the possibility that *objective* measures of individual performance might enhance the subjective measures traditionally used to assess individual performance. His main result proved discouraging: Objective measures derived from GitHub log data yielded assessments that tended to be inconsistent with more traditional subjective assessments.

In this study, we set out to test the robustness of Buffardi's findings through a larger scale replication involving multiple software engineering courses at three different universities. While replicating all but one of Buffardi's significant findings, our study detected ten of 16 possible significant positive correlations between objective and subjective measures – eight more than Buffardi's original study. In demonstrating that objective measures derived from GitHub log data align with traditional peer and instructor evaluation, our work contributes an empirical foundation for more valid and comprehensive assessment strategies that incorporate both objective and subjective data.

While incorporating objective data into the assessment of individual performance in team software projects appears promising, one needs to keep in mind the limitations of the objective data considered in this work. As we have argued, GitHub data on students' commits, line changes, and issue involvement cannot tell a complete story about an individual's value or contribution to a team. Team members may take on roles and tasks that add value to a team without being directly involved in writing code. In future work, we would like to obtain a more nuanced understanding of individual student contributions by considering log data on a broader range of GitHub activities, including writing issues, updating and commenting on issues, participating in code reviews, and moving issues within a Kanban board. All of these tasks can make potentially valuable contributions to a team and are readily traceable through GitHub. In addition, we would like to consider individuals' *communication and coordination activities* in the online chat tools (e.g. Slack, Discord, Microsoft Teams) commonly used in collaborative software development. We have in fact collected such data as part of this work and are eager to incorporate it into future analyses.

Another aspect of individual performance not considered in this work is the *quality* of students' GitHub artifacts. In fact, in prior work (Hundhausen et al. 2021), we developed principled metrics for assessing the quality of issues, commit messages, and overall software product quality. In the future, we would like to explore correlations between these and other quality metrics and the GitHub metrics considered in this study.

Finally, while this work focused on using objective GitHub data as a basis for *summative assessment*, in future work we would like to explore the value of using GitHub data as a basis for *formative assessment*. To that end, we are developing *visual analytics dashboards* (e.g. Verbert et al., 2014) that present metrics that student software development teams and their instructors can use to dynamically monitor team processes and progress. We suspect that the greatest value of GitHub measures such as the ones explored in this work may lie not in their ability to more accurately assign grades, but in their ability to *stimulate individual and team reflection and discussion on software development processes and progress*. If students and teams are given the tools to monitor and reflect on their processes and progress relative to best practices, they should be in a better position to make course corrections that enable them to improve their individual and team performance.

Acknowledgments

This research is supported by the National Science Foundation under grant no. DUE 1915196. We are grateful to all our student participants who consented to share their data for research purposes, and to Kevin Buffardi for supplying us with additional data from his study and helping us to better understand what he did.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the National Science Board [DUE 1915196]; National Science Foundation [DUE 1915196]; National Science Foundation [DUE 1915196]; National Science Foundation [DUE 1915196].

ORCID

O. Adesope http://orcid.org/0000-0003-0620-500X

References

Abikoff, H., Courtney, M., Pelham, W. E., & Koplewicz, H. S. (1993). Teachers' ratings of disruptive behaviors: the influence of halo effects. *Journal of Abnormal Child Psychology*, *21*(5), 519–533. https://doi.org/10.1007/BF00916317

About issues. (n.d.). *GitHub Docs*. Retrieved November 19, 2021, https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues

ACM/IEEE Joint Task Force on Computing Curricula. (2013). *Computer science curricula 2013: curriculum guidelines for undergraduate degree programs in computer science (p. 518)*. ACM and IEEE Computer Society. https://doi.org/10.1145/2534860

Akoglu, H. (2018). User's guide to correlation coefficients. *Turkish Journal of Emergency Medicine*, *18* (3), 91–93. https://doi.org/10.1016/j.tjem.2018.08.001 PubMed

Begel, A., & Simon, B. (2008). Struggles of new college graduates in their first software development job. *SIGCSE Bull*, *40*(1), 226–230. https://doi.org/10.1145/1352322.1352218

Braught, G., Maccormick, J., Bowring, J., Burke, Q., Cutler, B., Goldschmidt, D., Krishnamoorthy, M., Turner, W., Huss-Lederman, S., Mackellar, B., & Tucker, A. (2018). A multi-institutional perspective on H/FOSS projects in the computing curriculum. *ACM Transactions on Computing Education (TOCE)*, *18*(2), 1–31. https://doi.org/10.1145/3145476

Bruegge, B., Krusche, S., & Alperowitz, L. (2015). Software engineering project courses with industrial clients. *ACM Transactions on Computing Education (TOCE)*, *15*(4), 1–31. https://doi.org/10.1145/2732155

Buffardi, K. (2020). Assessing individual contributions to software engineering projects with Git Logs and user stories. *Proceedings of the 51st ACM technical symposium on computer science education* (pp. 650–656). New York: ACM. https://doi.org/10.1145/3328778.3366948

Buffardi, K., Robb, C., & Rahn, D. (2017). Learning agile with tech startup software engineering projects. *Proceedings of the 2017 ACM conference on innovation and technology in computer science education* (pp.28–33). New York: ACM. https://doi.org/10.1145/3059009.3059063

Build software better, together. (n.d.). *GitHub*. Retrieved December 29, 2021, https://github.com

Campbell, D. T. (1979). Assessing the impact of planned social change. *Evaluation and Program Planning*, *2*(1), 67–90. https://doi.org/10.1016/0149-7189(79)90048-X

catme.org. (2021). *Welcome to CATME - smarter teamwork*. CATME. https://info.catme.org/

catme.org. (n.d.). *Peer Evaluation*. Retrieved December 29, 2021, https://www.catme.org/help/student/student3.html

Clark, N., Davies, P., & Skeers, R. (2005). Self and peer assessment in software engineering projects. *Proceedings of the 7th australasian conference on computing education - volume 42* (pp. 91–100). New York: ACM. https://dl.acm.org/doi/10.5555/1082424.1082436

Clear, T. (2009). Thinking issues: the three p's of capstone project performance. *ACM SIGCSE Bulletin*, *41*(2), 69–70. https://doi.org/10.1145/1595453.1595468

Craig, M., Conrad, P., Lynch, D., Lee, N., & Anthony, L. (2018). Listening to early career software developers. *Journal of Computing Science and Engineering Coll*, *33*(4), 138–149. https://dl.acm.org/doi/10.5555/3199572.3199591

Devadiga, N. M. (2017). Software engineering education: converging with the startup industry. *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*, 192–196. https://doi.org/10.1109/CSEET.2017.38

Domínguez, C., Jaime, A., García-Izquierdo, F. J., & Olarte, J. J. (2020). Factors considered in the assessment of computer science engineering capstone projects and their influence on discrepancies between assessors. *ACM Transactions on Computing Education*, *20*(2), 14:1–14:23. https://doi.org/10.1109/CSEET.2017.38

Ellis, H. J., & Hislop, G. W. (2016). Pathways to student learning within HFOSS. *Proceedings of the 17th annual conference on information technology education*, 168.

Exter, M. (2014). Comparing educational experiences and on-the-job needs of educational software designers. *Proceedings of the 45th ACM technical symposium on computer science education*, 355–360. https://doi.org/10.1145/2538862.2538970

Fagerholm, F., & Vihavainen, A. (2013). Peer assessment in experiential learning assessing tacit and explicit skills in agile software engineering capstone projects. *2013 IEEE frontiers in education conference (FIE)* (pp. 1723–1729). Piscataway, NJ: IEEE. https://doi.org/10.1109/FIE.2013.6685132

Farrell, V., Ravalli, G., Farrell, G., Kindler, P., & Hall, D. (2012). Capstone project: fair, just and accountable assessment. *Proceedings of the 17th ACM annual conference on innovation and technology in computer science education* (pp.168–173). New York: ACM. https://doi.org/10.1145/2325296.2325339

Faul, F., Erdfelder, E., Lang, A.-G., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Berhavior Research Methods*, *39*(2), 175–191. https://doi.org/10.3758/BF03193146

Garousi, V., Petersen, K., & Ozkan, B. (2016). Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review. *Information and Software Technology*, *79*, 106–127. https://doi.org/10.1016/j.infsof.2016.07.006

GitHub.com. (2021a). *About issues*. GitHub Docs. https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues

GitHub.com. (2021b). *About project boards*. GitHub Docs. https://docs.github.com/en/issues/organizing-your-work-with-project-boards/managing-project-boards/about-project-boards

GitHub.com. (2021c). *About pull requests*. GitHub Docs. https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests

Herbert, N. (2007). Quantitative Peer Assessment: Can students be objective. *Proceedings of the 9th Australasian Computing Education Conference (ACE2007)*. 30 Jan–2 Feb 2007. Ballarat, Victoria. https://eprints.utas.edu.au/4889/

Herbert, N. (2018). Reflections on 17 years of ICT capstone project coordination: effective strategies for managing clients, teams and assessment. *Proceedings of the 49th ACM technical symposium on computer science education* (pp. 215–220). New York: ACM. https://doi.org/10.1145/3159450.3159584

Hewner, M., & Guzdial, M. (2010). What game developers look for in a new graduate: interviews and surveys at one game company. *Proceedings of the 41st ACM technical symposium on computer science education* (pp. 275–279). New York: ACM. https://doi.org/10.1145/1734263.1734359

Hundhausen, C. D., Carter, A. C., Conrad, P., Tariq, A., & Adesope, O. (2021). Evaluating Commit, Issue and Product Quality in Team Software Development Projects. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 108–114). New York: ACM. https://doi.org/10.1145/3408877.3432362

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2014). The promises and perils of mining GitHub. *Proceedings of the 11th working conference on mining software repositories* (pp. 92–101). New York: ACM. https://doi.org/10.1145/2597073.2597074

Latané, B., Williams, K., & Harkins, S. (1979). Many hands make light the work: the causes and consequences of social loafing. *Journal of Personality and Social Psychology*, *37*(6), 822. https://doi.org/10.1037/0022-3514.37.6.822

Layton, R., Ohland, M., & Pomeranz, H. (2007). *Software for student team formation and peer evaluation: CATME incorporates team-maker*. *Faculty Publications - Mechanical Engineering*. https://scholar.rose-hulman.edu/mechanical_engineering_fac/485

Mead, N. R. (2015). Industry/university collaboration in software engineering education: refreshing and retuning our strategies. *Proceedings of the 37th international conference on software engineering - volume 2* (pp. 273–275). Piscataway, NJ: IEEE.

Meyers, C. V., & Brandt, W. C. (Eds.). (2015). *Implementation fidelity in education research: designer and evaluator considerations*. Routledge.

Murphy, C., Buffardi, K., Dehlinger, J., Lambert, L., & Veilleux, N. (2017). Community engagement with free and open source software. *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education* (pp. 669–670). New York: ACM.

Murphy, C., Sheth, S., & Morton, S. (2017). A two-course sequence of real projects for real customers. *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education* (pp. 417–422). New York: ACM. https://doi.org/10.1145/3017680.3017742

Ohland, M. W., Loughry, M. L., Carter, R. L., Bullard, L. G., Felder, R. M., Finelli, C. J., Layton, R. A., & Schmucker, D. G. (2005). Developing a peer evaluation instrument that is simple, reliable, and valid. In *4th ASEE/aaee global colloquium on engineering education* (pp. 302). Washington, DC: American Socieity for Enginnering Education.

Ohland, M. W., Loughry, M. L., Woehr, D. J., Bullard, L. G., Felder, R. M., Finelli, C. J., Layton, R. A., Pomeranz, H. R., & Schmucker, D. G. (2012). The comprehensive assessment of team member effectiveness: development of a behaviorally anchored rating scale for self-and peer evaluation. *Academy of Management Learning & Education*, *11*(4), 609–630. https://doi.org/10.5465/amle.2010.0177

Parker, H., & Holcombe, M. (1999). Campus-based industrial software projects: risks and rewards. *Proceedings of the 4th annual SIGCSE/SIGCUE ITiCSE conference on innovation and technology in computer science education* (pp. 189). New York: ACM.

Pinto, G., Ferreira, C., Souza, C., Steinmacher, I., & Meirelles, P. (2019). Training software engineers using open-source software: the students' perspective. *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)* (pp. 147–157). Piscataway, NJ: IEEE.

Pinto, G. H. L., Figueira Filho, F., Steinmacher, I., & Gerosa, M. A. (2017). Training software engineers using open-source software: the professors' perspective. *2017 IEEE 30th conference on software engineering education and training (CSEE&T)* (Piscataway, NJ: IEEE), 117–121.

Radigan, D. (n.d.). *What are story points and how do you estimate them? atlassian*. Retrieved December 29, 2021, https://www.atlassian.com/agile/project-management/estimation

Rehkopf, M. (n.d.). *User stories | examples and template*. Atlassian. Retrieved December 29, 2021, https://www.atlassian.com/agile/project-management/user-stories

Saltz, J. S., & Heckman, R. R. (2018). A scalable methodology to guide student teams executing computing projects. *ACM Transactions on Computing Education*, *18*(2), 1–9. https://doi.org/10.1145/3145477

Schneider, F. W., Gruman, J. A., & Coutts, L. M. (2011). *Applied Social Psychology: Understanding and Addressing Social and Practical Problems*. Thousand Oaks, CA: SAGE.

Sherriff, M., & Heckman, S. (2018). Capstones and Large Projects in Computing Education. *ACM Trans. Comput. Educ*, *18*(2), 2. https://doi.org/10.1145/3229882

Smith, T. M., McCartney, R., Gokhale, S. S., & Kaczmarczyk, L. C. (2014). Selecting open source software projects to teach software engineering. *Proceedings of the 45th ACM technical symposium on computer science education* (New York: ACM), 397–402. https://doi.org/10.1145/2538862.2538932

Sutherland, J. (2018, July 17). *Performance appraisals update*. Scrum Inc. https://www.scruminc.com/performance-appraisals/

Tafliovich, A., Petersen, A., & Campbell, J. (2015). On the evaluation of student team software development projects. *Proceedings of the 46th ACM technical symposium on computer science education* (pp. 494–499). New York: ACM. https://doi.org/10.1145/2676723.2677223

Van den Bergh, L., Denessen, E., Hornstra, L., Voeten, M., & Holland, R. W. (2010). The implicit prejudiced attitudes of teachers: Relations to teacher expectations and the ethnic achievement gap. *American Educational Research Journal*, *47*(2), 497–527. https://doi.org/10.3102/0002831209353594

Verbert, K., Govaerts, S., Duval, E., Santos, J. L., Van Assche, F., Parra, G., & Klerkx, J. (2014). Learning dashboards: An overview and future research opportunities. *Personal and Ubiquitous Computing*, *18*, 1499–1514. https://doi.org/10.1007/s00779-013-0751-2

von Konsky, B. R., & Ivins, J. (2008). Assessing the capability and maturity of capstone software engineering projects. *Proceedings of the tenth conference on australasian computing education - volume 78* (pp. 171–180). Australia: Australian Computer Society.