Mass Production of Individual Feedback

David Heaney
DCU
Glasnevin
Dublin
+353 1 7008449
david.heaney@computinsg.dcu.ie

Charlie Daly
DCU
Glasnevin
Dublin
+353 1 7005572
cdaly@computing.dcu.ie

ABSTRACT
We describe a system to improve the quality of feedback provided to an Introductory Programming course. The system uses web technology to create a graphical tool that the tutors can use to produce student-friendly feedback. It was used during the first semester in Autumn 2002. Analysis of exam results show that there was a statistically significant improvement and student surveys showed that they enjoyed using the system.

Categories and Subject Descriptors
Classroom Management, Web-based Techniques (online system to manage all students assignments and feedback)

General Terms
Management, Design, Experimentation, Human Factors.

Keywords
Education, Introductory Programming, web-based feedback.

1. INTRODUCTION
Computer Science courses tend to attract a diverse range of students. It is imperative that each student receives motivation and encouragement that is right for him or her. This is often difficult to provide with large classes. We describe a system which uses 2nd year students to tutor the 1st year students with the aid of a web based feedback system. In the Fall Semester 2002 the system comfortably handled a class of 200 plus students, with each student receiving complete and comprehensive personal feedback on their assignments each week. The data shows that there was a statistically significant improvement in the weaker students' grades, through use of this system.

We first address the problem of feedback and we examine possible solutions. Then we outline our solution, describe the implementation and analyse the performance, and hopefully quantify and qualify the improvement for the students.

ITICSE'04, June 28–30, 2004, Leeds, United Kingdom.

2. LARGE CLASSES, DIVERSE STUDENTS AND POOR FEEDBACK
Computer programming is difficult to teach and learn. Studies in Ireland [1] have shown that Computer Science has the highest failure rate, giving a 26.9% non-completion rate for Computing courses. In addition, international studies [2] show that many students do not know how to program to an acceptable level.

Jenkins [3] points to student diversity, poor motivation and low student expectation in an attempt to understand why programming is so difficult to teach. It is not uncommon for a class taking their first course on programming to have vastly different prior experiences and pre-existing skills. They have different expectations, motivations and hopes. This diversity is difficult to handle in large class sizes, which are not unusual of toady's courses.

Keller points out that both motivation and expectation problems can be addressed using appropriate feedback. In his ARCS Model of Instruction [4], Keller identifies feedback as a means to develop a student's confidence (expectancy) and satisfaction (motivation). Keller's Motivational Delivery Checklist says that a course which "provides feedback on performance promptly" and which "makes statements giving recognition and credit to learners as appropriate" will increase the value of the course. Phil Race [6] argues that "the greater amount of feedback that learners receive before the end of course assessment the greater their opportunity to learn from such feedback". In summary students need plenty of quality feedback and ideally this feedback should occur quickly.

In previous work [5], we had put a tutor system in place where the 2nd year students supervised and tutored the 1st year students' lab sessions. This was successful. But we found that the students just did not receive adequate or timely feedback on their assignments from their busy lecturers. We decided to extend the successful tutor system and enable the lab tutors to provide feedback.

Poor motivation is not a new problem and efforts have been made to solve it. Pair programming has been employed in various colleges [7], [8], [9] and [10] in an effort to keep students interested, and to manage large classes. Most of these studies report a reduction in the demand for help from the tutors and/or lecturers. They fail to conclusively show that pair programming improves learning, but our hunch is that it does when used correctly, and we have coupled the pair programming practice with a personal feedback system.

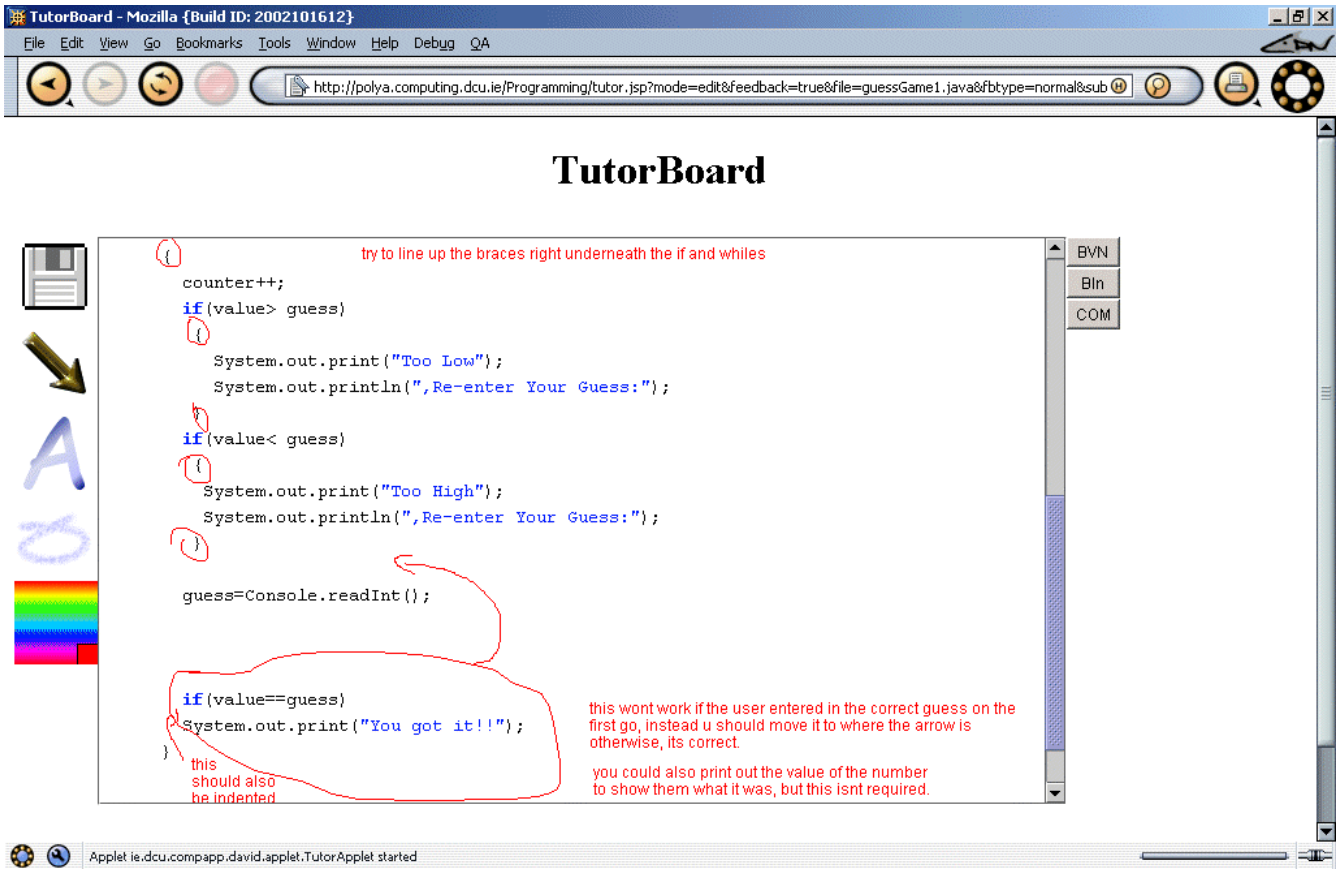

Figure 1. Screen shot of the feedback system in action.

2. MASS PRODUCTION OF INDIVIDUAL FEEDBACK

Mass Production of Individual Feedback is simply a phrase applied to our method of providing personal, quality, quick feedback to each student on a weekly basis through use of our 2nd year tutors and our online feedback system.

See figure 1 for a screenshot of the system.

2.1 The Tutors

Using competent 2nd year students chosen from the top of the previous year's class, we hoped to maintain a high level of feedback for the students. They were given a training course to help them provide clear and relevant comments.

Second year tutors provide numerous benefits. Firstly they have mastered the material to a high level. Secondly, they are aware of the difficulties introductory students have since they have recently encountered these difficulties themselves. Thirdly, there are many second years, they provide a perfectly scaleable solution to manage large class sizes (if you take the top 10% of second years you can always maintain a 1:10 ratio of tutors:students). Finally they act as a role model for the students. This helps the students' confidence, since if the tutors can have such knowledge of the material after just one year then the student believes that they can too. It also can help their motivation since students may also aspire to tutor next year, for monetary reasons or personal satisfaction. It all helps to motivate the students to become better programmers.

2.2 The System

2.2.1 Overview

The system allows the lecturer assign tutors to students and to set assignment details, including deadlines, specification files etc. The lecturer can also monitor the feedback and grades that the tutors are giving.

The system provides the tutors with the best tools for marking computer programming assignments including a whiteboard [11], with custom made tools, and predefined comments to catch common programming errors and keyword highlighting for clarity. Tutors can mark the assignments as soon as they are submitted, and the feedback is available to review by the students as soon as they are marked. Many places have online assignment submission but this deals with online marking.

All assignments can be submitted and all feedback can be reviewed wherever and whenever is convenient for the student via a standard we browser.

It is important that the tutors can provide the right type of feedback through the system. In programming things are not always right or wrong. A feature of many courseware tools like Blackboard [12] and WebCT [13] is exactly this sort of automated

feedback. Automatic feedback is very useful to a point and it is very fast. On the other hand, our system puts the emphasis on personalised student-friendly feedback that affords the tutor the opportunity of providing encouragement at a more personal level, thus increasing the student's motivation and satisfaction.

With the system there is less emphasis on whether the code works or not (a feature of automated marking) and more emphasis on good programming practice (such as, when to use a for loop, as opposed to a while loop, or drawing attention to good or bad variable and method names etc.)

2.2.2 Technical Details

There are two main parts to the system. The server end keeps a database of lecturers, tutors, students, their project files and the associated feedback. The client end is a drawing tool that allows a tutor to provide feedback on a Java assignment (more details below). Both parts are written in Java. The server is written using servlet filters (for authentication) and JSP pages to present information to the users. The client end is a Java applet. This is an adaptation of an open source whiteboard tool and an open source Java code formatter.

Managing the users was relatively easy as DCU has an LDAP server that stores usernames, passwords and details of all university computer users. We were able to access the LDAP server using a servlet filter that controlled access to the system. Thus the need to manage new students, lost passwords etc. did not arise.

3. IMPLEMENTATION

The system had its trial run in Semester 1 of 2002 in the Introductory Programming module run here in DCU for first year students. There was a range of students from different courses. Although the majority were Computer Application students (138) there were also Mathematical Science (19), Financial Mathematics (36) and Computer Linguist (16) students. In total there were 209 students registered on the system. We also picked 23 second years to act as tutors for the lab session. The lab sessions were two 2-hour labs twice a week. The students received an assessed assignment nearly every week, 8 in total.

During the second last Thursday of the semester a questionnaire was handed out to the tutors. Every tutor returned the questionnaire. The final week a student questionnaire was handed out to the students who used the system and were present in the lab that week. We got 128 responses back, which represents 61% of the students registered on the system.

4. RESULTS

In analysing the impact of the system we considered the usage of the system, the students opinion from the survey, the tutors opinion and how the system enabled them to mark a substantial number of students assignments quickly and easily. Finally we considered the end of exam results and compared them to the previous year where no such feedback system was employed.

4.1 Usage

We kept extensive logs of the tutors and students usage of the system. We were pleased to see that the students appreciated the easy access to the system. Log on sessions in the evenings and during weekends proved that students were getting quicker access to their feedback than they would be through traditional assignment submission processes. In figure 2 you can see the number of logins by day. Tuesdays and Thursday were the busiest days as those were the days the labs were scheduled. The graph does show that the system remained busy throughout the week and to a lesser extent through the weekend.

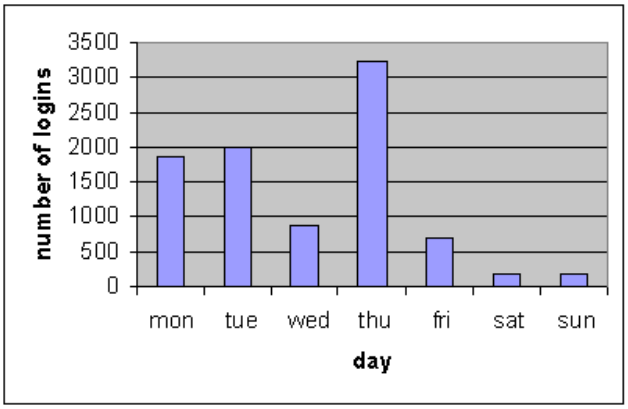

Figure 2. Total login sessions across the semester by day

4.2 Student Reaction

Although we were using inexperienced 2nd year tutors, it appeared that they were capable of providing helpful feedback to the students. In the survey we had a positive response from the students about the tutors' feedback. Indeed 88% of students agreed or strongly agreed that "the tutors were able to explain the material well" and 97% of the students surveyed agreed or strongly agreed that "the tutors knew the material very well".

On the most part the students agreed that we had met our target of making the feedback clear (53 out of the 128 responses strongly agreed) relevant (36 strongly agreed) and timely (40 strongly agreed). The students were even more enthusiastic about the convenience of being able to log in from anywhere. Here 75 indicated that they strongly agreed that it was useful to be able to submit code via a standard web browser.

They were also asked opinion questions on their experience with the system. For the most part they were positive and helpful comments on the system. Here are a few of the responses we got when we asked for "the best feature about the feedback system":

"It gave us a good idea about where we're going wrong in our programming"

"It is a clear and precise way of seeing errors."

"it made program very easy to read (with colours etc)"

"I could submit the code from anywhere at any time."

"Easy to use - I was able to check my tutors feedback anytime anywhere."

"easy to use; easily accessed; quick correction of work"

"easy to use, a lot handier than having to go find your tutor"

"It was good to be able to see the advice directly linked to certain parts of the code."

"Getting a pat on the back. We all like the praise we can get, ya know."

"It was easy to use and feedback was returned quickly"

"You could view your programs as you wrote it, with comments pointing to where you could have improved the program."

"You could ask experts about problems ex people in the class who had a higher skill of programming questions, and the answers were good."

They seemed to like the simplicity and that it was easy to use. They liked the convenience of being able to submit from anywhere and check their feedback too. They liked the encouragement they got. It was not all good though. We asked for "the worst thing about the feedback system" too. Here is what they said:

"undecided, maybe could have had audio, sound"

"We often didnt get feedback"

"it takes a while to get the feedback"

"not enough info if something was incorrect"

"Nada, seems pretty cool to me."

There had been some bugs and problems with the system that surfaced slowly as the semester went on. So there were some students who pointed this out to us here. Also some tutors were not as good at giving feedback to their students. Some students rarely got any feedback at all from the system unfortunately. But the negative comments were far fewer than the positive ones.

We also asked for ways they thought might improve the feedback applet. They said:

"There was maybe correct copies of code to actually compare yourself."

"Instant Messaging"

"scores were given on your assignment"

"Tutors themselves submitted a program to each student showing the best and most effective way to do each assignment after you had submitted your program with all the extra credits shown so that students can learn from past mistakes."

"People used it more and were more willing to give feedback"

"it worked more often"

4.3 Tutor Reaction

Using the system we were able to manage a class of over 200 students with the help of the 2nd year tutors. Each student submitted an assignment weekly and had received feedback by the end of the week. The simple well-designed interface meant that it was easy for the tutors to manage the their group of students.

From the questionnaire results, the tutors commented repeatedly on this:

"was able to mark assignments very quickly"

"it was easy to write over the code and show exactly where errors occurred"

The system handled 992 file submission (some assignments required more than 1 source file and most students did the assignments in pairs), and a total of 845 assignments marked by the tutors.

4.4 Exam Results

To analyse the effectiveness of the system, we compared two groups of students, one group had used the system and the other group had not. We omitted students who were not in the Computer Applications course and for whom we had no Leaving Certificate data. The Leaving Certificate is the final exam Irish students sit before they begin third level education. The amount of points they get from the Leaving Certificate solely determines which college courses they can take.

In our analysis of the exam results we focused our attention on the students who got low points in their Leaving Certificate. The Higher Education Authority [1] identifies these students as most likely to fail to complete their course of study. A comparison between these at risk students' results in Fall 2002 with their counterparts of the previous year shows a statistically significant improvement.

Figure 3 shows the average marks for the students in the various points range for 2001 (with no feedback system in place) and 2002 (using the system described in this paper). The numbers above the bars represent the number of students in each category. Notice the improvement for students with entry points of 350 to 370 points. In 2001 the mean was 38%, while in 2002 it was 48%.

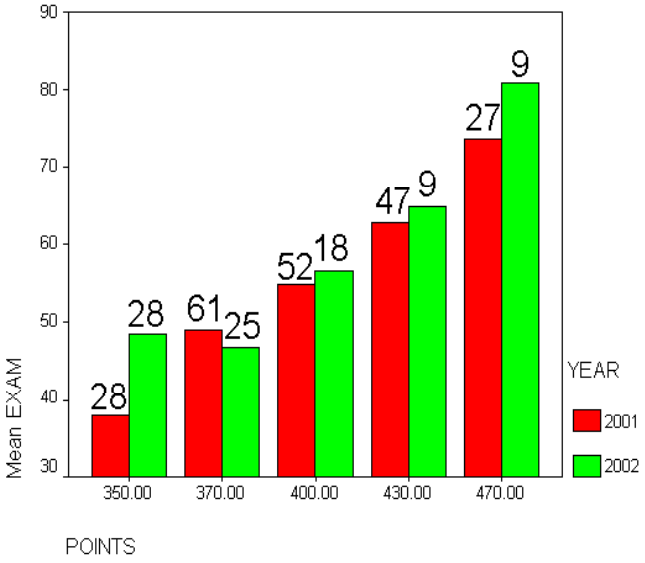

Figure 3. Comparison of exam results for 2002 and 2001 grouped into groups of equal points

In particular for the low points students (350 to 370), the t-test for equality of means produces a test statistic of t=–2.123. You would expect to see a t value like this by random chance only 4 times in 100. So, there is a statistically significant improvement in the at-risk low points group.

5. CONCLUSION

Feedback is an integral part of the learning process. Through our personalised, fast, quality feedback we succeeded in increasing student motivation and confidence. The system was very useful in managing large volumes of student assignment submissions. The exam results show that weaker students benefited most from

the system. This was perhaps because weaker students find it discouraging when they struggle over problems that other students find trivially easy. Our web-based feedback system was able to provide individual attention in order to encourage the weaker students to persevere with the task of learning how to program.

The students' comments on the system were very pleasing. Their comments are in agreement with what Keller [4] theorises. Feedback increases students' satisfaction and confidence. From the questionnaires the most dissatisfied students were the ones who were dissatisfied because their tutor had not given them enough feedback. This proves that the students themselves realise that it is important to get feedback.

6. FUTURE WORK

The system is in use again for the Fall Semester 2003. We have added a facility to allow the tutors to run the assignments online to aid them in the marking, plus some other significant improvements to the system.

We added self and peer assessment. Phil Race [6] points to this as a very useful method for improving learning.

Self-assessment aids the learning process and encourages students to reflect on their own assignments.

While with peer-assessment the students benefit from correcting other students assignments, they see other examples of how to do the programming assignment and it gets them thinking about what makes a good program. In the survey many of the students did request that they could see examples of the code. As an added bonus the system creates more feedback at no extra expense to the college.

We are looking forward to seeing the students' reaction to the new improved system, and analysing any effects the peer and self assessment had on the students learning.

7. REFERENCES

[1] Mark Morgan, Rita Flanagan and Thomas Kellaghan, "Higher Education Report A Study of Non-Completion in Undergraduate University Courses" published February 2001.

[2] McCracken, M et al. (2001), "An international multi-institutional study of introductory programming courses" Report by the ITiCSE 2001 Working Group on Assessment of Programming Skills of First-year CS students.

[3] Tony Jenkins, John Davy, "Diversity and Motivation in Introductory Programming".

[4] Keller, J.M. & Keller, B.H. (1989). Motivational delivery checklist. Florida State University.

[5] Reference omitted for the sake of anonymity

[6] Phil Race, "Never Mind the Teaching Feel the Learning". SEDA Paper 80, 1993.

[7] Charlie McDowell , Linda Werner , Heather Bullock , Julian Fernald, "The effects of pair-programming on performance in an introductory programming course" ACM SIGCSE Bulletin , Proceedings of the 33rd SIGCSE February 2002.

[8] Laurie A. Williams, Robert R Kessler. "Experimenting with Industries Pair-Programming Model in the Computer Science Classroom"

[9] Jerzy Nawrocki, Adam Wojciechowski. "Experimental Evaluation of Pair Programming.

[10] Nachiappan Nagappan, Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, Suzanne Balik. "Improving the CS1 Experience with Pair Programming"

[11] Merlin Hughes, "Draw the world: Create networked whiteboards with Java 1.1" http://www.javaworld.com/javaworld/jw-11-1997/jw-11-step.html

[12] Blackboard Learning System, http://www.blackboard.com/

[13] WebCT, http://www.webct.com