4th International Conference on Higher Education Advances (HEAd'18)
Universitat Politècnica de València, València, 2018
DOI: http://dx.doi.org/10.4995/HEAd18.2018.8054

Graphical enhancements of the Virtual Programming Lab

Mooney, Aidan and Hegarty Kelly, Emlyn
Department of Computer Science, Maynooth University, Ireland.

Abstract

It is generally recognised that providing consistent, meaningful written feedback is not an easy task, especially when dealing with large classes. Feedback does needs to be effective, meaning it has to be appropriate and timely and needs to be individual, where possible. Automated feedback within Computer Science has been around since the 1960's with the main goals in relation to computer programming being to implement an automatic assessment tool to provide consistent feedback and to alleviate examiners' workloads. The Virtual Programming Lab, a plugin for the Virtual Learning Environment Moodle, is one such tool that allows for Automated Feedback on computer code.

This paper presents enhancements to the Virtual Programming that have been developed to make interacting with the tool more user-friendly and provide more graphical feedback to teachers. The enhancements developed provide in-depth graphical feedback on assessment grades within a class and also teacher focused graphical views which provide more in-depth analysis of assessment submissions. Discussed feedback shows that the enhancements developed were all positively received with feedback highlighting the benefits of each.

Keywords: *Automated Assessment, Virtual Programming Lab, Timely Feedback.*

1. Introduction

Computer Science is often a new discipline for students entering third level education. In Ireland, for example, Computer Science will only become an examinable secondary level state subject from 2020. Computer Science has a notoriously low progression rate at the end of first year into second year, with programming modules seen as a major stumbling block (Quille et al., 2015). Programming tends to be an individual task at third level and one which can be very frustrating with struggling students feeling isolated and often embarrassed to ask questions. For novice programmer's multiple errors in code can be demoralising and frustrating with feedback from traditional compilers tending to be high level and confusing. Computer Science classes tend to be large and with this comes the problem of lecturers and demonstrators having to spend an increasing amount of time grading students work.

This paper looks at the Virtual Programming Lab (VPL) and enhancements that have been added to it. VPL is an automated feedback system which allows for tailored feedback that gives an automatic grade even if a program does not compile. VPL is a Moodle plug-in allowing for the integration of the automatic grading feature with a popular Virtual Learning Environment. In addition to the tailored feedback already featured in the VPL this paper presents enhancements that will allow teachers to get details of overall class performance in a quick easy to read format.

2. Literature Review

Within education there are many ways to provide students with feedback on their work, be it in the form of summative or formative assessment. This could range from verbal feedback to a class or an individual, to written or commented feedback. However, no matter what kind of feedback we provide it is generally recognised that providing consistent, meaningful written feedback is not an easy task, especially when dealing with large classes (Biggam, 2010). Feedback needs to be effective, i.e., it has to be appropriate and timely (Poulos & Mahony, 2008). A generic form of feedback is through assessment which is specifically intended to provide feedback on the students' performance with the goal of improving and accelerating their learning (Sadler, 1998).

According to Brookhart (2008) feedback needs to come while students are still mindful of the assessment they have completed. It needs to be presented to the student while they still think of the learning goal as a learning goal, not something they already did. Automated assessment within computer programming has been around since the 1960's. The main goals of automated assessment with programming are to implement an automatic assessment tool to provide consistent feedback and to alleviate examiners' workloads. The basic requirement for automated assessment of programming tasks centre around the comparison between a submitted student program and a supplied model solution by the teacher (Gupta & Dubey, 2012). Additionally, some automated systems attempt to test internal data representation of programs to assess their quality (Saikkonen & Korhonen, 2001).

One automated assessment solution for analysing computer programs is Junit (Tahchiev et al., 2011; MoodleDocs, 2018). Using Junit lecturers can provide a code base for students and then a test class to test the submitted work. Another notable software assessment solution is HackerRank (2018) which is a software system hosted online that provides casual and competitive programming questions for individuals and businesses. For lecturers, they can create, manage and grade programming assignments. However, HackerRank is more suited to non-educational purposes as it is not tied in with an Virtual Learning Environment and any assessment scores need to be manually imported to Moodle (or equivalent) at a later stage.

The Virtual Programming Lab (VPL) was developed at the University of Las Palmes de Gran Canarias in Spain (Rodriguez-del-Pino et al., 2012). Using VPL lecturers can manage and receive data from VPL's reporting function on those assignments, such as, the amount of times a program was run for submission, the amount of time that was spent on the assignment and a record of all individual attempted submissions. VPL is a Moodle plug-in allowing for the easy integration of grades in to Moodle from assignments.

3. Methodology

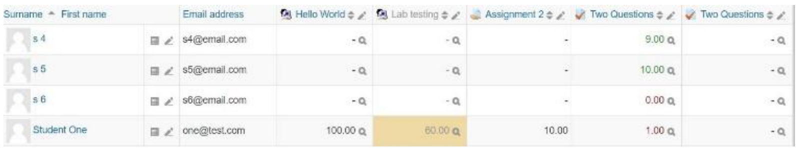
VPL was piloted in the Computer Science department at Maynooth University during the second semester of the academic year 2015-2016. The feedback was very positive from the teaching staff and students. It was decided to continue using VPL and incorporate in to the lab sessions as the core method of assessment of programming assignments. Lecturers observed the many benefits from VPL such as being able to write scripts to check for errors in student's assignments upon submission and automatically award a grade.

Since VPL was first trialled a number of projects have been undertaken to enhance VPL and make it an even more useful system for lecturers in terms of feedback. This paper will look in detail at some of these projects and highlight the enhancements delivered.

4. Enhancements

4.2. Enhancement 1

The first enhancement was the addition of more detailed graphical feedback on grades. In the standard Moodle build, grading feedback is only given as a table of scores as shown in Figure 1, which can make it difficult to quickly survey a large set of results.



Surname ▲ First name		Email address	Hello World	Lab testing	Assignment 2	Two Questions	Two Questions
s 4		s4@email.com	-	-	-	9.00	-
s 5		s5@email.com	-	-	-	10.00	-
s 6		s6@email.com	-	-	-	0.00	-
Student One		one@test.com	100.00	60.00	10.00	1.00	-

Figure 1: Sample Moodle Grade Table.

It was decided that the graphs would be added to the grading pages by editing the source code of the PHP grading pages. Implementing the graphing capability involved a number of steps including adding JavaScript graphing scripts, creating PHP files needed for graphing and connecting to the database and adding styling features to the graphing capabilities. A sample created graph and user interface is shown in Figure 2. A bar chart was chosen as you can quickly compare information, revealing highs and lows at a glance.[1]

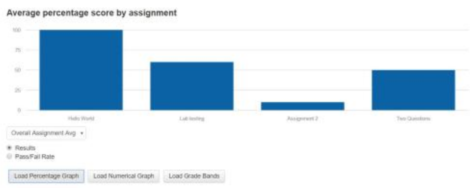


Figure 2: Sample graph showing average percentage scores by assignment.

To ensure that these graphs are only available to teaching staff additional PHP checks were performed that loaded the graphs and graphing elements just for teaching staff. These graphs allow for a birdseye view of the assignments allowing staff to focus on any badly answered.

4.3. Enhancement 2

The next enhancement developed were teacher graphical views. The goal of this enhancement was to create a visualisation engine for VPL. Specifically using web development libraries and tools, a front-end framework would be developed that provides an instant overview of high-level metrics on student performance. This enhancement attempted to re-represent data in a more creative, comprehensive and user-friendly manner.

[1] http://theathenaforum.org/sites/default/files/WHich%20chart%20is%20right%20for%20you.pdf

With this re-representation of data, it will provide lecturers with a new level of analysis that can be carried out on their students' performance. Figure 3(a) shows the view with the current VPL submission page while Figure 3(b) shows the created user interface for the submission page. It can be observed that the standard VPL submission list provides only a list of the data. The new adaption (Figure 3(b)) displays data on all 226 students about a single VPL activity (in this case, a particular lab question) in a much more convenient manner.

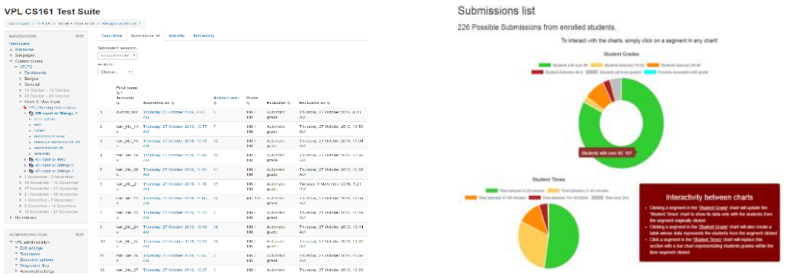

VPL CS161 Test Suite
NAVIGATION
ADMINISTRATION
Submissions list
226 Possible Submissions from enrolled students.
To interact with the charts, simply click on a segment in any chart!
Student Grades
Student Times
Interactivity between charts

Figure 3 (a): VPL Submission Page, and, (b): Created User Interface of the Submission Page.

The 'Student Grades' graph is a doughnut chart that has six grade categories for student grades for that particular lab question. A teacher can hover over a segment to find out the exact number of students within that segment. The 'Student Times' graph is a pie chart displaying similar data but based on the time taken by students on the lab question. The 'Interactivity between charts' list briefly explains how the interaction between different charts works, and what outcomes will be seen from this interactivity. Doughnut charts and pie charts were chosen here as they show relative proportions of information

Clicking a segment in the 'Student Grades' causes two things to happen. Firstly, the 'Student Times' will be updated according to the segment clicked. For example, if the 'students with over 80%' was clicked then the 'Student Times' would update and display the times taken only by those 187 students who got over 80%. Secondly, a table will be dynamically generated to contain students only from the segment clicked. Clicking a segment in the 'Student Times' causes the 'Interactivity between charts' list to disappear, and a bar chart displaying grades of the students within the time segment specified will appear - Figure 4. A bar chart was chosen here as they are especially effective when there is data that splits nicely into different categories allowing you can to quickly see trends in the data.[1]

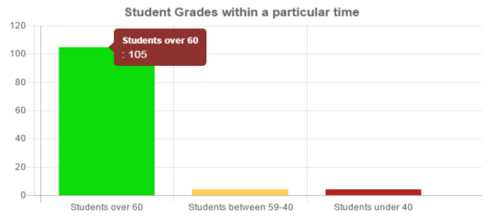


Figure 4: Bar chart for student grades generated by clicking within the Student Times chart.

Additionally, this application allows for users to retrieve data from one or more VPL activitiy; in this case, lab questions. Every lab question has a unique ID associated with it assigned by Moodle. Once the user has inputted the required IDs, the specified data will be retrieved and presented. Firstly, two pie charts are shown upon retrieval (see Figure 5). These two charts present the average grade and time of students over the lab question IDs specified. Clicking on a segment in the charts will dynamically generate a html table. Again pie charts were chosen here to show the relative proportions of the data.[1]

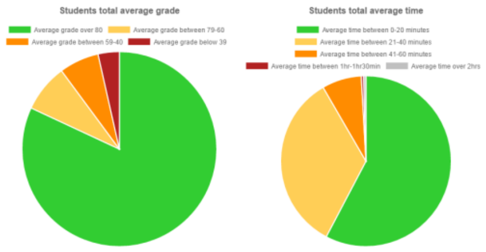


Figure 5: Pie charts showing students average grade and time.

In addition, a modified box plot for student times is generated. This modified box plot, shown in Figure 6, provides the user with the information that a traditional box plot would, like the max/min values, the median and upper/lower quartiles but supplemented with an additional two pieces of information. Firstly, there is a scatter plot-like graph that sits beside the box plot that shows the exact spread of student times. Secondly, an untraditional presentation of the Gaussian model is represented by the dotted lines forming a diamond like shape. It shows the mean and standard deviation of the data. When the box plot is hovered over, it provides exact details of the data presented. Box plots were chosen here as they are suitable for comparing range and distribution for groups of numerical data.[1]

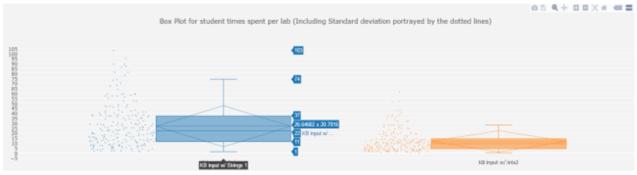


Figure 6: Box plot of the student times in the retrieval page.

5. Feedback / Assessment

5.1. Enhancement 1

The ability to group items in VPL within a particular section was solved using the Flexible Sections Format plugin. This allowed for the nesting of section topics and the ability to freely move VPL activities between topics. Labs and activities would be hidden when placed inside a hidden topic and shown otherwise. Unit testing was carried out to confirm that all implemented changes worked and the outcomes were as expected as the enhancement was being developed. Each graph that was developed was compared against the results in the database to ensure no data has been misrepresented.

Once the enhancement was completed a group of testers tested the system and provided results to a survey. These testers were lecturers and other teaching staff and they were provided with dummy data from a submitted assignment in a VPL activity to test the system. Due to the rigorous nature that the enhancement was created under the results of the survey were very positive. In relation to the question relating to the ease of use of the graphs, 70% of the respondants said they were extremely easy to use with 30% saying they were very easy to use. When asked if the displayed information was useful for lecturers and course managers, 50% replied that it was extremely useful with the other 50% replying they were very useful. In relation to how useful they felt the changes were overall to VPL, 50% of the respondants said that they were extremely useful while the remaining 50% said they were very useful.

5.2. Enhancement 2

To measure how effective the visualisation engine was as a solution to the problems identified in the current version of VPL, a questionnaire was distributed to users while they interacted with the submission and retrieval page. The testers of these enhancements were lecturers, students and technical staff. We now provide feedback to a number of the questions asked.

When asked iof they were able to tell what data each of the charts represented, 87.5% of them could identify & understand what each chart represented as seen in Figure 7(a). In relation to whether the new page provided an overall better “experience & representation of data” compared to the same page in the original VPL 100% of the respondents agreed that it did as seen in Figure 7(b).

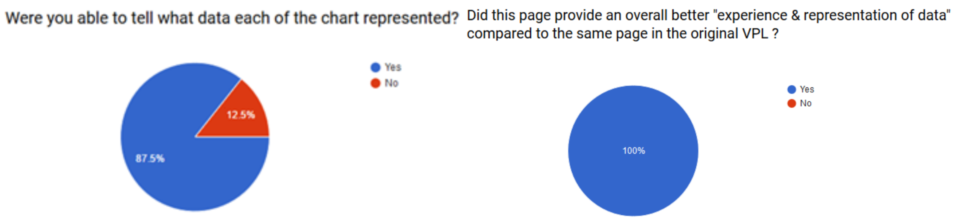


Figure 7: Pie chart results for the questions: (a): "Were you able to tell what data each of the charts represented?", and, (b): "Did this page provide an overall better "experience & representation of data" compared to the same page in the original VPL?

The testers were then asked about what they liked best about the new system. Some of the responses were:

"The use of graphs to visualise the data is a much more efficient approach than the previous project. The system has a nice flow to it and has a simple yet attractive interface."

"Graphing facility on page one."

"The graphical visualisation of the data."

"Informative graphs that you can clock on to obtain individual and group specific data."

"I like all the colours, updated in real time, offered pointers."

It can be seen from these responses that there was significant mention of the visualisation engines graph capabilities. This is quite an important result as it was felt that the question proved that the main aim in implementing this enhancement was achieved.

6. Conclusion and Future Work

Timely feedback on assessments is vital for successful learning and no matter what kind of feedback is provided it is generally recognised that providing consistent, meaningful written feedback is not an easy task, especially when dealing with large classes. Automated feedback, if targeted and individualised has been shown to be very effective. The Virtual Programming Lab allows for just such feedback. This paper looks at a number of enhancements to the Virtual Programming Lab to allow for more detailed graphical feedback on grades and teacher graphical views providing more detailed information regarding some assessment with in depth analysis provided. All of these enhancements were tested and all feedback for extremely positive highlighting the benefits of including such enhancements into the Virtual Programming Lab.

Further enhancements being developed include incorporated JFLAP in to the Virtual Programming Lab, developing a system to allow for easier creation of bash scripts for VPL and the development of a GUI to allow for customisation of the VPL system to allow all users get a positive experience using it. All of these enhancements will further improve VPL and make it easier for people to integrate it into their courses.

Acknowledgments

We would like to acknowledge the work of Alex Black and Ademola Akingbade in this project.

References

Biggam, J. (2010). Using automated assessment feedback to enhance the quality of student learning in universities: A case study. T*echnology Enhanced Learning. Quality of Teaching and Educational reform.* Springer.

Brookhart, S. M. (2008). *How to give Effective Feedback to your students*. Alexandria, Virginia, USA: Association for Supervision and Curriculum Development

Gupta, S. & Dubey, S. K. (2012). Automatic assessment of programming assignment, *Computer Science & Engineering*, vol. 2, no. 1, pp. 67.

HackerRank (2018), https://www.hackerrank.com/. (Accessed: 8th February 2018).

MoodleDocs (2018), Junit Question Type. https://docs.moodle.org/20/en/Junit_question_type. (Accessed: 8th February 2018).

Poulos, A., & Mahony, M. J. (2008). Effectiveness of feedback: the students perspective. *Assessment & Evaluation in Higher Education*, vol. 33, no. 2, pp. 143-154.

Quille, K., Bergin, S. & Mooney, A. (2015). PreSS#, A Web-Based Educational System to Predict Programming Performance. *International Journal of Computer Science and Software Engineering (IJCSSE),* 4 (7). pp. 178-189. ISSN 2409-4285.

Rodríguez-del-Pino, J. C., Rubio-Royo, E., Zenón, & Hernández-Figueroa, J. (2012). A Virtual Programming Lab for Moodle with automatic assessment and anti-plagiarism features. *Proceedings of WorldComp12*, Las Vegas, USA.

Sadler, D. R. (1998). Formative assessment: revisiting the territory. *Assessment in Education: Principles Policy & Practice*, vol. 5, no. 1, pp. 77-84.

Saikkonen, R., Malmi, L. & Korhonen, A. (2001). Fully automatic assessment of programming exercises. *ACM Sigcse Bulletin,* ACM, vol. 33, no. 3, pp. 133-136.

Tahchiev, P., Massol, V., Leme, F. and Gregory, G. (2011). JUnit in action. 2nd ed. Greenwich: Manning, pp.3-5.