Improving the Technical Infrastructure of a Programming Course

Uolevi Nikula, Satu Alaoutinen, Jussi Kasurinen, Toni Pirinen
Department of Information Technology, Lappeenranta University of Technology, Finland

Abstract

Decreasing interest in computer science and programming has caused trouble to many institutions recently. To tackle this issue an active improvement of the first programming course was started four years ago. In this paper the adopted action research approach is described together with the improved results from the course. The improvement actions have so far focused on the technical infrastructure of the course including the used programming language, development environment, weekly assignments, course project, and teaching materials. Even though the technical infrastructure still has deficiencies, the limit of improvement possible with efforts focusing on the technical aspects is approaching, and continued improvement requires moving the improvement focus on the pedagogical aspects.

1. Introduction

The number of students selecting computer science has recently reduced in many institutions [1] along with a decreasing interest in programming [2]. We tackled this problem by doing a major revision for our first programming course and we have thereafter continued the improvement actions for four years now. In this paper we describe the adopted action research approach [3], the data that was used to direct the improvement actions, and the observed changes in the course results.

The improvement work started from the problems of decreasing student count and deficient programming skills after the course. The improvement actions were implemented as process improvement actions following the IDEAL improvement model [4] focusing on improvement of the technical infrastructure used in the course covering used programming language, development environment, weekly assignments, course project, and teaching materials.

The rest of the paper is structured as follows. Section 2 describes the action research approach used, and Section 3 summarizes the actions taken and results attained. Section 4 provides the discussion and conclusion of the paper.

2. Action Research

Conducted improvement work followed the action research approach where the authors participated actively in implementing changes. Action research is based on four premises [3]: purpose is established before action, practical action is taken in the problem setting, the action informs the theory, and the reasoning and action are socially situated. Since the course was implemented annually, the goals and actions were identified for each course based on the theory created as part of the data analysis of the previous course. The collected data included both quantitative data (e.g., number of participants and assignments completed, grades, and survey results) and qualitative data from open questions in the surveys as well as other feedback from the students during the course. The qualitative data analysis was used to understand the problems students experienced while the quantitative data was used to assess their overall significance. Thus, strictly speaking, the reasoning, action formulation, and action taking were not collaborative [3] even if the student feedback played an important role in the analysis and action planning. However, bearing in mind the annual cycles in education, we found this approach to suit well for our situation.

3. Course Improvement

Fundamentals of Programming is the first programming course in our university and it is taken annually by circa 170 students. About half of the students major in Computer Science, Electrical Engineering and other engineering disciplines while the other half covers students who have Computer Science as a minor. The course is scheduled for the first fall term, but it is not enforced so some leave the course for a later time. The course workload is 5 credit units (ects), or about 135 working hours, and the course is kept on the 14 week fall term.
The course was given with the C-programming language in 2001 through 2005. As the latest lecturer change took place in 2005 the course materials were analyzed and the course project was revised to make it more practical. The biggest practical issue with the course was that students returned course projects throughout the year and reported that “I have returned all the other assignments but this one two years ago – can you check it now?” Thus, to get the course under control, a strict policy of not accepting partially completed course assignments from previous years was enforced in the fall of 2005. The course result analysis revealed that the proportion of students passing the course fell sharply from the previous year (Figure 1), but the analysis failed to pinpoint any individual problem with the course. It was only evident that the students found the course difficult. The personal experiences of using a Windows workstation to connect to a C-compiler and editor running on a UNIX server with the Exceed connectivity software supported the feeling of unnecessarily complex course implementation.

For the fall 2006 the technical infrastructure of the course was totally revised. The programming language and tools were changed to Python [5], weekly programming assignments and quizzes were introduced, and a Finnish language programming guide was developed [6]. Consequently, the proportion of students completing the course increased from 36% to 53%, and the student feedback turned positive. Since many students still failed the course, the analysis continued and in the final survey revealed that many found the project difficult and further complained its difficulty in the open questions. Since the project work had been developed the previous year for the C-language course, it followed the UNIX filter idea which was apparently not familiar to the students.

In 2007 a new project framework was introduced that required students to modify the functionality of an existing application rather than developing code from scratch [7]. The proportion of students passing the course increased, and the perceived difficulty of the project reduced. The difficulty was questioned with a scale from 1 (difficult) to 4 (easy) and the average rose from 1.84 to 2.24 while the difficulty of all the course elements rose only from 2.45 to 2.51. The majority of the student feedback was comparable to the following comment: “The project turned out to be a really positive experience; it was much nicer to code something when you could see the result right away.” An analysis of the course grades revealed that the proportion of the lowest grades had increased (Figure 2), which was interpreted as a sign of less motivated students being able to pass the course. This line of thinking was also supported by student feedback: “When I started doing the assignments and spend time doing them, I realized that I started to understand what was going on there.” On the negative side the amount of students claiming the quizzes unclear increased, and the grade distribution suggested that getting a good grade from the course was difficult.

In 2008 three changes were done. First, the students were asked to report problems with the quizzes and the first reporter for each problem got an extra point for pinpointing it. Second, voluntary assignments were introduced, and students completing them got extra points for the grade. And third, the lectures were video recorded and made available after the lectures. Consequently the dissatisfaction with the quizzes reduced even though a study of the quiz points suggested that some questions were still unclear. The course grade average increased from 3.0 to 3.6 with the extra points which affected mostly the best grades (grades 3 to 5) but had little effect on the lower grades.

![Figure 1. Level of completing the course elements](image1)

![Figure 2. Proportional distribution of course grades](image2)
Based on the final survey about 25% of the students had used the lecture recordings, and many individual requests for videos also from other courses suggested that students found them useful. On the negative side the proportion of students that did not complete the course project increased in 2008. The proportion of students getting a grade from the course is not yet final since only three of the four course exams have been given so far.

4. Discussion and Conclusion

In this paper the improvement of the technical infrastructure of a programming course during four years was described. The work was conducted as action research with annual course implementations. Each improvement effort was started by analyzing the data from previous course and identifying the biggest problems that were tackled with actions. In the data analysis qualitative data that was used to identify issues and quantitative data was then used to validate the issue and estimate its importance for the students at large. Since student responses to the open questions were mostly short, they offered only limited insight into the issues but still provided a general understanding of the problems.

The course improvement efforts have focused so far on the technical infrastructure covering the programming tools, study material, weekly assignments, project, and study material. The conducted measurements demonstrate a tangible improvement in the course results, especially in the proportion of students passing the course, but the limit of improvement possible with efforts focusing on the technical infrastructure appear to be approaching. Even if deficiencies can still be identified in the technical infrastructure, their importance and effect on improving the course results is becoming questionable.

Assuming that the course performance improvement through technical infrastructure is approaching its limit, the next step would be to move the improvement focus on teaching and learning. When the improvement work started four years ago, our working hypothesis was that the used technology had become an impediment that made the learning harder than necessary for the students. Our current working hypothesis is that we are likely to gain only marginal improvements with the current used technologies. Thus, to further improve the learning outcomes in the course, we have two options. We can either find better technologies to support the course, or move the improvement focus on the pedagogical aspects and try to help students to learn better by providing teaching that fits their learning styles better.

5. References


