2023-10-10

Tell It With Commits To Git

Pia NIEMELÄ
Tampere University, Finland, pia.s.niemela@gmail.com

Jenni HUKKANEN
Tampere University, Finland, jenni.j.hukkanen@tuni.fi

Mikko NURMINEN
Tampere University, Finland, mikko.nurminen@tuni.fi

See next page for additional authors

Authors
Pia NIEMELÄ, Jenni HUKKANEN, Mikko NURMINEN, Antti SAND, and Hannu-Matti JÄRVINEN

TELL IT WITH COMMITS TO GIT

Recipes for successful teamwork in computer science education

P. Niemelä *

ORCID 0000-0002-8673-9089

J. Hukkanen

ORCID 0000-0002-7691-5974

M. Nurminen

ORCID 0000-0001-7609-8348

A. Sand

ORCID 0000-0002-9508-4205

H.-M. Järvinen

ORCID 0000-0003-0047-2051

Tampere University

Tampere, Finland

ABSTRACT

In the realm of higher education, team work is considered a crucial factor in facilitating the development of collaborative and work-life skills among students. Studies have shown that collaborative tasks promote a sense of community, belonging, and well-being among students. However, uneven work distribution and vague team roles can lead to dissatisfaction and reduced commitment, ultimately resulting in assignment failure and impeding progress towards graduation. This paper examines students' feedback on Britton et al.'s (2017) team quality questionnaire, along with their contributions to team performance. Additional questions were added to gauge how students communicate and organize their work within their teams, with an analysis of their activity and contributions measured from their git repositories. Ultimately, this paper presents successful ingredients for team work and suggests strategies for ensuring a positive experience in dynamic team settings. These strategies include clearly defined roles, equal workload distribution, and accountability mechanisms.

*Corresponding author

P. Niemelä

pia.niemela@tuni.fi

1 INTRODUCTION

Conway's Law points out the resemblance between the software architecture and structure of the organization (Conway 1968). The law suggests that if an organization is divided into teams, the resulting software architecture will mirror the lines of communication and collaboration between those teams. This concept has been observed in practice, when large monolithic software systems transform into distributed microservices, where-after sizable organizations undergo a deconstruction into smaller agile teams that direct themselves and take ownership of their tasks. Furthermore, there has been a shift from extensive planning to the adoption of test-driven development (Kaufmann and Janzen 2003; Gupta and Jalote 2007), to the extent of extreme programming (Bell 2001) where tests play a crucial role as a de facto specification guiding the development process. The delivery of functional end products that meet customer's requirements remains the target but with shortened feedback loops.

Universities must be able to provide industry with new employers that have internalized the principles of the Agile Manifesto (Beck et al. 2001), and are able to form flexible, self-organizing teams (Gren and Lenberg 2020). The influence of Agile Manifesto on flattening organizations has been significant. Flat organizations eliminate unnecessary layers of management, reduce bureaucracy and enable faster decision-making, thus encouraging a shift away from hierarchical structures by empowering teams. Instead of relying on strict top-down control, Agile promotes shared responsibility and decision-making within teams (Conboy et al. 2010), and such Agile methodologies as Scrum or Kanban, to provide frameworks for organizing work, fostering collaboration, and allowing teams to prioritize and manage their own tasks.

In addition to Agile, university Web&Cloud courses introduce DevOps methodology. The main motivation for DevOps is the automation: to achieve continuous integration, delivery, and deployment (CI/CD) pipelines. Automation enables Agile teams to automate repetitive and manual tasks, allowing them to focus more on delivering value and innovation. Besides the automation, DevOps emphasizes collaboration and good communication between team members. The collaborative approach fosters a shared sense of ownership and collective responsibility of the outcome.

2 RELATED WORK

In Tampere University, the DevOps comes in the flavor of GitOps, Gitlab being the version control system integrated with the learning management system in use in courses (Colantoni et al. 2021; Beetz and Harrer 2021). However, DevOps falls short of its full potential due to the absence of continuous deployment (CD) adoption. The introduction of CD is currently postponed until more advanced, master-level courses that use the Kubernetes system. Conversely, continuous integration (CI), which involves the integration of unit and integration tests, is extensively employed. Additionally, the Gitlab issue board, which bears a resemblance to a Kanban board (Nakazawa and Tanaka 2016), is also heavily utilized to foster team interaction and facilitate the progress of work through sprints that are split into tasks shared among team members. Thus, both the Gitlab version control system and issue board serve as tools for monitoring team dynamics and promoting collaboration in ongoing projects.

Parizi et al. have explored the use of Git *"to capitalize on team-aware metrics"* (Parizi, Spoletini, and Singh 2018). The authors advocate for leveraging the power of Git-driven technology and associated features to measure a team member's contributions throughout the entire progression of the project – not just upon its completion. This approach ensures a comprehensive and precise assessment of individual performance anticipated to foster team-based learning, ultimately resulting in the cultivation of graduates equipped to meet the demanding standards of the software industry.

To investigate students' group formation, Auvinen et al. studied 150 college students who first individually solved exercises and then worked in teams of three on a class project (Auvinen et al. 2020). The study found that teams with both low-and high-performing students achieved almost the same results as teams with only high-performing students, meaning teams should comprise of both low- and high-performers rather than just one or the other. On the other hand, individual students' poor time management practices had a negative effect on their teammates' time management. Most teams assigned tasks to maximize the acquisition of technical skills, rather than training them.

"Can we pick our own groups?" is a common query that arises among students as they anticipate undertaking substantial assignments. Chapman et al. conducted research to examine the impact of group formation, specifically self-selection versus random formation, on group dynamics, outcomes, and students' attitudes towards the group experience (Chapman et al. 2006). Chapman's findings suggest that allowing students to self-select into groups may be preferable to random assignment, as it more closely simulates real-world work groups. The research implies that self-selection can contribute to enhanced group dynamics, reduced concerns, improved attitudes, and overall positive student experiences within group settings. However, it is important to note that self-selection may also have certain drawbacks, such as lower perceived efficiency and increased conflict. Thus, careful consideration is needed to balance the benefits and challenges associated with self-selection when designing group formation.

Even if Chapman's study suggests potential benefits of self-selection, it is essential to critically evaluate the implications in relation to equity. It is worth acknowledging that only a small number of students are fortunate enough to have high-performing and established teams readily available. This advantage becomes compounded if students are repeatedly allowed to self-select their groups. Allowing students to freely choose their own groups can perpetuate existing inequalities, as students with stronger social networks or prior established relationships may repeatedly form high-performing teams, while others may be left at a disadvantage.

2.1 TeamQ questionnaire in a retrospect

Students themselves should have their say on team quality, too. The team quality study conducted by Britton et al. demonstrates the potential of their tailored measurement tool, TeamQ, to assess undergraduate students' teamwork skills (Britton et al. 2017). Their results indicate that communication and problem-solving were the strongest areas of performance, whereas collaboration and team leadership were identified as areas posing significant challenges. Furthermore, the findings suggest that gender may influence individual teamwork skills, with females scoring higher than males in all four areas. This data provides a valuable insight into the efficacy of the tool, suggesting it can be used to effectively evaluate individual teamwork skills in undergraduate education. In the context Web&Cloud courses in Tampere University, TeamQ has been incorporated into reflective post-course questionnaires in a yearly basis since 2020.

In this study, we aim to answer the following research questions:

1. What is the optimal size for a team?
2. How do teams divide work and follow-up on the process?
3. What can course personnel do to ensure better team work experience?

3 RESEARCH CONTEXT

The subject of this study pertains to the Web&Cloud domain, focusing on two consecutive courses: Web for Content Authors and Information Scientists (aka WebCAIS) and Web Development 1 (WebDev1). WebCAIS targets first- and second-year students and concentrates on frontend web technologies, such as HTML, CSS, and JavaScript. Building on this foundation, WebDev1 provides a comprehensive overview of both frontend and backend web technologies, with an introduction to Node.js as a major new technology. This course is intended for third- and fourth-year students who possess a significant programming background, including a fundamental understanding of project work such as Agile project management.

The development process for both WebCAIS and WebDev1 adopts an iterative approach characterized by cyclic design and redesign phases. During the 2019-2020 academic year, the initial stages of development began with the transfer of grading from manual to auto-grading, as documented by earlier studies (Niemelä and Nurminen 2020). After that, there has been a recent shift in focus towards enhancing students' self-efficacy and optimizing their overall learning experience, particularly within the context of collaborative teamwork settings. This shift recognizes the importance of empowering students to take ownership of their learning, foster their sense of confidence and competence, and promote meaningful engagement and communication with their peers.

3.1 Tools used: Gitlab, Plussa and Peer-Review Service

Gitlab plays a pivotal role in final assignments; in WebDev1, also in weekly exercises. Git repositories are created subsequent to the group formation. To streamline project management and task coordination, the Gitlab issue board is recommended as a tool, the rationale being Gitlab's utilization as a version control system, too. This board offers a Kanban-like interface for managing issues. As a primer, students are provided with a few Plussa exercises to familiarize themselves with the issue board. In terms of documentation requirements, groups are advised to employ issues to assign tasks within the board. When utilized effectively, this approach provides a comprehensive overview of each group's progress allowing tasks to be transitioned through various stages such as the backlog "TODOs", then "Doing," and ultimately "Done". These transitions also serve as indicators to other group members not to intervene in ongoing work.

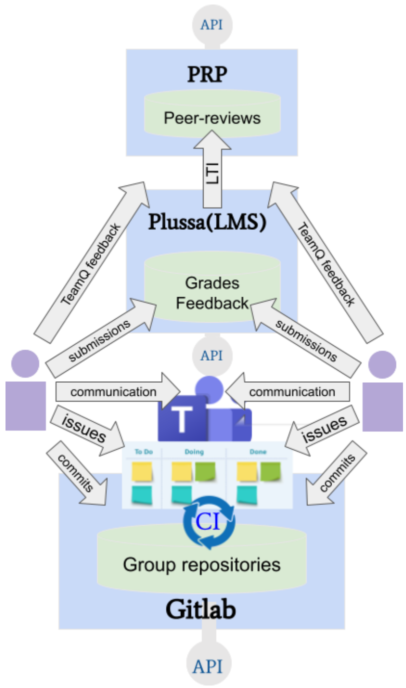

Fig. 1. The teamwork env

The already-mentioned Plussa is the actual learning management system (Karavirta, Ihantola, and Koskinen 2013). Once the assignment is implemented, tested and committed in the Gitlab version control system, the assignment will be graded by giving its Git URL to Plussa. Fig.1 illustrates the teamwork context, which includes work division as agreed tasks in Gitlab issue board, thereafter committing code to Gitlab, followed by submissions to Plussa system, which grants the actual grade that will be stored in the system. In addition to being integrated with Gitlab, Plussa can be extended with other grading modules integrated via Learning Tools Interoperability (LTI) protocol. One such module is Peer Review Platform (Heino 2019), that is used for both intra-course and -team peer-reviewing purposes. In studied courses, the peer-review is intra-team: team members give feedback to each other of the quality of collaboration.

4 RESULTS AND DISCUSSION

After the assignment, students reply to the TeamQ questionnaire by Britton in Likert-scale [1..5], see Fig. 3 and 4. Upon initial observation, it is apparent that students consistently assign significantly positive scores to one another, which can be attributed to a phenomenon called reciprocal mutualism. This behavior stems from the anticipation that providing positive ratings will result in receiving high scores in return. Interestingly, the question regarding passive-aggressive receives the highest score, the question in

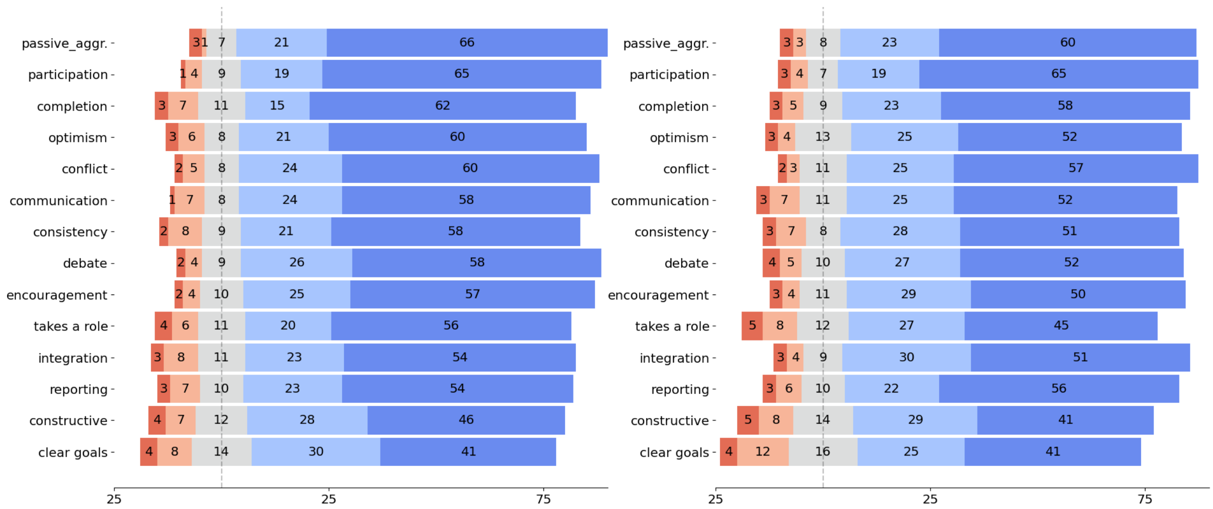

Fig. 2. TeamQ results, WebCAIS (N=320) and WebDev1 (N=512)

its entirety being: *”Displays appropriate assertiveness: neither dominating, submissive, nor passive aggressive”*. Students tend to interact with each other in a pragmatic and straightforward manner, and this specific question tends to elicit thought-provoking and intimidating responses. Consequently, some students feel compelled to seek clarification and confirmation from course personnel to ensure that the questionnaire serves a genuine purpose. Even after the confirmation, the highest scores for one’s peers may still be perceived as the safest response to such a ”bold” question.

Constructive feedback and clear goals get the least points. Constructive feedback necessitates both courage and a willingness to assume responsibility for the project’s success. Similarly, low scores in the clear goals category suggest communication issues and deficiencies in work division. These challenges are indicative of broader project management issues and suggest a need for clearer instructions to be provided.

4.1 Ideal team sizes

Table 1. The votes for the ideal size in both course implementations

Course	N	Votes for [1,2,3,4,5]	Sum	Avg	Std
WebCAIS	605	[17, 27, 137, 36, 2]	219	2.90	0.79
WebDev1	956	[10, 104, 311, 78, 9]	512	2.95	0.71

The result of the ideal team size is clear: three clearly outscores other options between one and five, see Fig. 4.1 and more in detail Table 1. The portion of ”two” is significantly bigger in WebDev1, where the first 2020-21 implementations used it as a default size. This observation suggests that either the experience with this size was genuinely positive, or alternatively, participants may have opted for it to avoid expressing any criticism towards their peers. In WebCAIS, the portion of ones is bigger than in WebDev1: 7.8% vs. 2.0%. For the most of the students, the final assignment of WebCAIS is their first

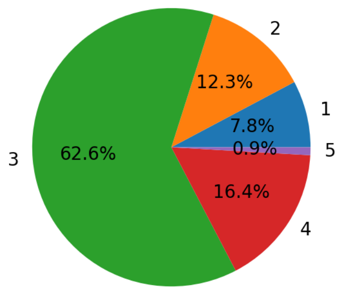

Fig. 3. The ideal team size in WebCAIS

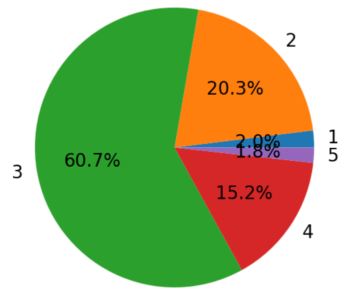

Fig. 4. ..and in WebDev1

group work done in Git, thus the skill levels vary a lot. The experienced students may feel tempted to do everything by themselves and they do not see benefits of teamwork.

Size rationales

The following list summarizes the rationales on each team size grouped by the values of [1..5]; the rationales were asked only in WebCAIS.

1. the students felt frustrated due to lack of collaboration, uneven skill levels, poor communication, and individual members who do not contribute adequately, and these students thus can be categorized as ”disappointed”.
2. the students highlighted optimal scheduling, coordination, task division based on individual strengths, allowing one to focus on JavaScript while the other could contribute to CSS; and suggested pair programming as an approach.
3. the reasons for the ideal team size of three were not very different from two, that is, effective workload distribution, balanced contributions, efficient communication, learning opportunities, and the ability to divide tasks based on individual skills and interests.
4. the students who preferred four highlighted the benefits of even workload distribution, flexibility in collaboration, reliability, technical expertise, and balanced workloads.
5. only two students rationalized for their ”fives” by emphasizing creativity gains and probability of getting a functional group even if members were selected at random.

Even if a larger team means less work, students did not automatically up-vote it, for the downsides, such as the challenges in coordination, started to weigh more. There are more opinions, ideas, and perspectives to consider, leading to longer discussions and potential conflicts. Students also highlighted the risk of uneven workloads: with a group of four, five or bigger, there is a higher chance of having an uneven distribution of workload. It may be more difficult to divide tasks equally among members, leading to potential disparities in effort and contribution. This can result in some members feeling overwhelmed with too much work, while others may have less responsibility, causing dissatisfaction within the group. Limited participation and engagement are real threats in larger groups, because team members may have fewer opportunities to actively participate and contribute, yet it is worth noting that some members may intentionally limit their participation as a strategy to avoid work (Järvinen, Niemelä, and Virta 2019).

4.2 Git for screening students' input in team work

Table 2 displays the Git commit data including additions, total changes, and the percentage of additions. The data reveals the workload for the assignment in the WebDev1 course implementations are at least 10-fold higher than in the WebCAIS course implementations. Additionally, it appears that the students in the WebCAIS course have a higher proportion of deletions compared to those in the WebDev1 course, which could signal that the students at the WebDev1 course are more experienced coders, and hence, do not need to delete that much.

Table 2. Students' Git commits as additions, total, and percentage of additions

Course	Impl.	N. committers	Avg.add.	Avg.total	Add.-%
WebCAIS	spring-2021	36	431	680	70.4
	spring-2022	35	434	495	65.8
WebDev1	fall-2020	171	10349	12155	90.8
	spring-2021	27	14324	18333	89.5
	spring-2021	71	4738	6556	75.5
	fall-2022	171	4538	5288	75.9

4.3 Work division and follow-up in teams

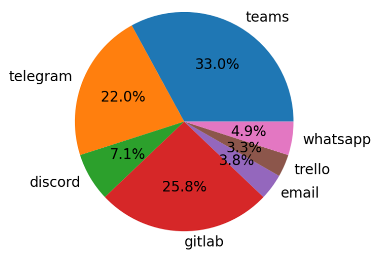

Fig. 5. Follow-up tools

The question about work division and follow-ups were answered by WebDev1 students only. Dividing tasks effectively is important for the final assignment to succeed. A good work division promotes collaboration, maximizes individual contributions, and ensures the timely completion of project objectives. WebDev1 teams adopted diverse approaches for task division, such as allocated tasks in Agile manner; gave a complete responsibility for specific tasks to a selected member after assessing the complexity and workload of each task versus individual preferences and strengths, made decisions collaborative decision-making, where teams engaged in discussions and decision-making processes to collectively divide tasks; and assigned tasks based on immediate project needs, availability, and skill sets, ensuring flexibility and adaptability; divided tasks in sub-tasks, that were smaller and more manageable.

Effective progress monitoring is a crucial factor for successful collaboration and project completion. The teams utilized various strategies, including maintaining open lines of communication through different channels like Teams, Telegram, and Discord (as shown in Fig. 5), to ensure efficient workflow and accountability. These channels were used to exchange updates on completed and ongoing tasks, as well as any difficulties encountered during the project.

4.4 Constructing functional teams

Teams can be constructed numerous ways. During the history of WebCAIS and WebDev1, there have been different experiments. In WebCAIS 2020-2022, groups were the size of three and the roles were fixed: a project manager, an architect and a developer. A project manager was selected based on the Nexus questionnaire (Korhonen 2014) by measuring self-directedness and commitment to studies. An architect was selected based on substance knowledge measured by weekly exercises and self-evaluations. A developer was a novice thought to learn from their peers. In the earlier WebDev1 iterations, students answered a Plussa questionnaire probing their preferences related to group work. Groups were then formed by the course personnel based on their answers. In later iterations, students formed the groups using Moodle virtual learning environment, where students first discuss their preferences and after finding their mates, they register their groups in Moodle group formation activity. Teamless students were grouped by course personnel. The size of three was the default, but as some students had a strong preference to work alone, they were allowed to do so.

In our current method for forming groups, the main criterion is the students' target course grade. While the approach is justified, it may result in groups that are too similar, lacking the necessary diversity and range of skills. In contrast, Auvinen et al. argue in favor of heterogeneity. To achieve a more balanced approach, it is crucial that alternative methods for forming groups are tested (Auvinen et al. 2020).

4.5 ”Mid-point check” and other means to intervene poorly-functioning teams

To prevent students from having negative teamwork experiences, course personnel must monitor the situation, regardless of whether groups are formed by students or the personnel. In an advanced-level web architecture course known as WebDev2, poorly functioning teams were identified and intervened through a mandatory mid-point check. After two weeks of project work, the groups needed to open a Gitlab issue and tag a lecturer. The lecturer then checked Git commits and the issue board to verify that all team members were sufficiently employed. If the check showed adequate activity and fair division of labor, the issue was closed. Otherwise, the lecturer contacted the students and reminded them of the importance of realistic scheduling and fair work division. This intervention often yielded responses even from students otherwise ghosting their peers. This added manual work to the lecturer but was beneficial in catching possible issues with poorly-functioning teams and falling behind the intended group work schedule. It would be beneficial to automate this process, utilizing available data and open APIs in the relevant systems wherever possible.

The findings demonstrated that teams performing well in the mid-project assessment were likely to maintain a high level of performance throughout the project. Conversely, groups displaying limited activity were at an elevated risk of encountering time constraints and experiencing an unfavorable group work experience.

5 CONCLUSIONS

RQ1 Optimal team size: Three outscored other options, in WebDev1 also two scored relatively high, when it was used as the default size.
RQ2 Work division and follow-up: Teams work in Agile manner by utilizing project management tools like Gitlab issue board and strengths of individuals, by communicating via Teams, Telegram or Discord and by adapting flexibly to the situation.
RQ3 Towards better team work experiences: Group formation is crucial, the size matters, as well as students' goals and expectations, such as target grade. Mid-point checks have proven to help, also automatic supervision tools based on git commits and issue boards were anticipated to be beneficial.

REFERENCES

Auvinen, Tapio, Nickolas Falkner, Arto Hellas, Petri Ihantola, Ville Karavirta, and Otto Seppälä. 2020. "Relation of Individual Time Management Practices and Time Management of Teams." *2020 IEEE Frontiers in Education Conference (FIE)*. 1–9.

Beck, Kent, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. 2001. The Agile Manifesto.

Beetz, Florian, and Simon Harrer. 2021. "GitOps: The Evolution of DevOps?" *IEEE Software* 39 (4): 70–75.

Bell, John T. 2001. "Extreme programming." *Thinking for Innovation*.

Britton, Emily, Natalie Simper, Andrew Leger, and Jenn Stephenson. 2017. "Assessing teamwork in undergraduate education: a measurement tool to evaluate individual teamwork skills." *Assessment & Evaluation in Higher Education* 42 (3): 378–397.

Chapman, Kenneth J, Matthew Meuter, Dan Toy, and Lauren Wright. 2006. "Can't we pick our own groups? The influence of group selection method on group dynamics and outcomes." *Journal of Management Education* 30 (4): 557–569.

Colantoni, Alessandro, Benedek Horváth, Ákos Horváth, Luca Berardinelli, and Manuel Wimmer. 2021. "Towards continuous consistency checking of DevOps artefacts." *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, 449–453.

Conboy, Kieran, Sharon Coyle, Xiaofeng Wang, and Minna Pikkarainen. 2010. "People over process: key challenges in Agile development." *IEEE Software* 28 (4): 48–57.

Conway, Melvin E. 1968. "How do committees invent." *Datamation* 14 (4): 28–31.

Gren, Lucas, and Per Lenberg. 2020. “Agility is responsiveness to change: An essential definition.” In *Proceedings of the Evaluation and Assessment in Software Engineering*, 348–353.

Gupta, Atul, and Pankaj Jalote. 2007. “An experimental evaluation of the effectiveness and efficiency of the test driven development.” *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*. IEEE, 285–294.

Heino, P. 2019. Peer-review Platform. `https://github.com/piehei/prplatform`.

Järvinen, Hannu-Matti, Pia Niemelä, and Ulla-Talvikki Virta. 2019. “Combining flipped ideas and online learning.” *SEFI Annual Conference*. European Society for Engineering Education SEFI, 566–577.

Karavirta, V., P. Ihantola, and T. Koskinen. 2013. “Service-Oriented Approach to Improve Interoperability of E-Learning Systems.” *2013 IEEE 13th International Conference on Advanced Learning Technologies*.

Kaufmann, Reid, and David Janzen. 2003. “Implications of test-driven development: a pilot study.” *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. 298–299.

Korhonen, Vesa. 2014. “Opintoihin kiinnittymisen arviointia kehittämässä.” *Nexus-itsearviointikyselyn teoreettista taustaa ja empiiristä kehittelyä. Tampere: Campus Conexusprojektin julkaisuja B*, vol. 3.

Nakazawa, Shun, and Tetsuo Tanaka. 2016. “Development and application of Kanban tool visualizing the work in progress.” *2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*. IEEE, 908–913.

Niemelä, Pia, and Mikko Nurminen. 2020. “Rate Your Mate for Food for Thought: Elsewhere Use a Grader.” *Proceedings of the 12th International Conference on Computer Supported Education - Volume 2: CSEDU,*. INSTICC: SciTePress, 422–429.

Parizi, Reza M., Paola Spoletini, and Amritraj Singh. 2018. “Measuring Team Members’ Contributions in Software Engineering Projects using Git-driven Technology.” *2018 IEEE Frontiers in Education Conference (FIE)*. 1–5.