Learning Outcome Outcomes: An Evaluation of Quality

Daniel Brandur Sigurgeirsson
School of Computer Science
Reykjavik University
Reykjavik, Iceland
danielbsig@gmail.com

Marta Larusdottir
School of Computer Science
Reykjavik University
Reykjavik, Iceland
marta@ru.is

Mohammad Hamdaga
School of Computer Science
Reykjavik University
Reykjavik, Iceland
mhamdaga@ru.is

Mats Daniels
Department of Information Technology
Uppsala University
Uppsala, Sweden
mats.daniels@gmail.com

Björn Þór Jónsson
School of Computer Science
Reykjavik University
Reykjavik, Iceland
bjorn@ru.is

***Abstract*— Learning outcomes are a standard specification of knowledge, skills and capabilities that a student is expected to acquire by attending a course or a degree program. While, in theory, the process of evaluating learning outcomes appears to be trivial, in practice it is a complicated and daunting process. In this study, we evaluate how learning outcomes can be effectively applied. The work focuses on the quality of both the specification of the learning outcomes and the assessment of whether these outcomes are reached. We discuss different abstraction levels for learning outcomes and the issue of alignment between high-level and low-level learning outcomes. We also address the criteria for assessing whether a student is meeting a learning outcome.**

Our work is focused on project-oriented courses, where assessing learning outcomes is seen as particularly challenging. In particular, we draw on an empirical study focused on systematically collecting key performance indicators of the progress towards achieving learning outcomes. The data gathering was done during the course through in-class questionnaires and individual diary notes, as a complementary process to the traditional observations made by the teacher running the course. This data serves as the basis for understanding how individual students advance towards the stated learning goals. We also conducted a focus group discussion after the course to better understand how to interpret the data collected during the course.

An important result of our work is forming an understanding and vocabulary regarding learning outcomes and the assessment of how well students meet these learning goals in project-based educational settings. In addition to this, we make the following major contributions:

- **We present a systematic methodology to gauge how well students meet learning outcomes through in-class self-evaluation.**
- **We present the results of an empirical study of a process-oriented evaluation of the students' development towards stated learning outcomes.**
- **We state some lessons learned from this process that are applicable for designers of project-based courses.**

Keywords—learning outcomes; self-evaluation; project-based course.

I. INTRODUCTION

Learning outcomes are used as a baseline to measure the value gained by students taking a particular course or a degree program. In theory, students who pass a course should have achieved the learning outcomes of that course. For this reason, formulating high-quality learning outcomes is a crucial requirement for measuring the quality of an educational program. There are several approaches for assessing whether a student has achieved the intended learning outcomes as well as the student's level of mastery. For example, this can be done by grading project reports or written exams. But the question is how we determine whether the formulated learning outcomes are appropriate to ensure that the course is training students towards the intended learning? In other words, how do we measure the quality and suitability of learning outcomes, particularly for a project-based course, where the evaluation is hard to quantify?

Normally, faculty members rely on general guidelines for designing the learning outcomes of a course. These guidelines need to adhere to standards within each field of study. For computer science programs, the most mature of these are the ACM/IEEE computer science curricula 2013 (henceforth referred to as the "ACM Curricula") [1] and the European Quality Assurance Network for Informatics Education (EQANIE) standard [2]. A quality assurance process com¬paring the learning outcomes of a curricula to the ACM standard shows that the process raised the awareness of the faculty members on how abstract topics and learning outcomes from an international standard can be used when revising the curricula of a particular course [3]. Review of quality within a discipline such as Computer Science is very much dependent on the specifics of national quality assurance processes [4]. Similar approaches have become more common due to the importance of factors such as globalization and international harmonization frameworks, such as the Bologna process [15]. One such approach is the use of international subject benchmarks, which has the attractive feature that it is based on a transnational definition of quality.

Standards provide frameworks that describe learning out¬comes that should be included in the curricula and can be used to evaluate and monitor both individual courses and programs as a whole. Unfortunately, these standards focus on content and the guidelines provided are more tailored to theory-based rather than project-based courses.

Designing the learning outcomes for a project-based course and ensuring that the course satisfies the proposed learning outcomes is a daunting task. Students' evaluations of a particular course are used internationally as one indication of the quality of a course, usually measured through questionnaires. However, the surveys are commonly focused on the teaching and content delivery. These evaluations are typically done after the course is completed and focus on the course in general rather than each particular learning outcome. This paper presents an empirical study, where we followed a systematic approach for collecting key performance indicators to evaluate the learning outcomes of a project-based course based on in-class student feedback. The approach used in this study focuses on gauging the students' work towards learning outcomes. This is done via

in-class questionnaires on their achievement of each of the learning outcomes.

The course in question was given at Reykjavik University (RU) as part of the computer science program. The objective of the course is that students gain skills in practical software development, by changing and improving code that other developers have written. Addition¬ally, the students have to evaluate how well these changes integrate to the whole functionality of the system. To do this, the students must read code and documentation from other developers to be able to improve the code.

The course was offered as a three-week intensive course. After each week, the students filled in a questionnaire, evaluating their skills and competencies according to the pre-described learning outcomes for the course. In this paper, we compare the results from these three surveys. Additionally, we conducted a focus group shortly after the course, for better understanding of the results of the surveys. Furthermore, the students wrote diaries, and one of the authors observed the students through the course.

The paper makes the following contributions:

- We present a systematic methodology to gauge the quality of learning outcomes, via regular in-class student self-evaluation.
- We discuss the results from evaluating the impact of using student feedback for continuous improvement of learning outcomes of a project-based course.
- We present a list of "lessons learned" that can be used by project-based course designers.

The remainder of this paper is organized as follows: In Section II, we discuss the relevant background literature. In Section III, we describe the case that is used in this study. In Section IV, we describe the methodology, followed by the results in Section V. In Section VI, we discuss the results and outline the approach for student self-evaluation for evaluating learning outcomes. Finally, in Section VII, we summarize the results.

II. Background literature

Learning outcomes are a common tool to specify what knowledge, skills and capabilities students should gain from attending a course or a degree program. While this sounds simple in theory, the issue is quite complex. In this study we evaluate some aspects how the concept of "learning outcome" works. Our focus is on quality, both in the specification of learning outcomes and the assessment of whether they are reached.

As remarked by Woollacott [5], quality is a complex trait that requires a sophisticated understanding, not just of how well a particular product or service fits its proposed context, but also how those who use that service react to it. When defining quality, and implicitly when discussing issues of quality assurance, it is important therefore to identify the expectations of stakeholders with respect to what is being judged, as a vital component in examining performance. In a commercial context, it may be possible to make the claim that the customer, or contractor for the service or product, is the primary stakeholder, but when we move into the area of higher education the identification of a single customer appears to be problematic. Certainly, the simple assignment of the customer role to the student is not straightforward [6]. If there are multiple stakeholders, who each have a claim to interest in the success of the subject-specific domain, then a more useful way of approaching this problem may be to consider the concept of "curriculum responsiveness". This is the degree to which the educational program set out by a university, both taken as a whole and as exemplified in subject-specific domains, should be able to engage with the legitimate concerns and interests of students, academics, employers, government agencies, and the wider society [7]. In his discussion of quality education in engineering disciplines, Woollacott identified four kinds of curriculum responsiveness associated with four principal stakeholders [5]. The first is economic responsiveness, which deals with how the curriculum reacts to the demands for highly qualified workers who can engage in the tasks necessary for the smooth running of modern globalized economies. The second is disciplinary responsiveness, which is a function of how well the curriculum can adapt itself to ensure that students receive an education that is informed by the best scholarship and academic and professional practice. A third type of responsiveness is cultural or societal in nature and depends upon how easily the educational system can incorporate the cultural diversity of students, while the fourth is learner responsiveness, which reflects how the curriculum can accommodate the individual learning needs of students. The stakeholders in the first, second and third cases are the workforce, the discipline, and society as a whole. Only in the fourth case is the student the primary stakeholder and consequently only if we ignore the wider socioeconomic and academic consideration of higher education do we arrive at a concept of "student as customer". Instead, by allowing for multiple groups with a range of views, we find that the optimal situation would be that the sector should attempt to satisfy the expectations of as many stakeholders as possible.

It is natural to focus on the question of what makes a product or service fit for purpose, when assessing the quality of it. In terms of higher education, this is related to issues of curriculum and hence to its responsiveness to the variety of stakeholder expectations. Nevertheless, there is still the question of what exactly we measure in order to arrive at an assessment of the quality of educational provision. If one of the outcomes of higher education is, say, that we should produce a skilled workforce, then we need to know what it is that differentiates a skilled worker from an unskilled one. Similarly, if an objective is to produce graduates with a range of cognitive skills that will allow them to function well in a changing learning environment, we need to know how to assess whether efforts to accomplish this have been successful. This is usually done by trying to relate proficiency in some competency that a learner should acquire or develop to the measurement of some aspect of performance of a task that would rely on that competency. The difficulty is that competencies are internal attributes that can only be assessed by measuring some indicator or proxy during the completion of some task. The nature of this performance, and hence the indicator for competence that is measured, has certain inherent ambiguities. If one thinks of performance in terms of output, then the assessment is made by examining the quality of deliverables produced in the task. This type of outcomes-based assessment is one methodology that is used to derive measures of quality. Alternatively, one can think of performance in terms of the activities that underlie the output and focus on the behaviors

and processes required for successful completion of the task. This would lead to a process-based quality assurance methodology. Recent developments in higher education quality assurance have tended to promote an outcomes-based approach that focuses on the demonstrable products of the educational experience as expressed in learning outcomes. As pointed out by Ewell [8], this is partly due to the demands from public funding bodies that universities adopt a "culture of evidence" characterized by the continuous collection and use of evidence about student learning to improve teaching and the general educational experience. Nevertheless, the use of outcomes-based assurance methods can be justified on a number of sound pedagogical considerations. The requirement that course objectives and performance indicators be articulated in terms of the learnt competencies of the students themselves is much closer to the constructivist paradigm and its implementation in models of student-centered learning supported through such mechanisms as constructive alignment [9, 10]. At the level of the individual student, such learning outcomes are expressions of what learners are expected to achieve and how they are expected to demonstrate this. They necessarily focus on the student's competence rather than the process through which instruction is conveyed. This outcomes-based approach has been adopted in many countries as a result of the Bologna Process [11], which seeks to develop a common model for higher education throughout Europe in order to devise a common framework that would allow explicit comparison to be made between elements of different national higher educational systems. The aim was that all programs should be based on learning outcomes and, where necessary, curricula should be redesigned to accommodate this task. This has resulted in the development of national qualifications frameworks [12] in many European countries, (e.g. the Framework for higher education qualifications in England, Wales and Northern Ireland, the Scottish Credit and Qualifications Framework, the Icelandic Qualifications Framework, etc.) in which degree programs are mainly described in terms of competence-based learning outcomes.

III. The Study

This section describes the course history, the current structure of the course, the learning outcomes, and the background of the students participating in the study.

A. Course History

The bachelor program in computer science at Reykjavik University started in 1998 and the taught content was, at that time, strongly influenced by the 1991 version of the ACM/IEEE computer science curriculum [13]. The program had an extensive review in 2008 based on the 2001 version for the computer science subfield (ACM/IEEE 2001). The program includes 17 mandatory course units in computer science and mathematics for a total of 102 ETCS (one ETCS is 1/60 of a "student year") and a mandatory final group work project that is 12 ECTS for each student. In addition, students can select between four "emphasis lines", each of which consists of 30 ECTS in courses related to the focus subject of the emphasis line. At Reykjavik University, the semester is currently split up into two periods: a 12-week period, where students are typically enrolled in four courses, and a 3-week period, where students can focus on a single course.

The "RU Internship" course, which is studied in this paper, was created for the spring semester 2013. It was set up as a 6 ECTS course, and students were allowed to enroll twice in the course. In the course, students focus entirely on a single project of practical maintenance and redesign of a Learning Management System (LMS). Reykjavik University was using a system, which implemented a combination of a Learning Management System (LMS) and a Student Management System (SMS). The system was written in Classic Active Server Pages (ASP) and had a number of design flaws, and it was clear that the system would be replaced within a short timeframe. We therefore decided to use the course to build a prototype for a new system, solving the same user goals. The aim of the course was to gain insights into the requirements for a new system based on the experience of studying the current system. The student-developed system was named Centris and was proposed as a candidate for replacing the old system. In total, until spring 2017, 182 students have been involved in the project and have spent more than 30 thousand man hours working on the Centris project. The study described in this paper occurred during the spring 2015 semester during a 3-week period.

B. Structure of the Course during the Study

During the study, in spring semester 2015, the course was offered during the three-week period, for a total of 15 days. The focus in this course offering was first and foremost on the LMS part, but with occasional tasks originating in the SMS project. Students worked full time, typically from 9 in the morning until 5 in the afternoon, on assignments related to the project.

The students worked in pairs that they chose themselves, since this arrangement had given the best results in previous offerings of the course. First, the students were asked whether they preferred a particular partner to work with. Students with no particular preference were paired together, such that no pair had two members with a low average grade. Students who requested a particular partner had no restriction on the pairing. Each pair was assigned a single programming or code review task to begin with. Each task was planned such that it would be completed within three days at most. Tasks were assigned by the instructor, and once they were completed the changes to the code were reviewed by the instructor and three students that had taken the course previously. Students received guidance from the instructor at the time of task assignment (i.e., hints on how to implement the task and where to look in the source code). The instructor was also available to guide students upon requests. Teams had no restriction on following a specific development methodology (e.g., Scrum, Waterfall) for internal communication. However, there was one meeting in week two where students could discuss the progress. Moreover, students used a Facebook group for various informal discussions and questions during the course.

For logistical reasons, the enrolled students had to be divided into two groups in two different types of classrooms: a traditional classroom, and a teamwork room. The first room had a traditional setup: rows of tables with all chairs facing the instructor. The other room had a very different setup: one circular table and one long table, where students sat facing each other with no instructor seat in that room, and a single whiteboard on wheels which could be moved around. This setup was accidental but turned out to

provide us with useful data about the effect of classroom setup on project-based learning. Finally, and also for logistical reasons, some third-year students had to help first-year students in another course that was taught by the same instructor. Each student had one day, out of the 15, which they dedicated for peer assistance. This was good for the learning outcomes, because the students experienced by giving support to others that it is sometimes hard to understand the questions the students had. This experience taught the students to be rather structured, while seeking assistance in the course of the study.

C. The Learning Outcomes

The learning outcomes on the occasion of the study were written with the following goals in mind:

- The completion of the learning outcomes would be necessary as a preparation for a career in software maintenance.
- The learning outcomes would match what we had concluded, would be achievable in the course. The definition was based on the experiences from previous versions of the course.

The learning outcomes of the course are nine; seven at the "skills" level and two at the "competencies" level. The learning outcomes are shown in Table 1. The learning outcomes were written by the instructor of the course and reviewed repeatedly by key members of the faculty, until all stakeholders were satisfied. These learning outcomes were used as one of the bases for data collection in this study.

D. The Students Participating in the Study

27 students took part in the study. 24 had no prior experience in the course, while 3 students had completed the course already but wanted to gain more skills and competences in the subject. All students had completed at least three semesters (90 ECTS). Five students had completed four or five semesters (120 or 150 ECTS). The ratio between female and male students was around 1:4. The average age of the students was 25.8 years; the youngest student was 21 while the oldest was 38.

TABLE I. THE LEARNING OUTCOMES

At the end of the course, students should be able to:	
#	***Skills***
LO1	Read code, written by others, and describe its functionality
LO2	Improve code and documentation written by others
LO3	Apply the various standards and rules set by the project
LO4	Apply best practices in version management
LO5	Address issues raised by code reviewers
LO6	Operate in a group, which is itself a part of a larger team
LO7	Present (either orally or in writing) your work
#	***Competencies***
LO8	Identify which parts of a project documentation needs improving
LO9	Identify which parts of a project codebase needs improving

The most frequent age was 22, which is no surprise given the fact that most of these students start their study at the age of 20 and most of them were in their 4th semester. We believe that this group of students represented a cross-section of the student population at the time.

The students rated their experience of programming prior to attending the course. Around 60% stated that their experience had come solely from school projects, while 12% had some experience through summer jobs and 28% had proper industrial experience.

IV. DATA GATHERING METHODS

The data gathering methods were four: questionnaires; diaries; instructor's observations; and focus group discussions. In this section we explain the processes for data gathering and data analysis.

A. Questionnaires

Students answered five questionnaires of three types: (i) background questionnaire, (ii) three weekly surveys and (iii) a post-course survey. Those questionnaires were conducted to understand the students' background, gather data on their self-evaluation with regards to learning outcomes, and to assess their opinion on the whole course. Below we explain each type of questionnaires in detail.

(i) Background Questionnaire: This survey was prepared to study the background of the students, and their expectations from the course. The questionnaire includes seven questions. Five questions are generic background questions and two questions target students' self-evaluation with regards to the nine learning outcomes of the course. In total, 25 students out of the 27 completed the questionnaire.

The two questions on the learning outcomes were:

1. *In your own words, describe what you think each learning outcome really means.*

There was a text field for each LO. The students were allowed to answer in their native language.

2. *How good would you consider yourself in the following (LO1 – LO9), compared to your fellow students in the School of Computer Science?*

Students were given the following answer possibilities: a) Well below average; b) Below average; c) Slightly below average; d) Average; e) Slightly above average; f) Above average and g) Well above average.

(ii) Weekly Surveys: The students answered the weekly questionnaire three times, i.e., shortly after the end of each week. The purpose of this survey was to monitor the progress of the students, whether they perceived that they were making progress on the learning outcomes of the course, and their perception of the project itself. 21 students answered the first weekly questionnaire, 25 students answered the second questionnaire, and 19 answered the third questionnaire.

Students were asked to rate their improvement on each of the learning outcomes and also to rate how much the project had improved.

The question on the learning outcomes was:

Looking back on the week, to what degree did you improve your knowledge and skills in the following (LO1 – LO9)?

The students were asked to rate their knowledge on the following scale: a) Did not apply this week; b) I have not improved in this area at all this week; c) I have improved somewhat this week but I am still not competent; d) I have improved somewhat and feel competent and e) I feel like I have mastered this topic.

(iii) Post-Course Surveys: The students answered a questionnaire when the course had been completed. 19 students answered the survey. The aim of this survey was to learn more about the impression students had of the learning outcomes, and how they compared to their impression before the course started.

This questionnaire had four questions, of which two were about the nine learning outcomes. These questions are:

1. *How good would you consider yourself now in the following (LO1 – LO9), compared to your fellow students in the School of Computer Science?*
2. *In hindsight, how good do you think you were when the course started (related to LO1 – LO9), compared to your fellow students?*

For both questions the students had the following answer alternatives: a) Well below average; b) Below average; c) Slightly below average; d) Average; e) Slightly above average; f) Above average and g) Well above average.

Results were converted to a numerical scale between one and seven, where one means "Well below average" and the seven means "Well above average". Then, an average for all students was calculated for each learning outcome.

All students who had previously attended the course were asked to read and comment on the questionnaires. Several comments were received and the questions were revised accordingly. The questionnaires were posted online. Students had to identify themselves in each questionnaire using their national identification number (ID). Students could answer the questionnaires at any time and were able to complete them in isolation. All questionnaires were designed and hosted at freeonlinesurveys.com.

B. Diaries

During the course, each student kept a diary, where they noted what they had done during each day. The diaries were stored in shared documents on Google Drive that were open for all students in the course, as well as the instructor. So the students could see each other's diaries. Students were reminded periodically to fill in the diary, usually via a post to the Facebook group of the course. The students were given the following instructions:

"For each day, write down in a single paragraph or two what you did today. In particular, include any of the following if applicable: Were there any "a-ha!" moments, and if so, what caused it? Was there anything that made you particularly frustrated? If so, what? What was the complexity of the tasks you had today? Was there any noticeable improvement in your understanding today, or did the code improve significantly? What gave you the best support in solving the tasks? This could include Assistance from teacher, Assistance from other students, Project meetings, Project documentation, Project videos, Videos for other courses (Web Programming II, Web Services, etc.), Code review, Comments from reviewers on Pull Requests, External resources (stack overflow, blogs etc.)"

C. Focus Group

A focus group was held three weeks after the course. The aim of the focus group was to gain further insights into the impressions of students, and to give the students the opportunity to bring up topics which were not mentioned in the surveys. All students were given the opportunity to join, but only four students actively participated. We believe that the lack of participation was due to the fact that most students had already started their summer jobs.

There were nine questions prepared that the focus group covered. The questions were on the learning process of the students and whether they learned what they anticipated, how productive the students felt and their perspectives about working in groups. Also, there were questions on seeking help and being stuck in a given assignment. Two questions related particularly to the study described here:

1. *What did you think of answering the surveys and keeping the diary?*
2. *When answering the question "In hindsight, how good do you think you were when the course started, compared to your fellow students?" how did you compare yourself to other students?*

Each participant was given the chance to answer each question and to express themselves freely. The seating arrangement was informal, i.e., participants were sitting facing each other, with no table in the center. The conversation was recorded and participants were informed that the recordings would only be used as a part of the data collection and would be destroyed afterwards. The students participated equally (more or less) in the conversations.

D. Instructor's Observations

During the course, the instructor observed the behavior of the students. This was done in an informal way. Several interesting points were collected.

TABLE II. STUDENS' UNDERSTANDING OF LEARNING OUTCOMES

Learning Outcome (LO)		***Average***
LO1	Read code, written by others, and describe…	0.98
LO2	Improve code and documentation written by…	0.91
LO3	Apply the various standards and rules set by…	0.90
LO4	Apply best practices in version management	0.86
LO5	Address issues raised by code reviewers	0.91
LO6	Operate in a group, which is itself a part of…	0.99
LO7	Present (either orally or in writing) your work	0.98
LO8	Identify which parts of a project documentation…	0.85
LO9	Identify which parts of a project codebase…	0.88

V. RESULTS

A. Understanding of Learning Outcomes

Prior to the course the students were asked to explain each learning outcome in their own words (in their native language). Those answers were by one of the authors, who gave each answer a numerical value between zero and one, where zero indicates that the student did not understand the learning outcome at all, and one indicates that the student fully understood the learning outcome. The average grade for each learning outcome was then calculated. The results are shown in Table 2.

The learning outcomes were mostly well understood. Two students did not understand LO4 at all, and two students did not understand LO8. Other answers indicated either full understanding, or that students mostly understood what each learning outcome meant. It was interesting to note that in LO5, many students mentioned that it was important to be able to accept criticism without taking it personally, i.e., not letting their ego get in their way.

B. Students' Weekly Self-Evaluation

For each of the learning outcomes, the students were asked to rate their improvement, by selecting one of the following: (a) Did not apply this week (b) I have not improved in this area at all this week (c) I have improved somewhat this week but I'm still not competent (d) I have improved somewhat and feel competent (e) I feel like I have mastered this topic. Figure 1 shows the results. Each learning outcome has three horizontal bars: the topmost bar represents week 1, the middle bar represents week 2, and the bottom bar represents week 3. Within each bar, the five choices are colored with different colors.

It is interesting to note that in three cases (LO1, LO5, and LO6) the number of students that said they "had mastered the topic" dropped between weeks 2 and 3.

The impact of LO3 seems to have been the most in week 1, as most of the students completed their first tasks in week 1 and were forced to ensure that the changes they made to the code were in conformance with the coding standard of the project. In weeks 2 and 3, most students seem to have adjusted to the requirements of the course in this regard.

Since students did not have to present their work until the end of the course, LO7 is the one with the least change. This may due to the fact that students are required to present their work in other courses as well. In general, students seem to perceive that they did significantly improve in most of the learning outcomes in the course.

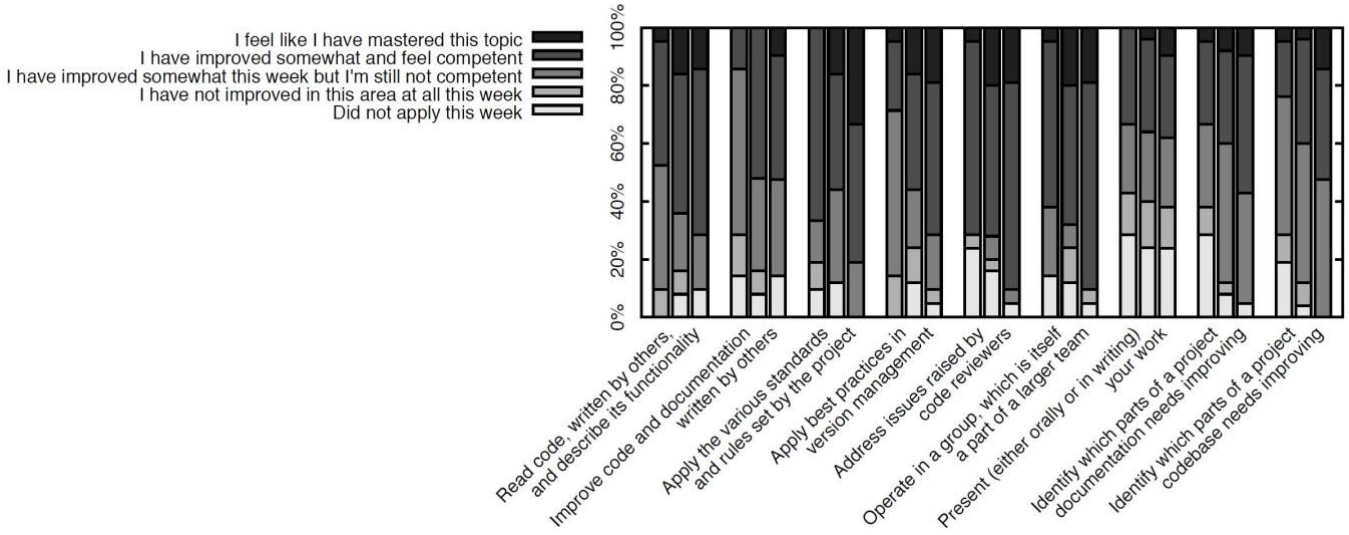
C. Students Self-Evaluation After the Course

Students answered a post-course questionnaire after they had completed the course. One of the questions was on their self-evaluation on their accomplishment of the learning outcomes compared to their fellow students. This question was asked both in the survey they answered before the course started, and after they had completed the course. In the post-survey, they were also asked to rate themselves in hindsight, i.e., to assess how qualified they believed themselves to have been before the course started. The results are shown in Table 3. There are a number of interesting points in these results. First, students generally rated themselves in the "Average" category for most learning outcomes. However, LO6 stands a bit out, since students rated themselves as "Slightly above average". This may be due to how group work is dominant at Reykjavik University, but that alone would not explain these results, since students were probably using their fellow students for comparison.

This particular fact was brought up in the focus group (see below), but the participants in that group did not have any clear explanation for this. Another interesting fact is that the grade they gave themselves in hindsight was generally lower than the grade they gave themselves before the course started.

The student answers in the diaries were usually short, i.e. one to two sentences per day. The students described what they were working on during a given day. They were instructed to describe any problems they had or discoveries during the day, but they did not do that.

FIGURE 1. IMPROVEMENTS IN LEARNING OUTCOMES



D. Results from Observations

The instructor observed that the students were not very active in requesting external help, such as by asking questions on stackoverflow.com, or by posting issues for libraries the students were using on their Github pages. Closely related to this was that students tended to get stuck on some problems. Many of the students seemed afraid to ask for assistance and seemed to assume they were supposed to solve the problems individually. Both of these topics were discussed further in the focus group.

Another observation was that the atmosphere in the classroom with the classical setup seemed less relaxed than in the other room, and there seemed to be less discussions in that classroom. Finally, it was obvious that some students were struggling more than others. In general, students who entered the course with a lower average grade and/or lower grades in the prerequisite courses seemed to have more trouble understanding how to solve their tasks. When those students were paired with students with a better preparation, they were more likely to get help from that student.

E. Results from the Focus Group

The main results from the focus group include that when asked what the students learned first and foremost, they mentioned that they had learned how to work in a large group project and that they learned by having to study code written by others. This corresponds to learning outcomes LO1 and LO6.

When asked whether they were happy with the performance of the group, they seemed very happy with how much the group achieved. They also mentioned that this helped them when they got stuck themselves, then at least they could see that others were making progress and therefore the project itself was not stuck.

Having the working area set up like a classic school room is not ideal, since it does not encourage teamwork. Students preferred one of the following: a) Round tables, with five to six students at a single table; b) Opposing tables, with four to five students in a row, and another row facing them; c) Tables set up in U-shaped formation. The reason given by the students was that this facilitates interaction between them, which the traditional classroom setup did not encourage.

There was quite some time spent on discussing how active students are in getting help when they get stuck with some problem. Students are not taught to ask questions online (such as on stackoverflow.com, in Github projects, etc.). In some cases, they are specifically discouraged from asking questions online. This makes them afraid that they are plagiarizing by using solutions or answers given to them online. Often, students seem to get stuck in some problem, and they seem afraid to ask for help, perhaps because they are not sure what the problem really is.

There were mixed emotions regarding the questionnaires and the diaries. Some students said that forgetting to log each day in the diary was causing them discomfort after a few days, when they had to think backwards about what they were doing. Others said that it was a nice way to complete each day, to write down what they were doing, thereby getting a better overview over their progress.

In general, students were very satisfied with the course, and said that they believed it was a good preparation for them for future studies and work roles.

VI. Discussions

In this section we discuss the results of the weekly surveys. In general, the results show that students reported improvement in the learning outcomes. There were three learning outcomes—LO1, LO5 and LO6—where there was a decrease in the number of students that replied "I feel like I have mastered this topic" between weeks 2 and 3. The difference was small in all cases. One possible explanation is that students got "overconfident" in week 2, but then realized in week 3 that they probably could learn a lot more.

Two learning outcomes—LO5 and LO6—stand out in terms of the number of students which seem confident in their abilities at the end of the course. In both cases, close to 90% of the students selected either "I have improved somewhat and feel competent" or "I feel like I have mastered this topic". The discussions in the focus group reflected this as well, as the participants mentioned both these as something they had learned in the course.

Learning outcome LO3 was the only learning outcome where the option "Did not apply this week" got more responses in week 2 than in week 1.

The learning outcomes were rewritten in 2015, and we believe that overall they represent very well the knowledge and skills a student should acquire by attending the course. Based on the results presented here, however, a few observations can be made:

- Learning outcome LO8 was noticeably least relevant. In all three weeks, almost half of the students either said that this learning outcome was not relevant, or that they did not improve at all. This is probably because the emphasis on this learning outcome in the course was minimal. There was only one presentation in the course at the end, when some of the students had already answered the survey. Given these results, this learning outcome needs some revisiting, either by changing or removing it, or by changing the setup of the course.
- LO1 talks about code and documentation at the same time. This could be split up into two separate learning outcomes.
- There should probably be a new learning outcome which emphasizes the students' ability to express themselves when presenting a problem with their code, and their ability to get an appropriate help from others via various channels. However, this may be something that should also be addressed in other courses, as it should not only be present in an elective course like the RU Internship course. We believe that this is an essential skill in software maintenance, and as such should be addressed in one of the core courses.

VII. Conclusions and Lessons Learned

This paper presents a systematic methodology to gauge how well students meet learning outcomes through in-class self-evaluation. An empirical study of a process-oriented evaluation of the students' development towards stated learning outcomes has been presented and discussed. Here we list of some of the lessons learned from this study and possible future improvements.

Data Gathering Process: In general, the approaches used for data gathering were systematic and helpful. We would definitely use the same data gathering methods in future.

More Focused Diaries: Diaries should be more focused, and students should explicitly state what their "feeling" about each day was (was it a success? was it a failure?), whether something was helpful (if so, then what), etc. In practice, the diaries turned out to be more like a list of tasks which the students completed, with not much insight into how they were completed, what was stopping them, or what was helpful. The setup of the diaries could be modified, possibly by providing the students with an example of what a typical entry could look like. It would also be informative to ask them to give each day a grade, where the grade would depend on how much they felt they achieved on that day.

Reference Points in Weekly Questionnaires: In the weekly questionnaires, when asking about improvement in a given learning outcome, it should be clearer what the reference is, whether students should only talk about the difference in the given week, or whether they should be using the entire course. It was mentioned in some of the options that they should use just a single week, but not all of them. Students were asked beforehand what learning outcome(s) they thought would be emphasized the most and what they thought was most exciting. It would have been good to ask again at the end of the course "What was really emphasized most?" as well as asking whether they thought differently about what they found to be exciting.

Analyze the Results Earlier: It would have been preferable to analyze the results from the questionnaires sooner, such that those analysis could have been used more to bring up certain topics in the focus group.

Follow Up after Industry Experience: The question "Does this course offer good preparation for software maintenance?" has not been asked in this work. We would certainly like to research this further, and we propose that this question should be asked after students have spent some time in industry, when they have gained experience and can reflect on how it compares to their experience from the course. In attempts to answer this question, further steps could be taken. We could use the Kirkpatrick Model [14] and add other measuring tools to address this question.

Acknowledgment

We would like to thank all the students that took part in the course and provided us with data for this study.

References

[1] ACM/IEEE Joint Task Force on Computing Curricula. Computer Science Curricula 2013. ACM/IEEE, 2013.

[2] European Quality Assurance Network for Informatics Education. Euro-Inf.: Framework Standards and Accreditation Criteria for Informatics Programmes, 2009. http://www.eqanie.eu/pages/quality-label.php Accessed: 26 April 2018.

[3] Larusdottir, M. K., Daniels, M.; McDermott, R., (2015). Quality assurance using international curricula and employer feedback. Australian Computer Science Communications, 160:19–27.

[4] McDermott, R., Daniels, M., Larusdottir, M. K (2014). Subject-level quality assurance in computing: Experiences from three national perspectives. Proceedings of the Frontiers in Education Conference (FIE) 2014, pg. 1–8.

[5] Woollacott, L.C., (2009). Taxonomies of Engineering Competencies and Quality Assurance in Engineering Education, Engineering Education Quality Assurance, 2009, pp 257-295

[6] Baldwin, G., (1994). The Student as Customer: The Discourse of "Quality" in Higher Education, Journal of Tertiary Education Administration Vol. 16, 1, 1994

[7] Moll, I., (2004). Curriculum responsiveness: The anatomy of a concept. In H. Griesel (Ed.), Curriculum responsiveness: Case studies in higher education (pp. 1–19). Pretoria: SAUVCA, South African ViceChancellors Association.

[8] Ewell, P.T. (2008). Assessment and accountability in America today: Background and context, New Directions for Institutional Research S1, 7–17.

[9] Biggs, J. (1999), Teaching for Quality Learning at University, Society for Research in Higher Education and Open University Press, Buckingham.

[10] Biggs, J. (2003), "Enhancing teaching through constructive alignment", Higher Education, Vol. 32 No. 3, pp. 347-364.

[11] Adam, S. (2006), "An introduction to learning outcomes", in Froment, E., Kohler, J., Purser, L. and Wilson, L. (Eds), EUA Bologna Handbook, Raabe, Berlin, p. B2.3-1.

[12] European Centre for the Development of Vocational Training (Cedefop), (2010), The development of national qualifications frameworks in Europe, Publications Office of the European Union, Luxembourg.

[13] Tucker, A. B. (1991). Computing curricula 1991. *Communications of the ACM*, *34*(6), 68-84.

[14] Galloway, D. L. (2005). Evaluating distance delivery and e-learning is Kirkpatrick's model relevant?. *Performance Improvement*, *44*(4), 21-27.

[15] Reinalda, B., & Kulesza-Mietkowski, E. (2005). *The Bologna process: Harmonizing Europe's higher education*. Farmington Hills, MI: Barbara Budrich.