Experiments on Applying Peer-Review Software for Programming Assignments

Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen, Jari Porras

Abstract: *We tested a web-based peer-review application MyPeerReview in a Web programming course to get experience about the usability of application and process as well as the value of peer-review for students and teachers. Recent studies describe methods of analyzing peer-review data to produce better accuracy and reliability. We carried out an inquiry concerning students' experiences and produced statistics for analysis of student assessments. Whereas students' attitude about the application and process were positive, attention must be paid for the evaluation pattern to get valuable and more precise evaluations.*

Key words: *Peer-Review, Student Assessment, Programming.*

INTRODUCTION

Use of student peer-review has been studied in numerous educational institutions but according to Garousi [3] it has not been used on many design-oriented engineering courses. Motives for its use derive from the observations that peer-reviewing can, for example, **boost learning outcomes** and **provide accurate results in estimating** the quality of submissions. The principles of peer-reviewing are simple and easily adaptable for various contexts. Existing peer-review systems are generally aimed for handling single submissions, e.g. essay, and do not suit to evaluation of programming assignments, which often include multiple program files. Due to these kinds of problems we implemented a new peer-review system, MyPeerReview. The system is available to download at http://sourceforge.net/projects/mypeerreview/ and system is described in [5]. In this paper we present analysis of the system applied on a Web programming course.

It is unreasonable to expect people to spend much time to learn system functionality when that time and effort could be allotted to actual work. Problems with the tools and methods for conducting the process can sometimes defeat the gained benefits. A slow and/or poorly designed system that wastes students' time frustrates users quickly. Majority of users may use the system only once or twice so their time should be spent efficiently. Therefore the **usability** and especially **learnability** play an important role when using this type of application.

In order to have a positive effect on students' learning experience the system providing the framework for the process should a) minimize the requirements needed to use the system, b) perform well under load and volumes of data and c) be simple to use to minimize the impact of having to learn a new system. In other words, the overhead caused by the system in performing the required use cases should be cut as short as possible. To motivate the students to complete their peer-review tasks delicately, they have to be motivated and the process itself has to be efficient and support their learning.

The paper presents introduction to related work and then discusses on issues using peer-review on programming courses. Results section presents deployment experiences on a web programming course both from system usability and peer-reviewing process approach.

RELATED WORK

The benefits of peer-reviewing can be divided in two parts: a) the students who learn from others' solutions and received feedback and b) the teachers who can get assistance for the evaluation. To gain these benefits different approaches are needed; while teachers can use ratings to help the evaluation, students need comments and proposals to improve.

In case of student assessment in programming courses the peers are exposed to various styles of coding and asked to give critical evaluation of others' work. This can give students an opportunity to better their own proficiency as a programmer, and deepen their understanding of used languages and techniques. Also, the time spent on studying the solutions in practice is always beneficial when teaching new concepts.

Seeing different approaches is particularly insightful when handling a common problem that has “solutions that meet the requirements” instead of only a “single correct answer.” The teacher can present a few examples of such solutions, but it is essential for the students to realize the multitude of approaches. Sometimes, an absolute measurement of a solution’s suitability is hard to determine – its suitability can depend on factors like compatibility, reliability, optimized execution time, scalability etc. Often in practice, finding the best possible solution is not even desired, but instead, producing a working solution under the given parameters in time is all that counts.

Studies note that students find the opportunity to inspect peers’ solutions interesting because it can e.g. give them new ideas and a chance to learn analytical abilities from varying styles in solving problems [2]. Increased feedback about the assignment is usually appreciated especially if the teachers are only capable of giving out limited reviews. Also, peer-reviewing can incite peer interactivity within the group and provide ground for development of a community of active learning. Students stuck with a problem can seek the group’s help in form of suggestions and critical evaluation of their solutions. The group can then confirm good approaches and point out flawed ones, for example.

From the peer-reviewing point of view growing numbers of students can be turned into an advantage. For example Bouzidi & Jaillet [1] have reported that students can provide reliable and accurate results when group sizes are large (242 students in their case) giving a chance to assign enough peer-reviews for each. However, while they applied the process in a very controlled way on the assessments of mathematics exam solutions, the process we describe is used on evaluating student submissions on a programming course. It still applies, though, that the process will scale without burdening the individuals; a student’s workload along with the accuracy of student assessments depends on the ratio of authors per reviews. More students mean more work to the staff, but adding a peer-review process into a content of a course barely adds any work.

Studying the validity of peer-review can be straightforward. At its simplest it can be done by taking the combined results of a group’s evaluation of a given peer solution, and then comparing it to the teachers’ evaluation. The process can be fine-tuned to produce even more accurate results by using different algorithms. Loll et al. [6] have proposed counting weights to reviews and also compared the reviews to teachers’ reviews to estimate their knowledge of the subject. Increasing the number of reviewers and grouping reviewers evenly by the measured quality of their reviews may also be used to improve the quality and motivation of peer-reviewing [4, 9].

To get valuable reviews motivating the students is important. On students’ perspective, making the most of the process often depends on personal attitudes and opinions. At a glance, peer-reviewing can seem as extra burden for a student who feels his input mostly benefits everyone else, namely, the staff and the peers – not the reviewer himself. In case the student is committed to improve his skills only to pass the course requirements the claim can be valid regardless of the arguments for its use. It is hardly ever a good idea to simply force people to adopt a process and expect them to contribute in a meaningful and productive way. Instead, the key is to inform the participants of the purpose of the system, emphasize the benefits, assess the concerns involved with the process, and using tools that enable the process as unobtrusively as possible. To motivate the students to put effort into their reviews, the reviews can also be rated by the receiver (and/or the teacher). In fact, evaluating the quality of reviews can be a key factor in motivating students to give proper assessments.

The problem with the skill level differences among the students can substantiate in several ways. It can be argued that novices benefit the most from the process, whereas advanced students are only contributing while not getting much from their less adept peers. However, for the best learning outcomes, students should be exposed to good solutions, as it can be true that reviews of poor solutions mostly benefit the receiving author. Ranking the students by performance and arranging them in an optimal way is one way of fighting the problem of uneven reviewer groups.

It is not always clear how to label solutions good or poor. For example, a perfectly working solution can have hopelessly obfuscated source code while an apparently well-formatted and syntactically correct code can be riddled with logical mistakes and unacceptable features in its design (wasteful use of resources, blatant disregard of security and so on). Clearly, identifying and assessing problems like these can be completely out of reach for a novice. One method of dealing with this is to allow students to respond on their own level and use fine grained criteria in the reviews. Comments and ratings on different aspects of the submission can help them to analyze the solution with at least some success. E.g. Reily et al. [7] broke down the reviews in code formatting, conventions, documentation, interface design, and modularity.

Negative attitudes toward peer-reviewing can emerge from other reasons as well, such as feeling that the received feedback is of little use. People receiving only superficial or neutral comments may not find them helpful in improving their work, especially if it appears to be dishonest and undeserved. One source of this problem can be peers' reluctance to give critical feedback on others work. Reasons for this can be, for example, fear of retaliatory reviews, unwillingness to make critical judgments-based on lack of expertise, collusion among students, or plain laziness. The effects of "rogue reviews" are discussed in more detail by Reily et al. [7].

Students' tendency to overlook problematic parts in their peers' work can be addressed by several ways such as training, and showing examples of good reviews. The group's performance can be calibrated by having students to do preliminary reviews on example solutions prior to starting actual exercises. Studies discussing biased reviews also mention the importance of double-blindness in combating the effect of personal factors [8].

REVIEW PROCESS IN WEB PROGRAMMING COURSE

Deployment of the MyPeerReview-system was conducted in a Bachelor's level Web programming course (worth 3 ECTS) that concentrated on the basics of HTML, CSS, JavaScript, PHP and Ajax techniques. The students were expected to have at least some experience in programming and adequate skills in using web-based systems, but the preliminary knowledge of the students varied a lot. In addition to weekly practical assignments the course included a personal project on which we applied the peer-review process. The topic of the project was freely choosable by student but it had to be approved by the teacher. The guidelines were loose and offered a chance to use any suitable methods, as long as the project a) was done using web programming techniques, b) used a database, c) implemented user management in some form, and d) was set up online accessible from the university's network.

Participation to peer-review process was required in order to pass the course, but the students were informed it would not affect the final grades. The hours spent in peer-reviewing were, however, taken into account in the course work load. The submission of projects began immediately after the project deadline and because of the course schedule and the oncoming end of semester, the phase was very short, lasting only a day. In this phase the students were instructed to 1) register an account in the MyPeerReview system and 2) submit their project there. In the end, total of 24 students managed to complete the steps in time. The submissions included uploaded archive file of all of the source codes, project report having five pages or less, and a URL to the application with possible username and password combination(s) for testing purposes.

The review phase started moments after the submission phase closed. The entries were inspected only superficially by teachers just to confirm no files or URLs were missing. First, each of the 24 authors were assigned five randomly chosen peers for review. In their reviews the students were expected to read the project report, test each application online, download the files and assess various aspects of the submission. Sections with listed items were each rated individually by using a selection matrix. Since the projects contained several files and hundreds of lines of code on average, the reviewers were instructed to spend half an hour for each review and not to perform a line-by-line inspection, but to study the code just to get a sufficient view on its structure and design.

The review form can be divided in 3 major sections: program analysis, used techniques and overall score. The evaluation was constructed out of multiple-choice questions (mandatory) and open feedback as text fields (optional) so the students could explain the rationale behind their ratings. Numerical scales were labelled *Very good* (5), *Good* (4), *Acceptable* (3), *Poor* (2), and *Unacceptable* (1). An option (N/A) was included for unexpected cases in which reviews could not be performed, so the overall results would not be distorted by technical problems experienced by individuals. The overall score had no option for problematic submissions so the students were instructed to rate those Unacceptable (1).

The students were given nine days to complete the reviews. After the deadline the system was closed for any changes. The results were collected and authors were able to access the reviews of their work. By the end all the reviews of the 24 students were completed. In addition, we had four other students who participated in peer-reviewing process, but their reviews were not taken into account when analyzing the results.

After the students had had time to examine their reviews, we asked for feedback from the interface and other aspects of the peer-review process in a post-test survey. Comments, opinions, and ratings were collected regarding 1) usability and given instructions, 2) review questions, 3) anonymity, and 4) peer-review as a teaching method. 15 students replied with comments and we analyzed their responses. The results of the evaluation and survey are presented in the following chapter.

RESULTS

The feedback about the MyPeerReview system was promising and indicated that the basic design of the system was working short of the spotted bugs. Having some issues with forms and the interface design in the earlier test, we saw no failures in the student input this time. Most of the complaints concerned site navigation and some students had struggled with the interface and had to "click around" to find what they were looking for. Evidently, this did not prevent them from completing the tasks correctly. Although we experienced some problems due to bugs in the system, which we managed to fix on the fly, all participants were capable of registering and uploading their work through the web-based interface as instructed. The teachers' assessment of the system suitability was definitely accepting. With appropriate modifications and additions, the process can be adapted in programming courses for the purposes of teaching and analyzing student performance.

According to the survey the most useful aspect to the students in peer-reviews is textual responses with reasoned comments. Discrete analysis of features may be helpful too, but mainly as a method of quick technical assessment of the submission. In terms of learning outcomes the preference of properly explained textual assessments is clear. Even a handful of succinct comments can help when combined with other reviews. Even though the students had no choice but to take part of the peer-reviews, they reported they appreciated the opportunity to have a look at the variety of different approaches and receiving feedback. On the other hand the complaints also indicate that students prefer to receive written explanations from their peers. A very high level of participation and the positive feedback on the review process indicate the majority of students did not consider the peer-review too burdening.

While analyzing the overall grades that students had given each other we had one major challenge. As previous studies have proven, students seemed to evaluate the works gently. More than half of the students got the second best grade when counting the average of received peer-reviews, and there were no lowest grades given at all. Fig.1 compares the grades given by teachers and students.

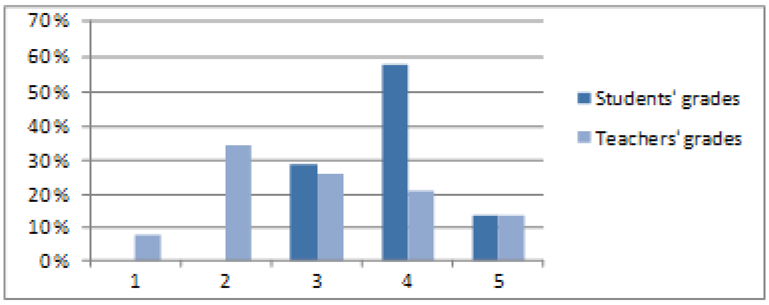

Fig.1. Grades of the assignment

Instead of normalizing the values we put the assignments in order based on average of given grades in peer-review and compared these results to teachers' evaluation. In addition, we also calculated the average grades using weighted base ratings. Instead of using the teachers' evaluation as a quality measure where the students who have completed their task better give more weighted grades introduced in [6], we counted this factor using the grades got from the peer-reviews. Our comparison pointed out that there was no remarkable difference in factors whether we were using either of these as the quality measure. Weighted evaluation slightly improved the results by reducing the difference of average between teachers' and students' evaluation. We believe that peer-reviewing can be used and improved for evaluation without teacher's participation when the evaluation pattern is well defined. When comparing the order between teachers' and students' overall evaluation of the assignments, the results followed each other. However, there were a couple of significant differences in evaluation between students and teachers. Based on re-examination of these assignments and comparing the overall grades to the technology-based evaluation, it seemed that these differences were a consequence of application layout and design, which were not listed as a part of rating arguments.

Although all the respondents felt that students' identity had not had an effect to the given and received grade, many of them still gave support to anonymity. We could not establish double-blindness in this test. Although we told the students to remove names from their peer-review submissions it turned out only a few of them did so. We believe the main reasons for this were that a) they did not bother to prepare two versions of the project (for the teacher and for the peer-reviews), b) many had set up their project on a personal web space (containing revealing addresses), and c) the applications were anyway publicly presented in the class a few days before the peer-reviews. The reviews, however, were anonymous.

CONCLUSIONS AND FUTURE WORK

Peer-review systems have not usually been used in context of programming and we found no systems which could have served our need, so we implemented such system. The results show that we have been able to implement an application that can be used in programming review process. This paper presented initial evaluation of the system.

Students found the use of peer-reviewing system for programming assignments useful. Feedback indicates that review of other's code and received comments are helpful. However, numerical feedback given by the students and teachers were clearly on different scales, which leads us to think that evaluation criteria must be defined more clearly and evaluations should also be rated. More attention must also be paid to valuable written feedback. This can be promoted by giving a change to evaluate the received review and by requiring the peer-reviewers to include certain aspects to their reviews.

To get evaluation that is usable for teachers evaluation arguments have to be split into pieces and asked separately from students, or the arguments have to be reported punctually in other means. Single overall score, which is easily based on feeling or emotion, is not precise enough, especially with inexperienced students.

REFERENCES

[1] Bouzidi, L., Jaillet, A. Can Online Peer Assessment be Trusted? Educational Technology & Society, 2009, Vol. 12, pp. 257-268.

[2] Diefes-Dux, H.A., Verleger, M.A. Student reflections on peer reviewing solutions to Model-Eliciting Activities, 39th IEEE Frontiers in Education Conference, 2009. FIE '09, Oct. 18-21, 2009, San Antonio, TX, pp. 674-680, ISSN 0190-5848.

[3] Garousi, V. Applying Peer Reviews in Software Engineering Education: An Experiment and Lessons Learned, IEEE Transactions on Education, May 2010, Vol 53, Issue 2, pp. 182-193, ISSN 0018-9359.

[4] Hundhausen, C., Agrawal, A., Fairbrother, D., Trevisan, M. Integrating Pedagogical Code Reviews into a CS 1 Course: An Empirical Study. ACM SIGCSE Bulletin, Vol. 31, Issue 1, pp. 291–295, ISSN 0097-8418.

[5] Hyyrynen, V., Hämäläinen, H., Ikonen J. MyPeerReview: An Online Peer-Reviewing System for Programming Courses. (submitted for evaluation, 10th Koli calling International Conference on Computing Education Research).

[6] Loll, F., Pinkwart, N. Using Collaborative Filtering Algorithms as eLearning Tools. Proceedings of the 42nd Hawaii International Conference on System Sciences, 2009. Jan 5-8, 2009, Waikoloa, Big Island, Hawaii, pp. 1-10.

[7] Reily, K., Finnerty, P. L., Terveen, L. Two Peers are Better Than One: Aggregating Peer Reviews for Computing Assignments is Surprisingly Accurate. Proceedings of the ACM 2009 international conference on Supporting group work, May 10-13, 2009, Sanibel Island, Florida, USA, pp. 115-124, ISBN:978-1-60558-500-0.

[8] Sitthiworachart, J., Joy, M. Effective Peer Assessment for Learning Computer Programming. ITiCSE '04, 2004. pp. 122-126, ISBN:1-58113-836-9.

[9] Søndergaard, H. Learning from and with Peers: The Different Roles of Student Peer Reviewing. ITiCSE '09, July 6–9, 2009, pp. 31–35, ISSN:0097-8418.

ABOUT THE AUTHORS

Ville Hyyrynen, Lappeenranta University of Technology, ville.hyyrynen@gmail.com

Harri Hämäläinen, Lappeenranta University of Technology, harri.hamalainen@lut.fi

Assoc.Prof.Jouni Ikonen, Dr.Tech, Lappeenranta University of Technology, jouni.ikonen@lut.fi

Prof.Jari Porras, Dr.Tech, Lappeenranta University of Technology, jari.porras@lut.fi