Preprint. Use following information for referencing: Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras. MyPeerReview: An Online Peer-Reviewing System for Programming Courses, 10th Koli Calling International Conference on Computing Education Research, Koli National Park, Finland, 28-31.10.2010, pp. 94-99. ISBN 978-1-4503-0520-4. doi:10.1145/1930464.1930481

MyPeerReview: An Online Peer-Reviewing System for Programming Courses

Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras
Lappeenranta University of Technology
P.O. Box 20, FIN-53850 Lappeenranta, Finland
Tel.: +358 5 621 11

firstname.lastname@lut.fi

ABSTRACT

In Lappeenranta University of Technology (LUT), there has been an interest in employing the peer-review process in context of teaching programming but previous attempts to find an appropriate platform have been unfruitful. In this paper we describe the considerations with regard to the design and implementation of a web-based peer-review system that enables the use cases involved with the process. We tested the system in a programming course to assess its suitability and usability from the students' point of view. The test confirmed the design to be working and laid out the groundwork for future development of the system.

Categories and Subject Descriptors

K.3.1 [**Computers and Education**]: Computer Uses in Education – *Collaborative learning*

General Terms

Design, Experimentation

Keywords

peer-review, teaching tools, programming

1. INTRODUCTION

Use of student peer-review has been studied in numerous educational institutions around the world. The principles of peer-reviewing are simple and easily adaptable for various contexts. 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. While reaching for higher quality in teaching with growing group sizes in the academic world, the promise of providing good results in an efficient way makes peer-reviewing even more attractive.

Our previous studies on using peer-reviewing for programming assignments have yielded poor results partially due to the lack of proper tools. The teaching tools currently used in LUT include the proprietary *Blackboard* [1] learning system among others but none of them offer desired functionality for handling the peer-review process. The results of trying to find a solution to this problem through customization of existing software have been unsatisfactory.

To answer this need we decided to design and implement a system that provides the required functionality. In this paper we describe the developed system. We also attempt to find methods to help to decrease the staff's workload brought on by the process, and study future directions for the development of the system.

2. CONSIDERATIONS FOR THE USE OF PEER-REVIEW IN PROGRAMMING

Reports and case studies that describe peer-review in the context of teaching programming usually end with positive conclusions. Peer-reviewing enables lecturers to measure the student performance through ratings, and it also puts students in a position to critically analyze the work of their peers. The chance of seeing different solutions can help the students to find new ideas and also view their own work more thoughtfully. This can be very beneficial in the context of programming where the quality of solutions often measures against several factors instead of one objective truth [15, 8, 16]. In terms of learning outcomes the exposure to different programming styles and approaches can progress both authors' and reviewers' skills when incompetent features and functions are being discovered and constructively criticized, and competent features can be pointed out and learned from.

Studies [15, 16] discuss the issues involved with the process and present methods to improve the accuracy of peer assessment [8] through using algorithms. In the next sections we list some of the systems used in peer-review studies, and explain the chosen approach in our case.

2.1 Existing Systems

There are currently several systems designed for managing user submissions for the purpose of student assessment. Suitability of *MyReview* [11], a web-based open source conference paper review system, was studied [6] in LUT earlier while organizing peer assessments but was found lacking in terms of usability. Systems for similar type of academic use include *ConfTool* [2] and *Precision Conference* [14], but both reserve all rights to the software and offer license-based solutions to organizations.

The University of Melbourne utilizes web-based *PRAZE* in their teaching in a variety of learning environments. It is currently being used and developed only at Melbourne, and studies [13, 17] have been

Preprint. Use following information for referencing: Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras. MyPeerReview: An Online Peer-Reviewing System for Programming Courses, 10th Koli Calling International Conference on Computing Education Research, Koli National Park, Finland, 28-31.10.2010, pp. 94-99. ISBN 978-1-4503-0520-4. doi:10.1145/1930464.1930481

published regarding the system and the use of peer-review. The system enables students to submit their work in form of an essay, a report, multimedia content, or any other type of file into the system for peer-review. Students can work as individuals or groups, and they can also rate the reviews according to given criteria. The lecturer can allow "group self assessment" to let the peers assess the contribution of the other members in the group.

PeerWise [3] is a peer-review system administrated by the University of Auckland. The system allows students to submit answers to questionnaires created by their peers on given topics. The platform supports discussion and evaluation of the student-created questionnaires. Students are expected to be self-reliant in using the system – after initial setup the staff's involvement with the process is minimal. Using PeerWise is free but it requires contacting the administrators.

Moodle [10] is an open-source course management system (CMS) targeted for educational use. Lykourentzou et al. [9] describe its use as a platform for a peer-review process in “an introductory level e-learning course on Web Design”. They found the presented system user-friendly and providing the necessary functionality to deliver the procedure of peer assessments. Using the modules for peer reviewing requires installation of the whole Moodle platform and was therefore rejected in our case.

Another open-source peer-review system is *iPeer* [7]. It is aimed for filling evaluations but lacks the possibility to upload files for evaluation. Therefore it was left outside of our study.

2.2 Considerations for System Design

We aim to minimize problems with well-structured user interface and simplify the process to enrich the student’s user experience. The design should enable the use cases involved with the process as efficiently as possible. Pivotal use cases for a student include at least registering a user account, submitting own work, and reviewing a submission.

Taken the time allotted for a review task, students should be spending proportionally less time dealing with system-related issues and more time doing exercises and reviewing others. We emphasize the importance of the aspects of **usability** based on Nielsen’s usability study [12] in order to improve student performance and learning outcomes. As almost all studies discussing the subject concur, **anonymity** of the student submissions should be the default – unless there is a reason for revealing identities, names or aliases are kept hidden. Submissions, however, can expose a unique identifier so that they can easily be referred to. For the staff, the system should provide **automation** of mechanical tasks (e.g. calculations, administrative actions) otherwise done by hand and also operate as a submission platform. Lastly, the system should be **open for development** by anyone and provide platform for unforeseen uses.

Problems with the tools and methods for conducting the process can sometimes defeat the gained benefits. A slow and/or poorly designed system wasting everyone’s time frustrates users quickly, which may also affect the quality of submissions. In order to have a positive effect on the user experience the system providing the framework for the process should be, at least:

- highly available by minimizing the requirements needed to access and use the system
- simple to use to minimize the impact of having to learn a new system
- able to perform well under load
- capable of handling volumes of data
- scalable to accommodate a reasonable number of courses and hundreds of users

In other words, the overhead caused by the behavior and the user interface of the system should be cut as short as possible. Low overhead is essential for any system since most of its student users are unlikely to spend any more time than necessary using it for completing the tasks. Besides, we realize many of the students may use the system only once or twice in their studying career so that time should be spent as efficiently as possible.

To lower the threshold of adapting the system is to make extensive use of already existing solutions, technology, and learned usage patterns. The system should look and feel like an ordinary website, and use HTML in its frontend with **optional** JavaScript and Ajax enhancements. Users should be able to apply their familiarity with other web-based systems (registration scheme, forms) as directly as possible.

We decided not to use Java-applets or build of extensions. Instead, students should be able use the system right away without installing applications or add-ons on standard browsers. These considerations are reasonable because both backend (the LAMP stack) and frontend (modern browsers) of the system provide sufficient tools to implement the required functionality. Installation and deployment of the system should be lightened by using a popular open source backend.

Given these considerations, we decided to discard the choice of altering existing peer-review/educational systems to meet our requirements. Instead, we built one from scratch by making extensive use of open source components. For its base we chose the *Drupal* [4] Content Management Framework (CMF) and built the system on top of that.

2.3 Drupal CMF

Drupal is a highly customizable, open source platform written in PHP. It has a modularized architecture that separates the database, the content, access management, layout structure, and page rendering in separate layers. There are currently hundreds of contributed modules extending the core functionality covering areas such as content display customization, e-Commerce implementation, enhanced site administration tools, and advanced user management. Drupal has succeeded in gathering an active and growing base of users developing its code, and also providing assistance with using the system on an open forum.

Preprint. Use following information for referencing: Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras. MyPeerReview: An Online Peer-Reviewing System for Programming Courses, 10th Koli Calling International Conference on Computing Education Research, Koli National Park, Finland, 28-31.10.2010, pp. 94-99. ISBN 978-1-4503-0520-4. doi:10.1145/1930464.1930481

Choosing a tested and supported framework as base of the system has the obvious advantage of making use of already written and maintained code. Setting up a Drupal site requires only minimal effort from a skilled developer, and it can happen in moments. Therefore, the initial steps of building a special-purpose web service can be taken very far without writing or altering code.

Functionality that is unattainable using the available (core or contributed) modules can be implemented by building custom modules. They can be created either from scratch or by branching off an existing project. The Drupal system and its APIs are extensively documented, allowing customization down to the minute detail without altering the original code. This, however, also means the system is very complex and time-consuming to learn, and developing for the platform requires expertise in both web techniques and the Drupal system.

3. SYSTEM DESCRIPTION

MyPeerReview introduces a custom built module to Drupal to provide special functionality, and to alter the behavior of the other modules. The system also contains a simplified theme that provides a cleaner interface. The system has been tested on the LAMP stack using the latest versions[1] of common browsers. Although the underlying framework allows virtually any modifications there are some restrictions on how courses and exercises can be set up and run.

The system consists of five distinct content elements: *course, exercise, solution, review form,* and *review*. The elements map to the underlying Drupal framework so that they can be handled as native data structures. However, the peer-review process requires bookkeeping of special relationships of elements (such as solution–review as one-to-many) so those are stored and handled as records in their own tables.

In MyPeerReview several courses can be set up, each having its own group of students completing exercises. A student can take part in any number of courses. All exercises that use peer-review contain a reference to a designated review form which is used to submit the reviews. Thus, a new, blank form must be created for each such exercise. Using the administrative tools the teacher can assign review tasks for any combination of the course participants regardless of whether or not they submit their own solution.

The system was designed to enable the use cases involved with running the process of peer-review. The intended process and its consequent activities are described next.

3.1 Outline of the Process

The peer-review process is essentially a procedure that is repeated for each of the exercises. The procedure can be divided into two phases, (1) uploading of solutions and (2) review of the solutions that can be broken further down into consecutive steps. The teacher controls the phases exclusively and can switch between them as needed. The details of the process are listed below. Initially, the exercise status is **on hold**. Neither submitting nor editing is allowed when the exercise is on hold. The final phase (*Complete)* marks the end of the exercise, and allows students to access the reviews from that point on.

1. Teacher opens the exercise **for solutions**
2. Students submit their solutions
3. Teacher sets the exercise **on hold,** and
   3.1. examines and accepts the submissions
   3.2. assigns review tasks
4. Teacher opens the exercise **for reviews**
5. For every assigned review task, each student
   5.1. fetches the solution
   5.2. evaluates the solution
   5.3. submits the review
6. Teacher sets the exercise **on hold,** and
   6.1. examines and accepts the submitted reviews
7. Teacher **completes** the exercise
8. Students access the reviews they received

In the process the student is expected to navigate through a certain path on the site. The path is depicted in figures 3 and 4 that also show the start and end points for the two phases (thick arrows guiding the expected path). The starting point can vary depending on certain circumstances explained further below. The ending point is also only suggested – the phase is considered complete for a student when all of his or her tasks are submitted. Each box represents a page load that can be reached either via a URL (HTTP GET) or form submission (HTTP POST).

3.1.1 Submission of Solutions Phase

Figure 1 shows the path to navigate through in the process of submitting a solution. The student returns to the home view after submitting and sees the options for logging out, and viewing and editing the submission within given time limits, if said actions are allowed. The figure shows just one possible way of completing the task; some steps may be skipped depending on the circumstances, such as when the student has already registered (or logged in), (s)he skips the registration (or login) part.

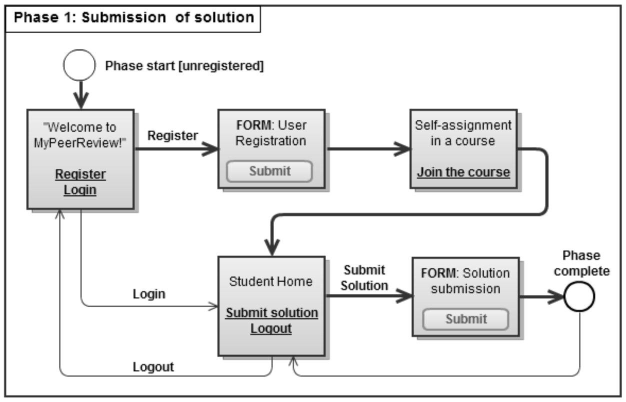


Figure 1: Phase 1 from student's point of view

[1] Firefox 3.6, Internet Explorer 8 and Google Chrome.

Self-assignment is used to reduce the staff involvement in the process. However, the teacher can, of course, assign students to courses manually. If self-assignment is not used, accounts must be either pre-created by the teacher, or validated as the students register in the system by themselves. These options may come into question in courses whose participants are less adept in using web-based systems, and/or if tighter control over user access is preferred.

3.1.2 Review Phase

Figure 2 describes the review phase for the student, in which (s)he simply goes through all of his or her assigned reviews until all of them are submitted. After submitting the reviews, they can be edited until the phase is closed by the teacher.

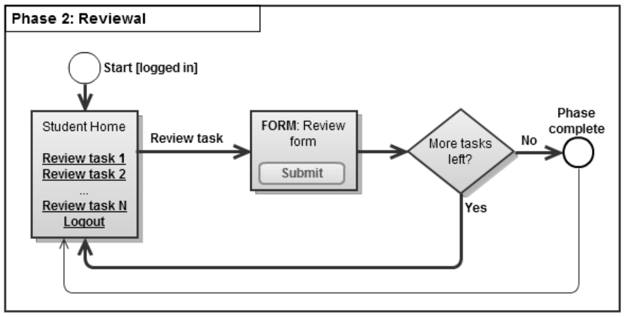


Figure 2: Phase 2 from student's point of view

The students interact with the system during the phases 1 and 2, but after reviews, the teacher may allow threaded, anonymous, and staff-moderated discussion between the authors and reviewers under each solution, visible only to the counterparts involved. This allows authors to ask clarifying questions and defend their solutions.

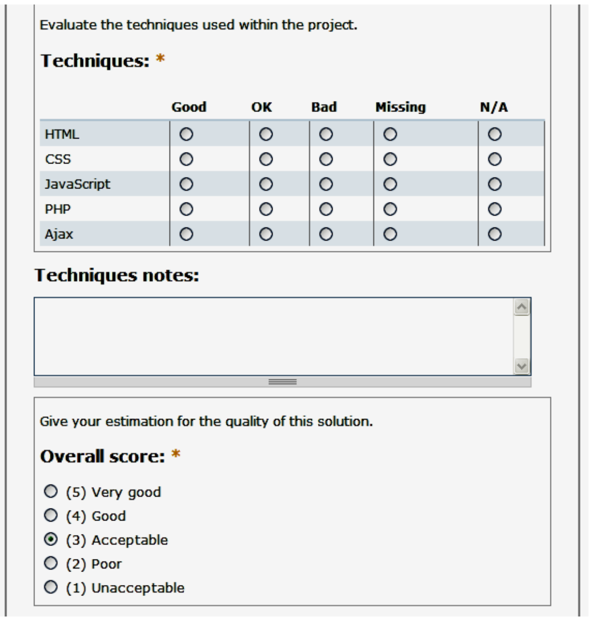
If there is a need for review of the *revised* submissions, the process can be repeated. This can be done by creating a separate exercise for the revised versions, or by allowing students to edit their submissions based on feedback. However, we did not test or plan the revision cycle in this version.

3.2 Student's Interface

The system has a relatively simple scheme for student navigation. We attempted to minimize the number of views to the system and gather all the central links and resources in one page, the student home view.

The courses and exercises have their own pages although they are probably less important to the students. They contain information that is most likely available on the official site and known by the students already. The system allows, however, using the course/exercise pages as the main source of information the teacher can equip with file attachments and HTML-formatted instructions etc.

Figure 3 presents some of typical review form components created with Webform module of Drupal. The shown form contains a matrix of selections, a list selection, and free text fields. Selection groups can be set to appear either as checkboxes (multi-select) or radio buttons (single-select). Other types of components that can be included in the submissions are hidden data fields, timestamps, grouping fieldsets, and file attachments, and they can be set either mandatory or optional. Lengthy forms can also be split by inserting page breaks if needed.



Evaluate the techniques used within the project.

Techniques: *

	Good	OK	Bad	Missing	N/A
HTML	○	○	○	○	○
CSS	○	○	○	○	○
JavaScript	○	○	○	○	○
PHP	○	○	○	○	○
Ajax	○	○	○	○	○

Techniques notes:

Give your estimation for the quality of this solution.

Overall score: *

- ○ (5) Very good
- ○ (4) Good
- [✓ (3) Acceptable]
- ○ (2) Poor
- ○ (1) Unacceptable

Figure 3: An example review form that contains a radio-button matrix, free text field, and a selection list

To reduce the required steps in small assignments for reviewing, the system is capable of displaying highlighted source code in a pop up. We did not test this functionality with students at this time due to the project arrangements. Running or compiling the uploaded code is inhibited because of security reasons. Executable applications are supposed to be run in external server dedicated for this purpose.

4. TESTING AND DEPLOYMENT

The presented peer-reviewing system for programming courses was tested on a Web programming course. During the course the students studied HTML, CSS, JavaScript, PHP and Ajax techniques, which indicate that single assignment submission has generally multiple files. The system was used for two assignments the first of which was during the system development and the second one was the final assignment for the course. The final assignment had free topic, but it had to demonstrate technologies studied during the course, including database and user management.

The student interface in MyPeerReview was designed so that minimal set of instructions would be necessary. However, information was provided also via email and the course web page. System was designed so that reviews could be done anonymously, but the feature was not forced so that students did not have to remove author information from their assignments.

Prior to starting the peer-review process, the students had to return their project file to a course assistant for official grading via email. The

Preprint. Use following information for referencing: Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras. MyPeerReview: An Online Peer-Reviewing System for Programming Courses, 10th Koli Calling International Conference on Computing Education Research, Koli National Park, Finland, 28-31.10.2010, pp. 94-99. ISBN 978-1-4503-0520-4. doi:10.1145/1930464.1930481

teacher created evaluation questions, which students has to answer. The peer-review submission phase started immediately after the assignment deadline. For the process students 1) created user accounts to the system, 2) self assigned to the course and 3) submitted their project there. The submissions included a URL to a web server where the project could be tested, together with needed user names and passwords.

After the submission phase each author was assigned five to six assignments to review and they were instructed to use no more than half an hour per review. After reviews teacher analyzed the returned reviews and allowed students to see the reviews which their assignment had received.

The system and its behavior was analyzed by asking students to fill out a survey about whether students felt that reviewing tasks had taught them, if they had received helpful review information, how usable the system was, and some analysis information the system collected (e.g. how much time a student used from opening a task to finalize his/her review). Generally many students found peer-reviewing to be very useful and the system usable. However, there are many issues which can be improved. More extensive description of the results is available at [5].

5. FUTURE WORK

During the development we noted and discussed different ways to enhance the performance and usefulness of the system. Below are listed some of the points that we think are relevant in the tested version.

- Design and implementation of:
    - Support for ratings of reviews
    - Summary views of received feedback for students
    - Automated timing of courses and exercises
    - Automated email notifications of new tasks
- Application of algorithms, specifically in:
    - Distribution of review groups based on skill
    - Weighed ratings
- Improved (programmable) views over gathered data
- Proper assessment of privacy issues in the system (access control)
- Support for group organization
- Better approach in organizing review tasks
- Enhanced students input forms for wider variety of material (multimedia etc.)

Aspects of integration with other environments were left open, but then again, Drupal offers a number of already developed mechanisms to feed formatted (XML or other structured) data out from the system. Another open issue was rethinking of the user management scheme to employ identification by using already existing (and verified) accounts. This would further cut the system-related overhead and raise the system's readiness. In addition to increased data security, outsourcing of user data could properly address other issues as well such as the concerns involved with gathering and storing records of personal information.

One of the questions related to integration is whether the submitted tasks in association with received and given reviews could be used for further purposes after the task and even the course is completed. Some students are already constructing their personal portfolios based on their achievements and traditionally peer-reviews have played a remarkable role e.g. in teacher portfolios. Technology and integration itself do not necessarily provide the greatest challenge as long as the appropriate electronic system for portfolio maintenance is being selected and the interfaces are defined and implemented, but it influences in fact to the whole peer-review process starting from the creation of the form for peer-reviews. The output and feedback the students get have to be disposable for external estimation or at least somehow adaptable to that.

6. CONCLUSION

In this paper we have presented a web-based system for peer reviewing called MyPeerReview which we have tested and deployed in a web programming course in LUT. The developed system is essentially a highly customizable platform for administrating users, and their submissions that can be reviewed. Therefore, although the goal was to design a peer-review system for programming courses and the demands those types of courses set to the process in particular, it should also be usable in other contexts as well.

One shortcoming of the tested process was that the comments and suggestions could not be used to revise the submissions, or use the gained insight in subsequent exercises. Another missing aspect was rating of the reviews. Evaluation of reviewer performance is an important facet in the process because it can motivate students to submit better reviews, and the ratings can be later used in algorithms that improve the accuracy of peer-reviews whereas the written comments can be used to develop the personal programming skills.

From a teacher's point of view, the group can work as a filter for clearly bad solutions having missing or corrupted files, serious security holes etc. We found that while performing the reviews, a number of fatal or obvious mistakes, such as corrupted files and missing features, were spotted in the submissions as expected. As the problems were reported, they were quickly addressed so that the reviews could resume. By letting the students do the mechanical inspection, the staff can then simply verify the group's findings and act accordingly, i.e. notify the author or take the error into account in the final score. This may be particularly beneficial in case of large groups and already burdened staff.

Although the attitudes and experiences of the students concerning peer-reviewing and the application were positive, the test and the survey revealed aspects in the developed system that definitely need attention and possibly redesign. The students were mostly happy with the structure and length of the review form. If ratings are used the students should be also given points of reference to see how they managed compared to the others. Also, the scales and criteria in the review form

Preprint. Use following information for referencing: Ville Hyyrynen, Harri Hämäläinen, Jouni Ikonen and Jari Porras. MyPeerReview: An Online Peer-Reviewing System for Programming Courses, 10th Koli Calling International Conference on Computing Education Research, Koli National Park, Finland, 28-31.10.2010, pp. 94-99. ISBN 978-1-4503-0520-4. doi:10.1145/1930464.1930481

should be similar to those used by the teachers and well-defined to make analysis straightforward. Regardless of the issues, we were able to verify that the basic design allows organization of the process with a minimal number of input errors.

Since one of the original objectives was to implement an open-source solution, we have shared our work for further development[2]. We have presented some of our ideas that should be implemented to make the application more advanced and tempting.

7. REFERENCES

[1] Blackboard Learning System. [Online] http://www.blackboard.com.

[2] ConfTool: Conference Management Software. [Online] http://www.conftool.net.

[3] Denny, P., Luxton-Reilly, A., and Hamer, J. The PeerWise system of student contributed assessment questions. In *Proceedings of the Tenth Conference on Australasian computing education* - Volume 78. Wollongong, NSW, Australia. PP 69-74.

[4] The Drupal CMS. [Online] http://drupal.org.

[5] Hyyrynen, V., Hämäläinen, H., Ikonen J., and Porras, J. 2010. Experiments on Applying Peer-Review Software for Programming Assignments. In *Proceedings of the International Conference on Learning and the Knowledge Society* (Riga, Latvia, August 26-27 2010). e-Learning'10.

[6] Hämäläinen, H., et al. 2009. Use of Peer-Review System for Enhancing Learning of Programming. *Ninth IEEE International Conference on Advanced Learning Technologies* (Riga, Latvia, July 15-17, 2009).

[7] iPeer. [Online] http://burrito.olt.ubc.ca/trac.

[8] Loll, F. and Pinkwart, N. 2009. Using Collaborative Filtering Algorithms as eLearning Tools. In *Proceedings of the 42nd Hawaii International Conference on System Sciences* (Waikoloa, Big Island, Hawaii, USA, January 5-8, 2009).

[9] Lykourentzou, I., et al. 2009. An Online Peer Assessment Platform for e-Learning Environments. In *Proceedings of The 5th E-learning Conference* on *Learning and the Knowledge Society* (Berlin, Germany, August 31 –September 1, 2009). e-Learning'09.

[10] Moodle.org: open-source community-based tools for learning. [Online] http://moodle.org.

[11] MyReview. [Online] http://myreview.sourceforge.net.

[12] Nielsen, J. Usability 101: Introduction to Usability. Jakob Nielsen's Alertbox, August 25, 2003. [Online] www.useit.com/alertbox/20030825.html.

[13] Pearce, J., Mulder, R. and Baik, C. 2009. Involving students in peer review: Case studies and practical strategies for university teaching. The University of Melbourne. 2009.

[14] Precision Conference Solutions. [Online] http://www.precisionconference.com.

[15] Reily, K., Finnerty, P. L. and Terveen, L. 2009. Two Peers are Better Than One: Aggregating Peer Reviews for Computing Assignments is Surprisingly Accurate. In *Proceedings of the ACM 2009 International Conference on Supporting Group Work* (Sanibel Island, Florida, USA, May 10-13, 2009). GROUP '09.

[16] Sitthiworachart, J. and Joy, M. 2004. Effective Peer Assessment for Learning Computer Programming. *ACM SIGCSE Bulletin* 36, 3 (September 2004), 122 – 126.

[17] Søndergaard, H. 2009. Learning from and with Peers: The Different Roles of Student Peer Reviewing. In *Proceedings of the 14th annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education* (Paris, France). pp. 31-35.

[2] http://sourceforge.net/projects/mypeerreview/ - MyPeerReview –project.