User Experience and Feedback on the RPI Homework Submission Server

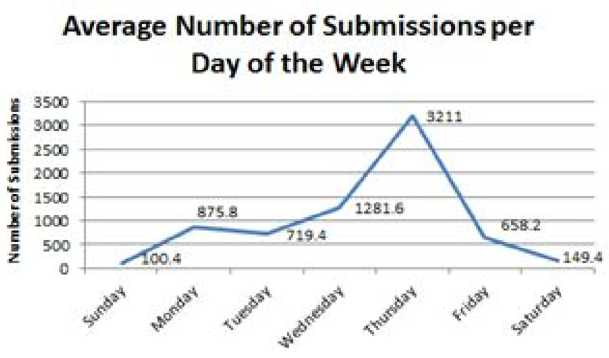
Andrea Wong, Rensselaer Polytechnic Institute, wonga6@rpi.edu
Eric Tran, Rensselaer Polytechnic Institute, trane@rpi.edu
Joe Jung, Rensselaer Polytechnic Institute, jungj5@rpi.edu
Ben Shaw, Rensselaer Polytechnic Institute, shawb3@rpi.edu
Marina Espinoza, Rensselaer Polytechnic Institute, espinm3@rpi.edu
Beverly Sihsobhon, Rensselaer Polytechnic Institute, sihsob@rpi.edu
Melissa Lindquist, Rensselaer Polytechnic Institute, lindqm@rpi.edu
Samuel Breese, Rensselaer Polytechnic Institute, breess@rpi.edu
Mattew Peveler, Rensselaer Polytechnic Institute, pevelm@rpi.edu, http://cs.rpi.edu/~pevelm
Barbara Cutler, Rensselaer Polytechnic Institute, cutler@cs.rpi.edu, http://www.cs.rpi.edu/~cutler/

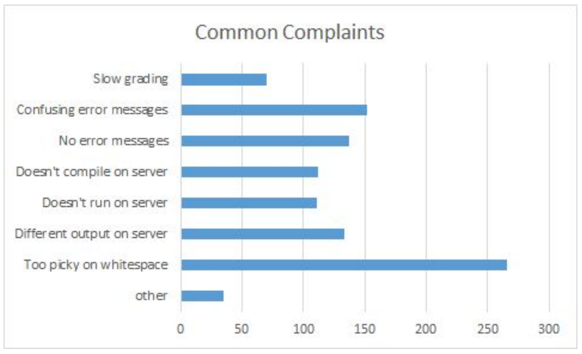
Abstract: The Rensselaer Polytechnic Institute (RPI) Homework Submission Server is an ongoing open source project that is used to collect, compile, and automatically grade the programming homeworks for students in our introductory and sophomore computer science classes. The server handles the viewing of homework, lab, test, and overall grades and late submissions and excused absences on homework. Our first hypothesis is that an electronic submission server is the preferred way for students to submit their coding homeworks because it provides immediate feedback about the correctness of their code and ensures fair and consistent grading since their code is compiled and run with the same test cases, on the same computer. Our second hypothesis is that students appreciate courses with a flexible policy for late submission of homeworks, allowing them to use a specific number of "late days" throughout the semester without penalty. The late day policy for each course can be configured by the course instructor and students can use these late days to better manage their time for difficult assignments and unexpected bugs in their code. We recently conducted a survey on user experience with the Homework Submission Server to test these hypotheses. We received 400 replies from approximately 850 students currently enrolled in courses using the server. This poster presents the results of our survey, including what students do or do not like about the server and specific feedback that we will incorporate as we continue development and expand the server to more courses at RPI and other universities.

Significance and Relevance of Topic: There are multiple ways to submit homework for a course: a hard copy of the assignment, uploading homework file(s) to a school or course management website - such as Blackboard Learn - or, for some schools, a homework submission server with automated testing and/or automated grading. When using a system that only collects electronic assignments, the TAs must then download submitted homeworks and compile and run the homeworks on their own computers. This offline system is slow and opaque; students only learn about compatibility problems when their homework is returned. In contrast, the RPI Homework Submission Server automatically tests and evaluates their code and gives immediate feedback. Students can study the output, correct errors, and resubmit before the deadline. In our survey, 91.4% of the students listed the ability to make multiple submissions and choose the submission to be graded as one of the most important advantages of the server. 90.7% of students also listed immediate feedback on auto-graded assignments as an important advantage.

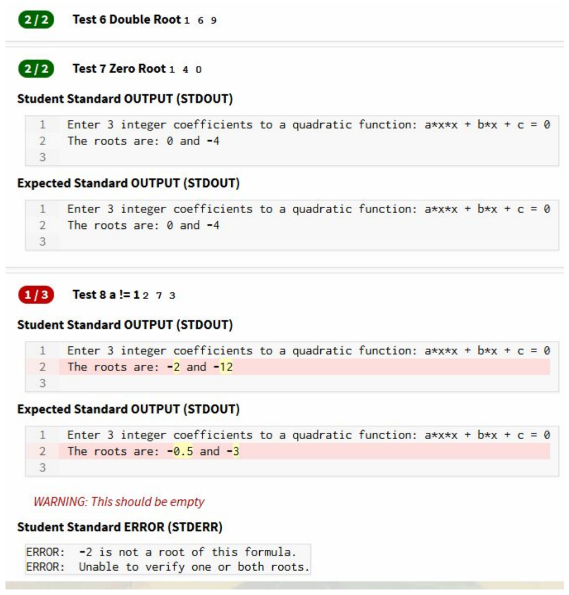
The server is an open source project, which allows other people and schools to set up their own version of the server, making the server more reliable because people outside the development team can test it and suggest fixes. Also, instructors from other schools can make changes and customize their own version of the server to suit their course.

Content: We present detailed responses from the survey, both as histograms of common answers and comments from students. For example, we asked students what they thought were the disadvantages of using the submission server. From personal experience as both students in the courses and as one-on-one programming mentors, we hypothesized that a major complaint would be “My code runs on my machine but not on the server!” (something the TAs hear frequently in office hours). However, to our surprise only 29% of the surveyed students said it was a disadvantage. Instead, 70.2% said that one of the largest disadvantages was how picky it was about whitespace when comparing the output of the student’s code and the expected code. In response, we will improve the awarding of partial credit being more tolerant of test case differences with spacing. Many of the students who filled out the survey also commented that compiler and run time error messages are confusing. We are looking into improving and customizing the error messages, specifically enumerating common coding mistakes and debugging suggestions − for example, how to solve a segmentation fault during run time.





Within the first six weeks of this fall semester, we received 30,818 total submissions over 3 courses from 850 students. We received 3,000 submissions in a single 24 hour period, and a max of 503 submissions in 1 hour (the hour before a big homework deadline). Despite the heavy load, the server was reliable: on average, students wait only 2.7 seconds before grading commences on a dedicated processor. We thought that students would complain about slow grading on nights with a large amount of submissions, but only 70 out of 400 responses from our survey noted that slow responsiveness (automated testing & grading) was an issue.



2 / 2 **Test 6 Double Root** 1 6 9

2 / 2 **Test 7 Zero Root** 1 4 0

Student Standard OUTPUT (STDOUT)

```
Enter 3 integer coefficients to a quadratic function: a*x*x + b*x + c = 0
The roots are: 0 and -4

```

Expected Standard OUTPUT (STDOUT)

```
Enter 3 integer coefficients to a quadratic function: a*x*x + b*x + c = 0
The roots are: 0 and -4

```

1 / 3 **Test 8 a != 1** 2 7 3

Student Standard OUTPUT (STDOUT)

```
Enter 3 integer coefficients to a quadratic function: a*x*x + b*x + c = 0
The roots are: -2 and -12

```

Expected Standard OUTPUT (STDOUT)

```
Enter 3 integer coefficients to a quadratic function: a*x*x + b*x + c = 0
The roots are: -0.5 and -3

```

WARNING: This should be empty

Student Standard ERROR (STDERR)

```
ERROR:  -2 is not a root of this formula.
ERROR:  Unable to verify one or both roots.
```

The server highlights in red and yellow differences between the output from the student’s code and the expected output and awards no, partial, and full points for each test.

The homework submission system is integrated with an easy-to-use interface for the instructor/TAs to award additional points (typically 40-60% of the overall assignment grade) for software quality and non-code portions of the assignment (proofs, order notation analysis etc.)

The overall system design also facilitates use of an external plagiarism detection tool[1] and our largest course is now using an external tool to digitize written exams for online grading[2].

[1] Measure Of Software Similarity (MOSS) http://theory.stanford.edu/~aiken/moss/
[2] https://gradescope.com/