An introductory course for software engineers

Fredrik Georgsson, Jonny Pettersson

Department of Computing Science,
Umeå Institute of Technology, Umeå University, Sweden

ABSTRACT

The aim of this paper is to show one example of how an introductory course for software engineers could be organised. Our course introduces the students to the main ideas of CDIO, allows them to practise conceive, design, implement and operate in a complex team based environment while developing basic communications skills. Furthermore, the students are introduced to ethical issues concerning working as a software engineer and will meet with professionals and learn what generic skills industry expect from students. The course is popular with the students and has made them understand the need of incorporating the practise of generic skills with the learning of technical and scientific knowledge.

KEYWORDS

Introductory course, software engineering, ethical issues, professional skills.

BACKGROUND

In a study program adapted to the ideas of the CDIO initiative it is essential to introduce a framework for engineering practice early in the education. The importance of this simple fact is captured in standard 4 of the CDIO. When developing the introductory course at our department we wanted a course in software engineering that would enable the students to

- understand the concepts of CDIO and how it is incorporated in their study program,
- practise conceiving, designing, implementing and operating in a team based environment,
- learn the basics in oral and written communication and working within a team,
- reflect on the ethical issues of being a software engineer,
- understand their future profession and start the process of being able to set goals for their future career.

This paper describes the introductory course and some design choices we made in the development of the course to meet the defined needs.

The outline of this paper is as follows; first, the underlying design choices of the course are described. Second is it shown how we put CDIO into a software engineering context. Third, the generic skills trained during the course are described. Finally, some quantitative and qualitative results of the course development are given.

THE MAIN IDEA OF THE COURSE

During the development of the course we used constructive alignment [1] to make sure the students would be able to reach the set learning outcomes of the course. The constructive alignment of the course are summarised in Table 1.

In order to practise conceiving, designing, implementing and operating in a team based environment the course was chosen to be project based to a large extent. The class was divided by the teachers into teams of five students and the task of the project was to implement a game suitable to twelve-year-old girls to be launched at a specified website. The website (kpwebben.se – in Swedish only) has a relatively strong ethical integrity and the game should be adapted to this. The teams should use the tool GameMaker (www.yoyogames.com) to do the implementation.

The project-aim was carefully selected. The challenge to construct a game made the task fun and interesting. It was also possible to use a powerful tool that required absolutely no prior knowledge in programming. All implementation work in GameMaker can be done without any code-writing. The reason for this was that we wanted to give everyone in a project group a chance to contribute to the project, not only those who had previous experience of software development. The target group of the game, twelve-year-old girls, made sure the students had to work carefully during the conceive and design phases of the project. The students (typically nineteen-year-old males) could not solve the task by introspection; they had to look outside themselves to figure out what twelve-year-old girls want from a game and then be able to construct such a game. The purpose of setting the target group of the game to girls was to naturally initiate discussions on whether there were differences in game preferences between boys and girls. The purpose of the ethical integrity of the proposed website was to make it possible to initiate discussions on ethical issues. Discussions on the ethical aspects of being a professional engineer were also initiated in the context of working together in a group.

The work with the project was divided into four different parts, where each part was devoted to each one of the steps in C-D-I-O. It was, however, made clear to the students that even if working with the project followed a rather strict waterfall model [2] this is not the case in real software engineering projects. We clearly pointed out that the simplistic waterfall model was for educational purposes and that they later on in their education would learn more realistic and agile project models.

The two study programs that have the course in their syllabuses are the five year master's program in computing science engineering and the three year bachelor's program in computing science. Since one aim of the course is to introduce the study program to the students, the program directors of the two study programs were responsible teachers on the course.

LEARNING HOW TO CONCEIVE, DESIGN, IMPLEMENT AND OPERATE IN A SOFTWARE ENGINEERING CONTEXT

At the very first lecture of the course, a discussion regarding different ways of organising a study program was initiated. In a teacher led discussion the students were made to see the benefits of having a strong context for learning and letting this context be an active part of the study program. It was explained why CDIO is a valid context and some examples of how CDIO influence the study program were given.

Each of the four C-D-I-O-parts of the course lasted approximately one week and the students devoted about 50% of their time to this introductory course. The other 50% were devoted to

an introductory course in mathematics. Each of the four parts of the course started with a lecture where the current concept was explained and exemplified in a software engineering context, as is described below.

Conceive – Problem solving basics, project plans, setting system goals, forms for working together, performing basic investigations,

Design – The design process, an in-class workshop on creative processes', brain storming, scenarios, personas,

Implement – Building the correct thing vs. building the thing correctly, testing, an introduction to extreme programming, pair-programming, time-boxing, the pomodoro method,

Operate – The importance of maintenance, tips on writing manuals, organising support, handling software errors, different types of software errors.

The different phases of the project work were assessed in the following way:

Conceive – At a cross-team seminar the conceive phase was discussed and special focus was put on the teams' findings on what a twelve-year-old girl would demand from a computer game in order for it to be "interesting". Some focus was also put on the implementation tool, what could and could not be easily done with the tool. The seminar was monitored by a teacher and at the seminar some teams were recommended to improve their research on the expectations of twelve-year-olds. The teacher also made sure that all students participated in the discussion and checked that all students were well prepared.

Design – At a cross-team poster session the game design was presented. Each member of the team had to orally present the poster to the cross-team during five minutes. During the following five minutes the cross-team asked questions and came up with suggestions of improvements. After the session the team had to collect all comments that the team members had received during the cross-team session. In this way the team received feedback from four different cross-teams. The posters were also assessed by the teachers and feedback on the poster design was given at tutoring sessions. The tutoring sessions are further described in the Learning Generic Skills-section.

Implement – The implementation was presented at an oral presentation where each team was given 20 minutes to present their implementation of the game and to justify design choices etc. The presentation was to be aimed at a teacher of the course. The audience (i.e. other teams) gave feedback to the team's presentation. Each member of the team had to have an equal amount of "floor time" during the presentation and everyone received individual feedback on their performance in the presentation from a teacher afterwards.

Operate – A group of twelve-year-old girls and a member of staff evaluated the games. The girls focused on the gaming experience and graded the games on a variety of different topics. The member of staff focused on the potential of the game concept. At a public presentation each team was given five minutes to pitch their game to a general audience and the pitch was assessed by a member of the staff.

LEARNING GENERIC SKILLS

In addition to the CDIO-lectures, there were also lectures on topics like oral communication, group processes, written communication, professional ethics and setting personal goals. Materials regarding scientific writing, oral presentation and how to make a poster was available online.

Every week each group of students was tutored by the program directors. During these sessions one particular topic was discussed and the previous week of the course was evaluated. Examples of topics discussed were: the anxiety of oral presentations, how to study efficiently, problems involved in writing reports etc.

In addition to the project work the students were also assigned the task of writing an individual report on their professional role as software engineers, focusing on the ethical aspects and the generic skills that would be expected from them as professionals. To aid the students in their work, twelve companies were invited to present themselves and participate in a panel discussion. The companies were asked to focus on technical aspects of their work in their presentation and were informed that the students would be interested in knowing more about generic skills and ethical aspects during the panel discussion. During the afternoon there were several occasions when students and professionals could mingle. Several companies were represented by more than one person so the student to professional ratio was about four to one, making mingling meaningful.

The individual report should also contain an analysis of correlations between the observations drawn from interacting with the professional software engineers, the CDIO-syllabus [3], the syllabus of the study program and the ACM Code of Ethics for software engineers [4]. The purpose of writing this analysis was to make the student reflect on

- whether the study program was aligned with the formal documents of CDIO and ACM
- whether the syllabus's of CDIO and ACM was aligned with the needs of the industry
- and finally whether the study program would help the students reach their goals of being efficient software engineers after graduation.

RESULTS

At the student appraisal for 2010 the students gave the course a 4.3 grade on a 5 grade scale. The students were particularly pleased with the lectures, the project and meeting the professionals. Some of the students felt that the report on their future professional role was unnecessary.

The course was similarly organised in 2009 and besides being popular with the students we have noticed a larger acceptance amongst the students for including the practise of generic skills in other courses. In addition to this, a course in interaction design, which previously was considered "fuzzy", largely due to the fact that it contained no programming, has gained in popularity amongst the students after the introduction of the new introductory course.

The introductory course has also been accompanied by a short term-introduction for each year at the very beginning of each term. At these introductions the program director meets with the entire class over an informal lunch. Each course in the upcoming term is introduced and its role in the program is explained and discussed with the students. In addition to this the past term is followed up based on an informal discussion taking its origin in the students' appraisals of the courses last term. The students find these term-introductions very valuable.

The idea of using a website with strong ethical standards did not work as well as intended. The reason is simply because the students soon understood that their games were not really to be launched at the website this requirement did not become as steering as we intended. Still we believe that all games developed would have met the ethical requirements of the website. To use twelve-year-old girls as target group was very successful. At first it was clear that the students projected their own values on how a good game should be and these

values was, in some cases, definitely not what the target group wanted. One example was high-score lists. For nineteen-year-old men high score lists, preferably lists available on-line, seems to be essential of the gaming experience. For tweleve-year-old girls it is not. In fact, the girls did not seem to be that interested in scores at all.

The games constructed by the students can be found at http://www8.cs.umu.se/kurser/5DV107/HT10/spel/
Instructions are mostly in Swedish.

Table 1 Constructive alignment of the course

Learning objective	*Assessment*	*Teaching*
Be able to understand and perform the conceive phase of a project	Seminar	Lectures, tutoring
Be able to understand and perform the design phase of a project	Poster session	Lectures, tutoring
Be able to understand and perform the implementation phase of a project	Oral presentation of project	Lectures, tutoring
Be able to understand and perform the operate phase of a project	Oral presentation of product, target group evaluation	Lectures, tutoring
Working together in a group	Project	Lectures, workshop, tutoring
Understand the skills needed for a professional software engineer	Report	Lectures, workshop, meeting with professionals
Insight into the ethics of being a professional	Report	Lectures, meeting with professionals
Oral communication	Discussion seminar, oral presentation	Lecture, tutoring
Written communication	Report, poster	Lecture, tutoring

REFERENCES

[1] Biggs, J.B. (2003). *Teaching for quality learning at university.* Buckingham: The Open University Press.

[2] Sommerville, I (1990). *Software engineering*. Addison-Wesley.

[3] CDIO. *CDIO Syllabus Report*. http://www.cdio.org/framework-benefits/cdio-syllabus-report

[3] ACM. *ACM-Code of Ethics*, http://www.acm.org/about/code-of-ethics

Biographical Information

Fredrik Georgsson is a senior lecturer at the Department of Computing Science at Umeå University. He is the program director of the five years master's program in Computing Science and Engineering. In addition to this he also works with increasing the industrial involvement in all engineering programmes at the Umeå Institute of Technology.

Jonny Pettersson is a lecturer at the Department of Computing Science at Umeå University. Previously he was the study director of the three years bachelor's program in Computing Science. He has a strong interest in methods for personal and leadership development.

Corresponding author

Dr. Fredrik Georgsson
Department of Computing Science
Umeå University
SE-901 87 UMEÅ, Sweden
+46 70 242 1099
fredrikg@cs.umu.se