PRISM: A PEER REVIEW MANAGEMENT SYSTEM FOR PROGRAMMING ASSIGNMENTS

Dr. Mike Morrison, University of Wisconsin-Eau Claire, morriscm@uwec.edu

ABSTRACT

This paper describes the development and use of PRISM, a Web-based student peer review system for evaluating MIS programming assignments. PRISM supports open, single blind, and double blind reviews of programming assignments in a variety of different development environments The paper describes the PRISM architecture, explains how the system has been used in a variety of different MIS courses, and discusses the advantages and disadvantages of its use.

Keywords: online web based peer review system

INTRODUCTION

When teaching programming classes, the instructor often learns as much or more about the programming environment and programming techniques than the students he or she is teaching. Sometimes this learning comes from helping a student track down a particularly difficult program syntax or logic error. Sometimes it comes from observing a new way a student approaches an old programming problem. To take advantage of this knowledge gained through helping others, it follows that collaborative learning occurs through requiring students to evaluate each other's assignments. This approach can also improve the quality, quantity, and timeliness of assignment feedback, allow students to develop evaluation skills; and possibly lighten the instructor's grading load.

Potential problems exist with peer reviews however. For example, will students collaborate to cheat on the reviews and attempt to inflate their scores? How can an instructor motivate students to perform a thorough, rigorous review? Finally, does managing peer reviews add to the instructor's workload rather than lighten it? A peer review management system must address these problems to be successful.

This paper reviews existing peer review management systems, describes the architecture of PRISM, a Web based peer review system, and discusses the author's experiences with using PRISM in a variety of programming courses during the past four years.

EXISTING PEER REVIEW SYSTEMS

A number of online peer review management systems exist (e.g., 1, 2, 3, 4, 5, 6). Most support peer reviews of English composition assignments. One, the Peer Grader (2), allows peer review of computer programming assignments. Several systems have students assign scores to the work they are reviewing. One system was used with an upper division software engineering class (6), but supported documentation reviews rather than computer program reviews. All of the cited systems were designed to allow several students to review each assignment. In general,

evaluations of these systems report high user satisfaction levels with the process (2, 5, 6), and improved learning (2, 4, 5).

The PRISM system differs from other peer review management systems in several important ways. First, it was been specifically developed for reviewing programming assignments, although it can be used for writing or other types of assignments. Second it allows an instructor to choose the level of anonymity required for a particular review. Third, PRISM was not designed to tabulate scores assigned by student reviewers. Instead, it allows the instructor to view review comments and then use them to help the instructor assign the score. Fourth, PRISM is designed to require each student to review one and only one other assignment for each assignment.

PRISM ARCHITECTURE

PRISM allows students to turn in their assignments in personal folders stored within a shared class folder. Figure 1 shows a typical folder structure.

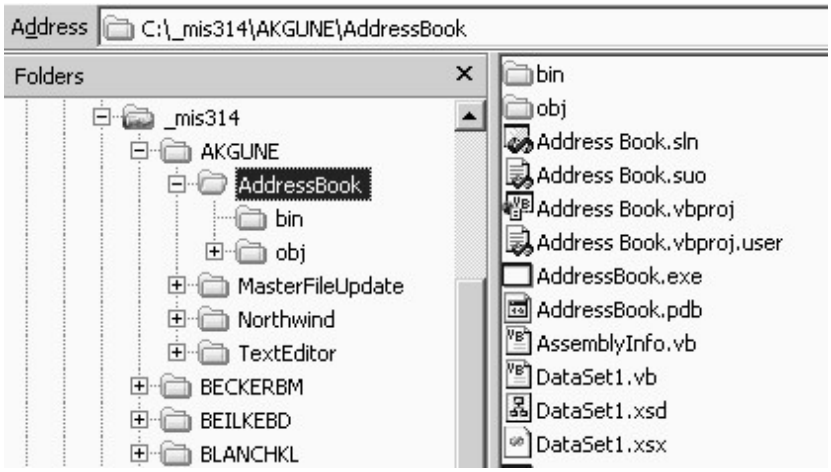

Figure 1: Student Files

The instructor creates a parent folder, which in this case is named _mis314. The instructor also creates a personal folder for each student. In this example, each student's email username serves as his or her folder name. The actual mechanism by which the students submit their work can be through mapped drives to the personal folders, and/or through FTP accounts which allow them access to their personal folders. Either approach requires setting shares and security permissions appropriately for each student.

In the PRISM review process, the instructor first logs on to the system, and enter the assignment information. Next, the instructor writes the evaluation questions on which the students base their reviews. A Web interface allows the system to display questions and responses using common HTML form elements such as single line text inputs, multi-line text areas, vertical radio inputs, or horizontal radio inputs.

After the instructor creates the assignment and questions, he or she then makes the reviews available to the students. When a student logs in to the system, he or she downloads the program files to be reviewed, and submits a review. PRISM notes and time-stamps these actions, which creates reports to flag late reviews, and flag students who download files for being reviewed prior to turning in their own assignment. (This would be considered cheating, because the

student can then base their own work on the work they are reviewing). The time-stamps on the files the students turn in are also used by the system to generate reports on assignments that are turned in late.

Figure 2 shows the Administration Menu and associated administrative Web pages in the peer review system. Menu selection number 4 randomly assign reviews to students. The system is

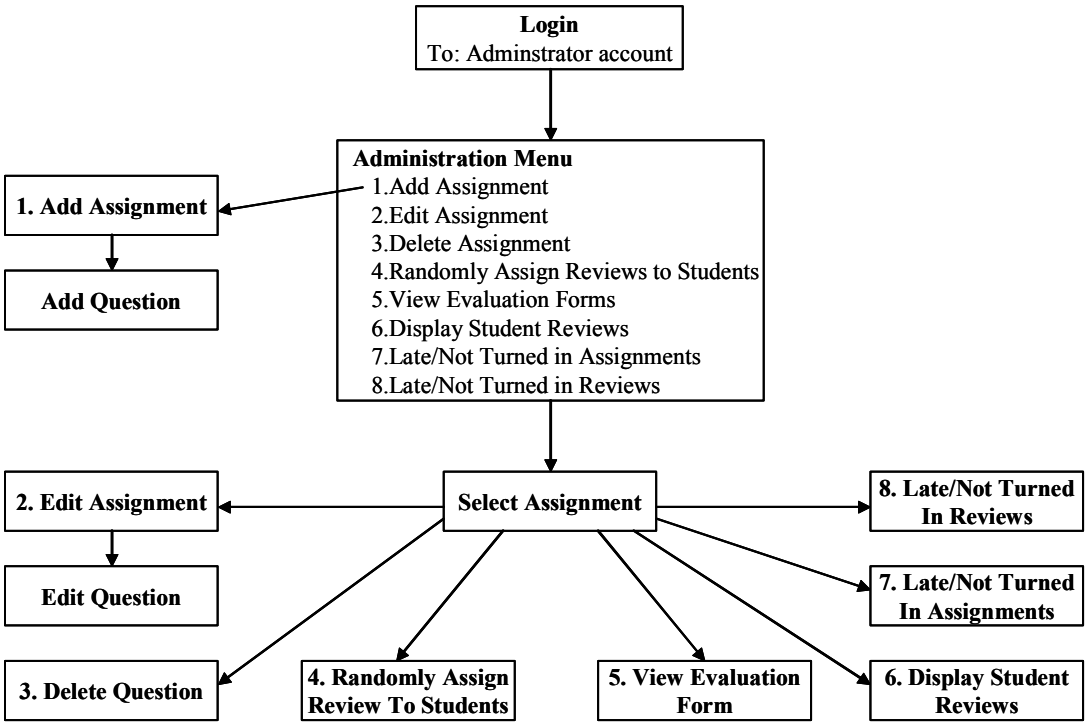

Figure 2: Administrative Web Pages

designed with the idea that each student reviews the assignment of one other student in the class for a particular assignment. The random assignment ensures anonymity when desired, and adds a degree of fairness to the system. Typically, most programming assignments come close to meeting assignment requirements. In this case, completing a review consists of pointing out the minor errors and omissions. For every assignment, however, a few are poorly done and require more time and effort to review. Random review assignment doesn't guarantee a student won't get two poor assignments to review in a row, but this feature at least satisfies students that they aren't being singled out in any fashion.

PRISM stores all of data in an Access database using the schema shown in Figure 3. The instructor must initially create user accounts for all students. Assignments are then entered into the system by specifying the path to the student files. (In the example in Figure 1, this would be c:_mis314.) Next, the instructor specifies the assignment folder, which in Figure 1 could be AddressBook, MasterFileUpdate, Northwind, or TextEditor. (Each of these folders represents an assignment that students in a class have completed.) The Assignments table stores the time when the programs are to be turned in, and the time that reviews are to be completed. Other assignment data specifies whether the system will use a single or double blind approach to anonymity, or be completely open where the system users know who they are evaluating and the

people being evaluated know who is doing the evaluation.

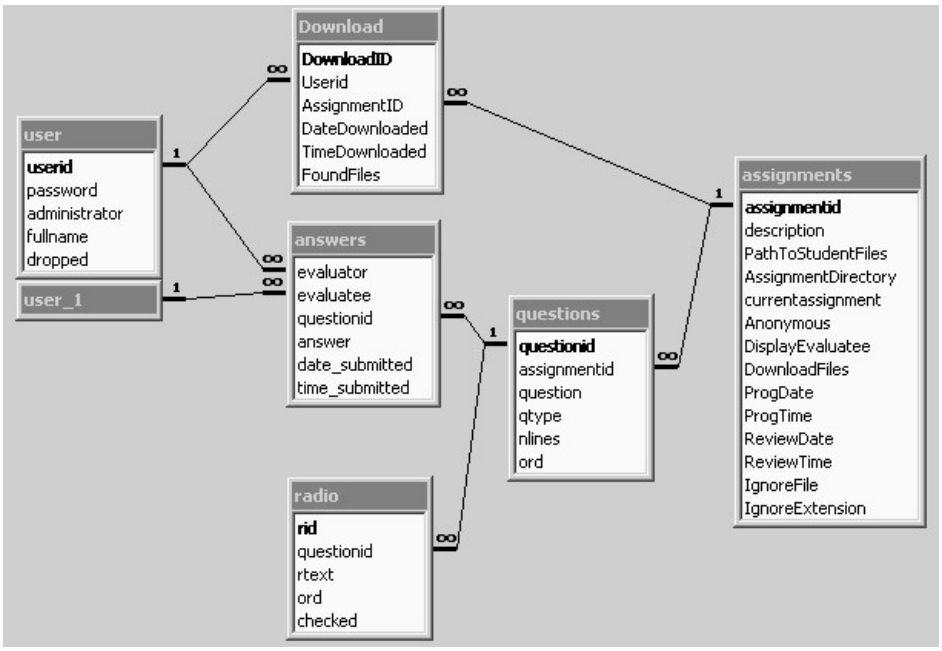

Figure 3: Database Schema

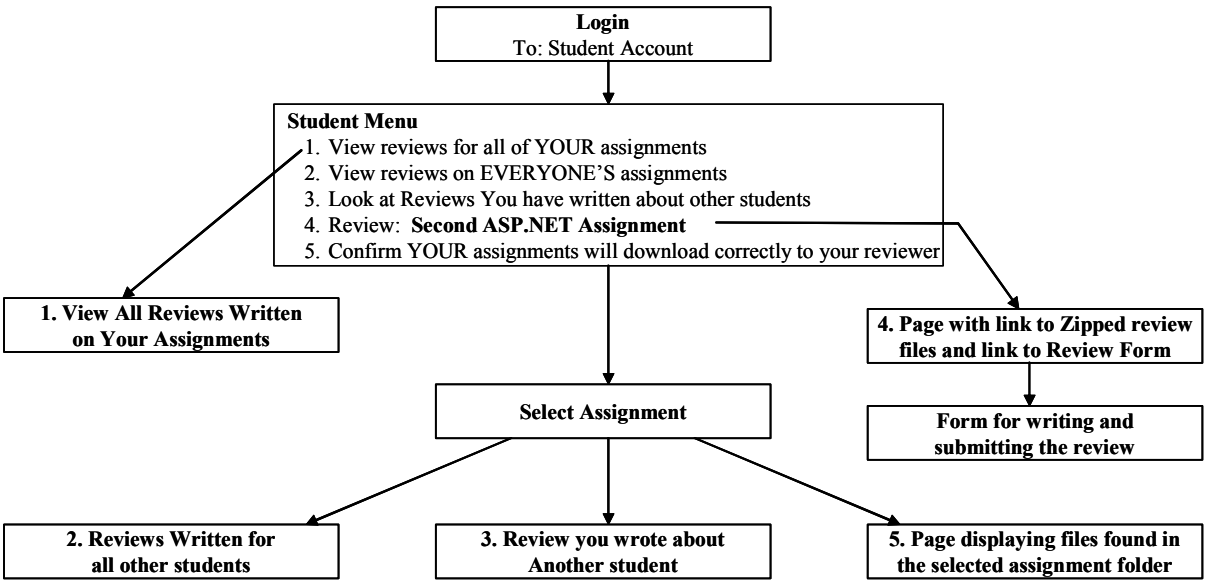

Figure 4: Student Menu and modules

Figure 4 shows the PRISM Student Menu and associated Web pages. A student can view all reviews that have been written about their work, as well as refer back to reviews they have written about other student assignments. The system also allows students to see reviews written about other students. When this is accessed, and if single blind or double blind reviewing is selected for the assignment, the system strips off the names of the students reviews and

randomizes the order in which the reviews are displayed. This prevents students from guessing whose review they are seeing by an alphabetic ordering of the reviews.

Since the review system depends on students turning in their work online and into folders named exactly as specified, the system provides a module to reassure them that their assignment folder is correctly named and that the files they turned in are actually in this folder (Module 5 in Figure 4). When a student has turned in his or her assignment and is ready to review another student's work, he or she can access Module 4. This process automatically compresses the review files into a single zipped file to make downloading faster, and to avoid problems inherent with a Microsoft Web server related to attempting to download executable files. For example, a Microsoft Web server doesn't allow downloading ASP and a variety of other Web server types of program files. Since these are exactly the files needed for reviewing, zipping them bypasses the problem.

Figure 5 shows the menu that the student sees after logging in to the system and figure 6 shows an example of what the form looks like with a single blind review.

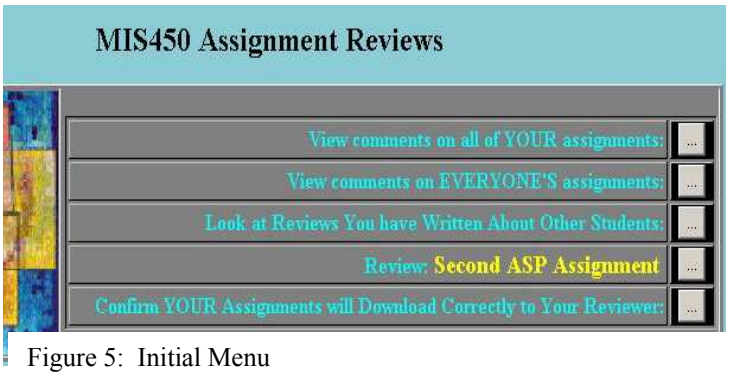

Figure 5: Initial Menu

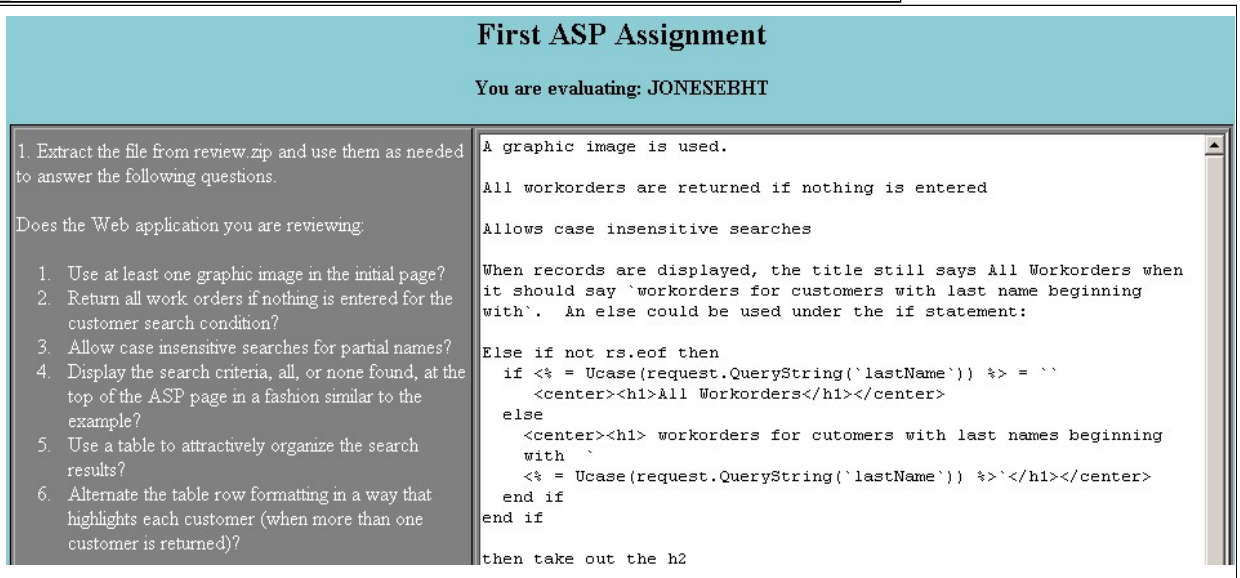

Figure 6: Student Review Form in Single Blind Mode

When a student begins filling out the Web based review form they have the option of submitting the form with it partially completed and returning to it later to complete it. When it is submitted,

whatever is currently entered on it is stored in the answers table and when the form is accessed later, any existing answers are written back to the form for editing.

PRISM EXPERIENCES

PRISM was first used during the Fall 2000 semester in a senior level Web programming class. A year later it was introduced in an intermediate level programming class. It continues to be used in both of these classes and has also been used in one Introduction to MIS class. In all of these classes, about 30 minutes of class time serves to introduce the students to the system and explain how to use it. No other time is spent to explain or train students on the use of the peer review system. In general, problems with using the system haven't occurred, which attests to its end user simplicity. Although the system can be used for other types of reviews, to date the reviews have related to programming assignments or to technical assignments using spreadsheets and personal databases.

Students usually require less than an hour to complete a review. Although the review questions are entirely up to the instructor, questions generally have a component that requires the reviewer to explain how to correct the problems they discover. In many cases, the student will review a better program than the one he or she submitted, and learn from it. Students often compliment the person they're reviewing in the review. In other cases, they learn what not to do while debugging someone else's assignment. During class discussions and individual discussions, students seem to be satisfied with the peer evaluation process.

From the instructor's perspective, PRISM requires additional work in terms of setup and configuration. PRISM also initially requires extra grading efforts since students also must be encouraged to make fair and thorough reviews. The current approach is to make the review part of the assignment worth 30% of the overall grade. The 70/30 split is arbitrarily chosen, but its purpose is to put a significant weight on the review, and let the students know that they are to take it seriously. Typically, it takes two or three assignments to convince everyone in the class that a cursory review isn't acceptable. During this period and later in the class, students are reminded that their assignment scores will eventually be based solely on the student evaluations of their work. Once student reviews are used to assign grades, the grading workload eases.

Students are understandably concerned that a review might be inaccurate or overly harsh. When their score is based on what they believe to be an unfair review, they are told to send the instructor an email message detailing exactly where they think the review is inaccurate. At that point, the instructor reviews the assignment and compares it to the peer review. Adjustments to scores are sometimes made – including to the review score of the person writing the review.

Although a formal experiment is needed to test this, four years experience with PRISM lends support to the idea that peer reviews are a valuable collaborative learning tool for programming assignments. By automating much of the process, the instructor workload has remained similar to what it was without using reviews, however the time distribution of the work load has shifted to requiring more work early and less late in the course.

FUTURE DIRECTIONS

It is possible that an instructor's overall grading work load can be reduced if peer reviews are extended by having several students review and score each assignment. As before, students would be allowed to ask the instructor to review what they believe is an inaccurate or overly harsh score. To test this, PRISM will be modified to allow each assignment to be reviewed by several students, have a score assigned to the assignment by each student, and average the score.

An experimental evaluation of the system is planned during the next year to see if there is statistical support for the belief that using PRISM for peer reviews improves the learning process.

REFERENCES

1. Chapman, O., Fiore, M. (2003). Calibrated Peer Review, A Writing and Critical Thinking Instructional Tool, White Paper. http://cpr.molsci.ucla.edu/
2. Gehringer, E. F. (2001). "Peer review and peer grading in computer-science courses," SIGCSE 2001: Thirty-Second Technical Symposium on Computer Science Education," Charlotte, Feb. 21-25, 2001, pp. 139.143.
3. Gousseva, J. (1998). "Literacy Development Through Peer Reviews in the Freshman Composition Classroom". The Internet TESL Journal, Vol. IV, No. 12, December 1998. http://iteslj.org/Articles/Gousseva-Literacy.htm
4. Pelaez, N. J. (2002). "Problem-Based Writing with Peer Review Improves Academic Performance in Physiology". Advances in Physiology Education, Vol. 26, No. 3, September, 2002. http://advan.physiology.org/cgi/content/full/26/3/174
5. Silva, E., Moriera, D. (2003) "WebCoM a tool to use peer review to improve student interaction". Journal on Educational Resources in Computing (JERIC), Vol 3, Issue 1, March 2003.
6. Wolfe, W. J. (2003). "Student Peer Reviews in an Upper-Division Mathematics Class". Exchanges The Online Journal of Teaching and Learning in the CSU http://www.exchangesjournal.org/classroom/1156_Wolfe.html