The Company Approach to Software Engineering Project Courses

David Broman, *Member, IEEE,* Kristian Sandahl, *Member, IEEE-CS,* and Mohamed Abu Baker
Department of Computer and Information Science, Linköping University
david.broman@liu.se, kristian.sandahl@liu.se, mohamed.abubaker@liu.se

***Abstract*—Teaching larger software engineering project courses at the end of a computing curriculum is a way for students to learn some aspects of real-world jobs in industry. Such courses, often referred to as capstone courses, are effective for learning how to apply the skills they have acquired in, for example, design, test, and configuration management. However, these courses are typically performed in small teams, giving only a limited realistic perspective of problems faced when working in real companies. This paper describes an alternative approach to classic capstone projects, with the aim of being more realistic from an organizational, process, and communication perspective. This methodology, called the company approach, is described by intended learning outcomes, teaching/learning activities, and assessment tasks. The approach is implemented and evaluated in a larger Master's student course.**

I. Introduction

PREPARING computer science (CS) and software engineering (SE) students for real-world jobs in industry is a challenging task. Feedback from industry has shown that software engineering topics such as testing, code reviews, release management, and team work are particularly important from a real-world perspective [1]. To meet such demands, the ACM/IEEE-CS joint task force on computing curricula recommends the inclusion of software engineering projects in a computing curriculum [2]. In such a project course, students are typically working in teams and developing a larger software system for a particular customer. These project-based courses that can span the entire last year of a curricula are often referred to as *capstone courses*. Capstone project courses have existed for many years [3] and several success stories have been reported in the literature [4]-[6].

However, these projects are typically performed in small development teams (three to eight students) where information is shared informally between the team members. These kind of projects can be successful when learning how to apply skills such as design, testing, and configuration management in a project setting, but give a very limited perspective of the actual problems faced when working in a real company. For example, a small team can informally communicate the design of a system, but if there are more than 20 developers, a more systematic approach is needed. Hence, the overall pedagogic problem discussed in this paper is how to design a project course that both enables student learning and gives a realistic appreciation of the *organizational*, *process*, and *communication* aspects of industry-based projects.

This paper describes an alternative approach to classic capstone projects, whose key components are:

- Instead of having small project teams, students are organized in *simulated companies*, each consisting of approximately 30 students/employees.
- The simulated company faces a *transition of the organization* from a traditional line organization with several departments to an agile organization containing self-organized cross-functional teams.
- The assessment includes *grading with individual time budgets*, i.e., project members need to report time and prioritize work tasks.

The main contribution presented in this article is the unique combination of these key components. This educational methodology is called the *company approach* to software engineering projects. More specifically, the detailed contributions of this work are that:

- The central ideas and concepts of the company approach are outlined based on the theory of *constructive alignment* [7] (Section II).
- The design and implementation of a course based on this methodology are described. The course was given twice, in 2009 and 2010 (Section III).
- An evaluation of the company approach was performed by conducting a questionnaire-based survey with closed questions during the course, and a questionnaire-based survey with open questions seventeen months after the course was given (Section IV). The results are discussed in Section V.

II. The Company Approach

This section describes the general ideas and concepts of the company approach methodology in the form of a *course template*, i.e., a generic description from which a specific software engineering course can be designed. The course template is based on the framework of constructive alignment, originally invented by Biggs [7] and further developed by Biggs and Tang [8].

Constructive alignment consists of two separate aspects. The *constructive* aspect concerns the learner's view and is based on the theory of constructivism [9]. The main idea of constructivism is that the students (the learners) construct their own knowledge based on what they already know and by performing activities by themselves. Hence, from this point of view, teaching is not about transferring knowledge to the student by lecturing, but instead is about empowering the student for active learning.

The *alignment* aspect relates to the teacher's performance and the way in which a course can be designed to align the following elements:

- *Intended Learning Outcomes (ILOs)* - specific learning outcomes that a student should have mastered by passing the course.
- *Teaching/Learning Activities (TLAs)* - various activities that should result in the intended learning outcomes.
- *Assessment Tasks (ATs)* - tasks for assessing the student's performance in relation to the intended learning outcomes.

The rest of this section presents the company approach methodology based on these three elements.

A. Intended Learning Outcomes

The overall goal of the company approach is that the student should gain a fundamental understanding of problems and challenges that occur in a real-world software engineering project. However, to be more concrete, the ILOs are categorized within three specific learning perspectives: *organizational* (O1), *process* (P1 and P2), and *communication* (C1 and C2). The intended learning outcomes include, but are not limited to, the student's being able, at the end of the course, to:

- *(O1) explain the meaning of the various roles and organizational structures in a software engineering project.* This means that the student should be able to explain the underlying meaning of classic SE roles, such as analyst, project manager, product manager, configuration manager, architect, developer, and tester. Furthermore, he/she should also be able to explain the different roles in an agile methodology, such as self-organized teams, product owners, and Scrum masters in the Scrum framework [10].
- *(P1) analyze the pros and cons of processes in general, and the difference between classic project management and agile methodologies in particular.* This means that the student should be acquainted with various process frameworks so that he/she can analyze and draw his/her own conclusions as to the leadership and management of projects.
- *(P2) analyze the basic fundamental constrains that are inherent in software engineering project.* Examples of such constraints are calendar time (time to deadline), budget time (man hours), product functionality, and measurable quality factors.
- *(C1) reflect on communication challenges when working in a larger heterogeneous software project.* The aim is to capture today's real-world heterogeneous aspect of engineering projects, including the diversity of cultural backgrounds, languages, and technical skills among employees/students.
- *(C2) explain the rationales of artifacts, such as architecture document, requirements specifications, and backlogs.* This learning objective emphasizes the communication aspect of artifacts typically created in a SE project. The aim is that the student should use and create a particular artifact - for example an architecture document - because they need this for precise communication between the various stakeholders, rather than because he/she is just obeying the teacher's instructions.

B. Teaching/Learning Activities

In this course template, the aim is that the students run the whole project by themselves. The teachers should not lead the project but only guide and coach the students by asking the right questions. One of the main challenges of this kind of course design is to define this framework, so that the students have a high level of freedom to be innovative without being so loose that the intended learning outcomes are missed.

The rest of this section outlines the ten most important teaching/learning activities of the methodology.

1) Role selection: At the beginning of the project, the students apply for a job and a role in the company by formulating a curriculum vitae (CV) and an application letter. If more than one student applies for a particular position (e.g., department manager), a closed election is held by the students. Apart from the chief executive officer (CEO), who is played by one of the teachers, the students elect their own leaders and key staff. This addresses learning outcome (O1).

2) Company meetings: Each week the students arrange a company meeting which all employees, including the CEO, attend. This is the main coordination and communication meeting between employees and is the only formal meeting required by the framework. This addresses (P1), (P2), (C1), and (C2).

3) Requirements elicitation: The students are not given any requirements for the project. Instead, they are told that the CEO has closed an agreement with a potential customer to pay for a preliminary study of the product. Hence, one of the learning activities is to schedule meetings with the customer and elicit requirements. However, during the project, at least one more customer is introduced which means that students learn about changing requirements and the development of products targeted for many customers with different priorities. This addresses (P1), (P2), (C1), and (C2).

4) Preliminary study and business meetings: The preliminary study ends with a simulated business meeting where the customer decides if he/she should purchase the whole project. The students learn about the challenges of prioritization, convincing a customer, and making commitments within a limited budget and time frame. This addresses (P1), (P2), and (C1).

5) Iteration planning and reviews: In the rest of the project time after the business meeting, the students are directed to use an iterative planning style with fixed iteration lengths.

Both the requirement of having iteration planning (what tasks should be solved in the next iteration?) and iteration review (what was accomplished in the last iteration?) are obligatory in the course framework. The rationale for this approach is to gradually improve the understanding of iterative development and regular customer feedback. This addresses (P1), (P2), (C1), and (C2).

6) Time reporting and project planning: During the whole project, all students have to perform time reporting for each hour that they work on the project. There should be a defined number of hours that each student should work, plus/minus some percentage. The overall project should be planned and monitored with respect to major milestones, deliverables, and activities. The purpose is to learn about planning and working under risk with limited resources, i.e., to learn when work is "good enough". This addresses (P2).

7) Transformation of organizational structure: The students are originally introduced to a classic organizational structure, consisting of several departments, with roles such as department managers, project managers, testers, and developers. However, they are also encouraged to form cross-functional teams, i.e., teams that cut across the departments and include both customer knowledge and technical expertise. The rationale for this gradual transformation of the organization is to have the students learn and reflect upon different organizational and process styles. This addresses (O1) and (P1).

8) Retrospectives and process improvements: Each iteration is followed by a retrospective meeting where the students discuss "what was good", "what went wrong", and "how can we improve it" in the next iteration. This addresses (P1) and (C1).

9) Release planning and expo: At the end of the project, the students present the product at a simulated expo, i.e., they are given the ability to orally present the benefits of their product in a convincing and user-oriented way. This addresses (P1), (P2), and (C1).

10) Self reflection and experience documentation: Finally, the students should write their reflections on their way of working. Both positive and negative findings should be documented and analyzed from both a process and product perspective. This addresses (O1), (P1), (P2), (C1), and (C2).

C. Assessment Tasks

The assessment tasks used for evaluating the performance of the intended learning outcomes are divided into two categories: the *process perspective* and the *product perspective*.

The processes perspective, which is the main focus of evaluation, evaluates how students discover problems and how they solve them. Assessments are inherently mostly performed by observation (e.g., at company meetings) and by interaction (with the CEO, supervisors, and the customer). Another important evaluation instrument is the time report, i.e., when and how much a student worked on the various parts of the project, and their self reflection on their performance. Assessed learning outcomes for the processes perspective are (O1), (P1), (P2), and (C1).

The product perspective concerns both quality and functionality of the actual delivered product from a customer point of view. Moreover, internal artifacts should be evaluated from a clarity and communication perspective. Addressed learning outcomes for the product perspective are (P2) and (C2).

III. Course Design and Implementation

A company approach-based course was designed at Linköping University, Sweden, and implemented in 2009 and 2010. This section gives a short overview of the essential design decisions made in comparison with the course template presented in previous section.

A. Course Overview

Each year the course had approximately 110-120 students, divided into four different companies. Each company was in turn divided into one research and development (R&D) department and one products and sales (P&S) department. The former included roles such as developers and architects, and the latter roles such as analysts, configuration manager, and testers.

The course was given in one semester in both 2009 and 2010, lasting, approximately 15 weeks (part time) in parallel with other courses. Each student was requested to perform 160 hours of full time work, within a certain percentage. In total, the project occupies six ECTS credits, which corresponds to four weeks of full time study, or 160 hours. The project course was given as part of a larger SE course, which includes both laboratory work and software engineering theory. Theory lectures and exercises were primarily given in parallel with the first part of the project course.

B. Students and Curricular Context

The students formed a heterogeneous group. 81% of the students were male and 19% female. Approximately 57% of the students were Swedish, studying an engineering program combining computer science with management and/or media technology. The management students studying their fourth year (which corresponds to the first year of a Master's of Science program) had a sound basis in mathematics, organization theory, economics, as well as fundamental knowledge in computer science. The media technology students had more courses in computer science and software engineering. The objective of both these curricula is to give a broad engineering education, specialized in computer science.

The other 43% of the students were exchange or Master's students studying their first year of a Master's of Science in software engineering and management or in computer science. The former program also includes management courses.

C. Teachers

During the two years, various teachers and teaching assistants (TAs) taught the course. In the first year the first author of this paper and designer of this course played the role of CEO and supervised planning and processes. Two TAs were supervisors, coaching students with aspects such as architecture, requirements elicitation, and testing. In the second year, four TAs were responsible for all aspects, including planning and processes. That year's CEO took a more passive role. One TA taught in both years and the acting customer was the same for both years.

D. Customers

This course uses a role-played but realistic customer. The product that all the companies were developing was a simple and minimalistic *Enterprise Resource Planning (ERP)* system, especially tailored for universities. One of the Department's professors played the customer role, but was in fact playing himself and his use of and need for a new ERP system. The requested ERP system is a basic web-based system with a centralized database. It should handle tasks such as basic economic budget planning, prognoses, and employee management. The student can select the technical platform themselves (e.g., Java, or Ruby on Rails). After the tollgate business meeting, another customer from another department was introduced, with slightly different needs. The main rationale for choosing a role-played customer instead of a real company from industry was the benefit of having control of the whole framework, including requirements.

IV. Evaluation Method

This section describes the research method used to evaluate the methodology of the company approach.

A. Data Collection

The course was evaluated by students with a questionnaire-based survey with closed questions during the course, and a questionnaire-based survey with open questions after the course. Both surveys were self-administered [11].

For the in-course survey, paper questionnaires were handed out after a mandatory project meeting ending ten minutes earlier than scheduled. The answers were collected per company in a closed envelope, and the company IDs were randomly coded afterwards. No identification of the individual student was possible. The questionnaires were collected in December 2009 and December 2010. Each round had four companies, and a total number of 187 questionnaires were collected (83% response rate). The data collected from 2009 and 2010 were analyzed together to compare two instances of the course.

For the post-course survey, the students from year 2009 were contacted by the examiner via LinkedIn. In April 2011 a questionnaire was sent via e-mail with three open questions to the 62 students. 19 answers were received (31%).

B. Analysis Method

The quantitative data (from closed questions) was entered in SPSS (Statistical Package for the Social Sciences), with seven independent variables (round, age, sex, industrial experience, software development experience, curriculum, and result from theory exam), and 18 dependent variables (the questions). Out of the 18 dependent variables, 14 were used for further analysis because the other four variables related to general course evaluation (e.g., connection to the theory part of the course). For each of the questions, the students answered using one of six options (e.g., Strongly Disagree, Disagree, Neutral, Agree, Strongly Agree, and N/A). Hence the data was analyzed as ordinal scale measurements.

In the open question survey, the respondents were self-selected in two steps (LinkedIn and answering the e-mail). Hence, they had a very positive attitude to the course. It is not meaningful to try to test hypotheses from this limited sample, but it is interesting to learn about the major categories of answers from these qualified respondents. To guide the analysis, the *constant comparative method* [12] was applied to discover the commonalities in the answers. The analysis was performed by the second and the third author of the paper, and the respondents were anonymous to them. After the categorization the frequencies of the findings were counted.

The first question *"Are you currently working? If so, in what industry and for how long (in months)? (Do not include the time for master thesis work)"* showed that 11 of the 19 respondents were working in industry for 3-13 months, 6.7 months on average. The second and third questions were answered with texts ranging from 2-38 lines comprising 3300 words in total.

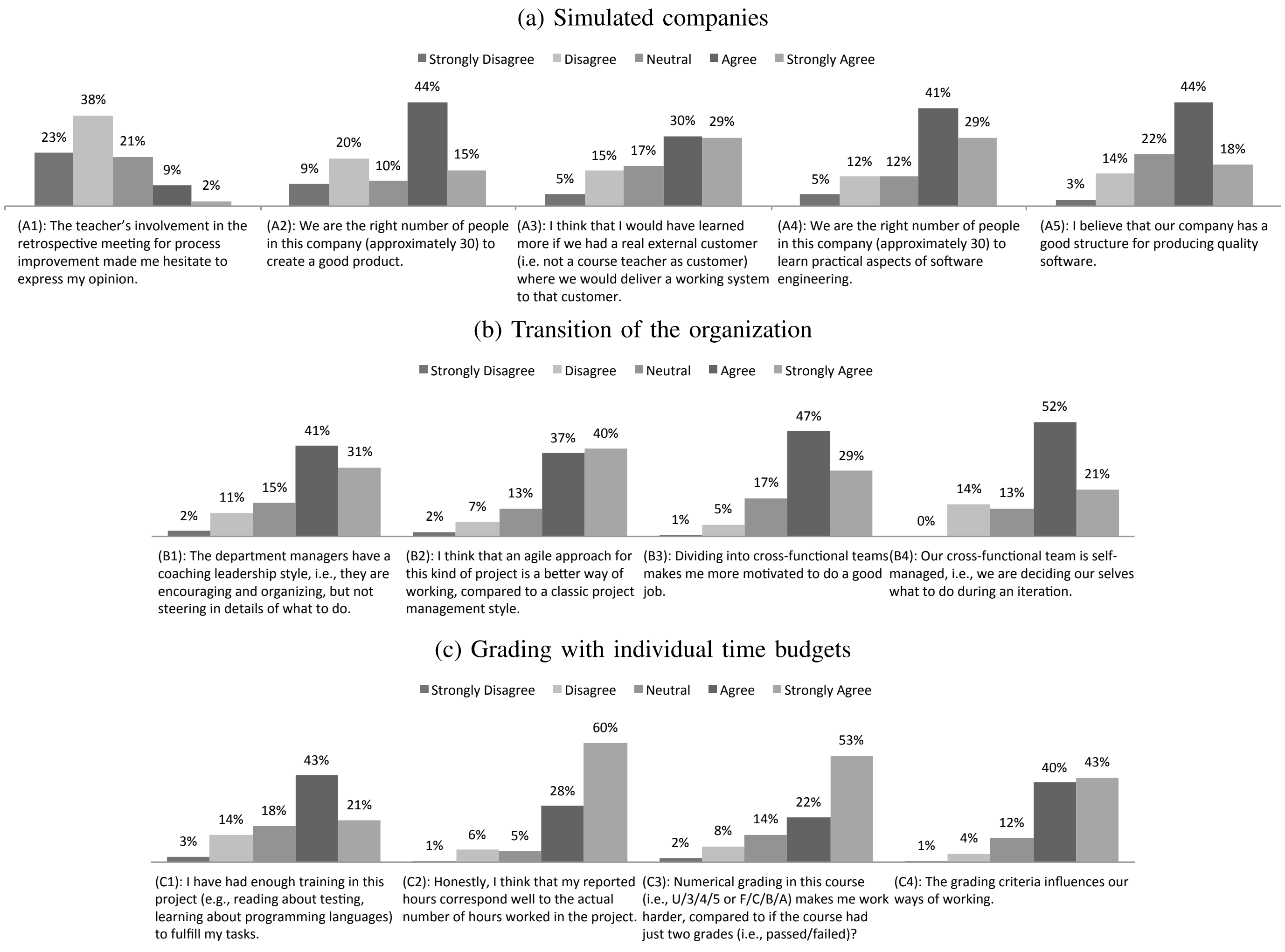
V. Results and Discussion

This section discusses the results of the open survey questions (Table I) as well as the closed question survey (Fig. 1 and Fig. 2). The discussion is divided according to the three components of the company approach: *simulated companies*, *transition of organization*, and *grading with individual time budgets*.

A. Simulated Companies

One of the main differences between a traditional capstone project (e.g., [4]-[6]) and the company approach is that students are not organized in teams, but in *simulated companies* that are organized by the students and consist of several collaborating teams. One natural question is what size should such a simulated company be? Considering Fig. 1a, a majority (70%) agrees or strongly agrees that a 30-person-sized company is good for learning practical SE (see statement A4). Somewhat surprisingly, according to statements (A2) and (A5), the students also agree (approx. 60%) that this size is right for creating good quality products. This may indicate a slight exaggeration by the students regarding the quality of the software they produce. However, because the main focus is the process and not the product (see Section II-C), the survey indicates that the size is regarded as reasonable by the students from a learning perspective.

Another question is if a simulated company approach should include a simulated customer or a real customer from industry. The current course implementation used a simulated (role-played) customer, but (A3) shows that 59% of the students would have preferred a real customer. This fact was also confirmed by a few students in the open survey (Table I). Either approach naturally has its pros and cons. An industry customer can give a more realistic scenario but is on the other



(A1): The teacher's involvement in the retrospective meeting for process improvement made me hesitate to express my opinion.

(A2): We are the right number of people in this company (approximately 30) to create a good product.

(A3): I think that I would have learned more if we had a real external customer (i.e. not a course teacher as customer) where we would deliver a working system to that customer.

(A4): We are the right number of people in this company (approximately 30) to learn practical aspects of software engineering.

(A5): I believe that our company has a good structure for producing quality software.

(B1): The department managers have a coaching leadership style, i.e., they are encouraging and organizing, but not steering in details of what to do.

(B2): I think that an agile approach for this kind of project is a better way of working, compared to a classic project management style.

(B3): Dividing into cross-functional teams makes me more motivated to do a good job.

(B4): Our cross-functional team is self-managed, i.e., we are deciding our selves what to do during an iteration.

(C1): I have had enough training in this project (e.g., reading about testing, learning about programming languages) to fulfill my tasks.

(C2): Honestly, I think that my reported hours correspond well to the actual number of hours worked in the project.

(C3): Numerical grading in this course (i.e., U/3/4/5 or F/C/B/A) makes me work harder, compared to if the course had just two grades (i.e., passed/failed)?

(C4): The grading criteria influences our ways of working.

Fig. 1. Results of the quantitative study where the diagrams are categorized according to the key areas of the company approach. Missing answers and N/A answers are for clarity omitted from the presentation.

TABLE I
MAJOR CATEGORIES OF THE ANSWERS TO THE OPEN QUESTION SURVEY. NUMBERS WITHIN PARENTHESIS AFTER THE STATEMENTS SHOW HOW MANY RESPONDENTS (OUT OF 19) THAT PROVIDED THE STATEMENT.

	Q2: Which were the most important things that you learned during the project course that are of value for you now in your professional life?	Q3: If you would have been the course leader for the course next year - what would you then have done differently in the course?
Individual (role)	Useful technical knowledge from my role (10), Time management (8), Good overview of a software project life-cycle (5)	Clearer instructions for roles needed for a faster start and less frustration (3)
Team	Agile methods useful (8), Team communication experience (6), Preferred way of working (8)	Do not change anything (8)
Company	Collaboration experience (9), Stretched planning horizon (2), Improved ability to make effort estimation (3)	More prepared organization at the start (5), Avoid grading-based organization (1), Frustration at low progress (1), Uneven technical skills of students (2), Real customer wanted (2)

hand less predictable from a course perspective. A simulated customer can be part of pre-defined problem scenarios (e.g., new unexpected requirements or customers during the project), which might be hard to realize in an industry setting. Using industry customers together with the company approach can be considered as interesting future work for evaluation.

An interesting question regarding this approach is how relevant the experience is, compared to working in industry. Posing such a question to a student with no working experience is of course less meaningful. However, as shown in Fig. 2, 47 students (26% of the students answering the question) state that they have more than six months of full time work experience. None of these students disagree about the industry relevance of the approach. Table I also shows that learning outcomes at the company level are useful later for graduated students who are working in industry. However, it is too early to draw any general conclusions, before having evaluated this approach at several universities, in different countries, and different curricula.

B. Transition of the Organization

The main idea of having a transition of the organization from line organization to an agile organization is to emphasize *active learning* where the students construct their own knowledge (constructivism [9]). The hypothesis is that the student then gets a deep understanding, thus avoiding having a surface approach [13] to learning. Fig. 1b shows the result of four statements concerning this transition. It can be noted that approximately 70% of the students agree or strongly agree with all four statements. Because this is a self-assessment, the evaluation does not show how much of the transition the students actually understand, but it clearly indicates that they understand the benefits of agile approaches with iterations and self-organization.

Table I also indicates certain problems with this approach. One concern expressed by the students is the frustration at the beginning of the project when they do not know what to do or what their roles mean. One suggestion for improvement is clarification of the role definitions at the beginning of the project. However, the main criticism does not concern the approach of having a transition, but rather that of how to start-up the project from the beginning. This frustration of slow progress (especially in the beginning) can also relate to the uneven technical skills that the students have in the project (note the heterogeneous aspect of the course implementation).

On the whole students are more satisfied working in cross-functional teams, which might be due to maturation, but more probably from the sense of producing quality products for the customer, which is a more concrete activity.

C. Grading with Individual Time Budgets

Assessing and grading project-based courses are difficult. It is clearly shown in the result of statement (C3) in Fig. 1c that numerical grading is very important for these students regarding work effort. Hence, the choice of how the assessment is performed also has a high impact on the learning outcome. Moreover, 83% (C4) of the students state that the grading criteria influenced their way of working.

One question concerning grading and assessment is whether grades should be group-based, individual, or a combination [14]. All these variants have pros and cons. For example, a group-based grade would probably make the students focus on the same goal instead of focusing on their own individual assessment. On the other hand, there is a risk that some students can hide in the group and get a good grade without having given a good individual performance. This grading problem was also mentioned by the students performing the open question survey (Table I).

This dilemma was addressed during the evaluation in 2009 and 2010 by giving out one grade for each company. Then, depending on individual students performance, the individual grade could either be higher or lower than the company grade. It should also be noted that the software project is only one part of the SE course, which also includes SE theory and laboratory exercises. The final individual grading problem was then solved by combining the project grade with the individual SE theory grade.

However, this is not the ideal approach, and further alternatives must be evaluated. For the current course (2011) a new concept is being tested where students also write individual reports during the project. These reports include: (1) what the student claims he/she has contributed with in the project, and (2) what are the most important things that he/she learned during the project. The aim is that such self-assessment can be a tool for the examiner when assessing individual students, which mitigates the problem that some students underperform. However, this approach of individual assessments has not yet been evaluated and is considered as future work.

A particular component of the company approach assumed to help with the individual assessment is the individual time budgets and time reporting. The key idea of using constrained time budgets is to achieve the learning outcome that student realize that resources are limited and therefore prioritization is necessary. Also, the individual time reports form a good foundation for assessing how individual students have performed and contributed during the project.

A potential problem is that students are not honest about their time reporting. Surprisingly, the statement (C2) that their reported hours correspond to their real work was the strongest agreement of the whole survey (88%). This gives some indication that individual time reporting is a good tool for individual assessment that can potentially be extended with further detailed reporting.

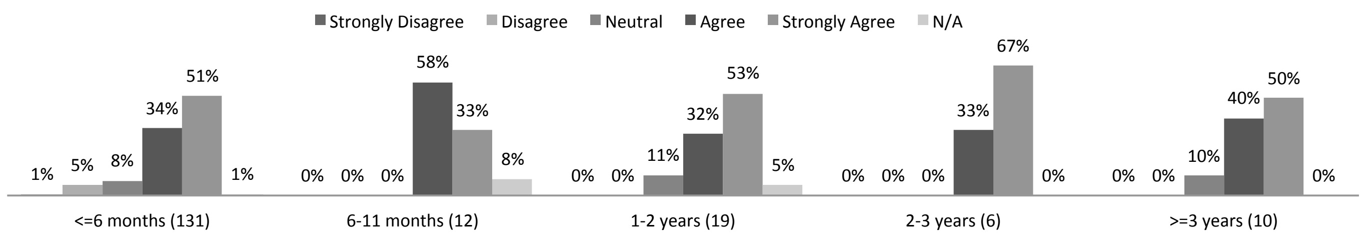


Fig. 2. Percent of agreement to the survey question *"I believe that the company approach in this course (i.e., that we are organized as a simulated company) makes our experience industry relevant"*. Each of the diagrams shows the percentage of agreement in relation to the number of years of full time experience within software industry or related industry that the students had taking the course. From the total number of questionnaires, nine students did not answer this question or did not fill in their working experience. The numbers within parentheses state the number of students within each category.

D. Threats of Validity

Some of the questions in the survey ask the students to rate the utility and relevance of the course. This can be a problem when a student with no experience has to rely on second-hand information to make a proper assessment. In Fig. 2 this is illustrated by showing the answers from students with varying degrees of experience to the question about industry relevance. The diagram shows that 47 students have more than six months experience, which is a large minority of 26%. In addition, the diagram displays that the distribution over agreement/disagreement is similar over all groups of students.

The students in the post-course survey are self-selected and not representative. This was taken into account in the analysis which evolves the major categories and the theory from what was obtained, rather than trying to numerically verify any hypothesis. The third question explicitly asks for improvement suggestions, which brought some negative experiences to the table. On the other hand, it is hard to imagine a follow-up survey without having self-selected respondents and the response rate is comparable to other surveys performed by the university with an industrial target group.

VI. Related Work

An overview of related work is given in Table II. The closest related work is a contemporary report by Meawad [15] who describes a project course with simulated companies. Students work in groups of 25-26 students, using an agile methodology (Scrum/XP), and have a competition between companies. The major differences to the company approach are that there is no transformation of organizational structure and time reporting is not used. Blake [16] simulates a real company with roles, and organizes a transition between organization structures by splitting the topic into two courses.The development method is iterative and the companies are built up of sub-teams. The company size is smaller (11-18 students) than the company approach and there are no requirements on time reporting. Coppit and Haddox-Schatz [17] also describe a software engineering project with simulated companies where approximately 30 students work together. Similarly to the company approach students are using an agile method. Even though students work together, grading is based on individual achievements. The differences are that there is neither a transformation of the organization nor any requirements on time reporting. Chaczko *et. al.* [18] describes an approach using simulated companies with component teams of five to seven students, but there is no figure given for the size of the companies. They are using the Spiral model, that is mostly iterative, and teams are self-organized as in the company approach. Another similarity is that the teams have requirements on managing deadlines and time budgets. The major difference is the lack of transition of the organization structure. Razmov [19] describes pedagogic principles similar to this article, for instance, students own their decisions and receive prompt feed-back over iterations. Practically there are several differences to the company approach. Jing *et al.* [20] describes instructor-led projects. The similarity to the company approach is the strict time schedule and the fact that the organization continuously changes depending on the learning outcome. Otherwise the course has very little in common with the company approach.

As can be seen in Table II the project courses have many commonalities with problem-based learning [21], especially when performed as a group activity in a constructivism approach. In practice, a project course needs more guidance in technology and processes than is normally accepted in problem-based learning. In all the work reviewed the authors emphasize the need for frequent feed-back to the groups both in order to support learning and in order to ensure a reasonable progress of the projects.

Related work is also reported on the design of successful capstone projects where the students practice and apply their engineering skills in a real-world settings [3]-[5], [22]. These courses have the role of summarizing a curriculum rather than infusing new knowledge. Broman [23] suggests an alternative to capstone courses where students from different years in a curricula are working together on one project. This approach is based on the ideas of the company approach, but extended to be used in a whole curricula. Other ways of preparing for a professional career are internships and on-the-job training. The idea of using competition as part of a course has also been used in other contexts [24], [25].

VII. Conclusions

This paper describes a new methodology for running software engineering project courses, called the company approach. The three key components of this approach are *simulated*

TABLE II
Comparison between the company approach and related work.

	Company approach	Meawad [15]	Blake [16]	Coppit and Haddox-Schatz [17]	Chaczko *et al.* [18]	Razmov [19]	Jing *et al.* [20]
Simulated company	Yes	Yes	Implicitly	Yes	Implicitly	Unkonwn	No
Company size	30	25	11-18	30	5-7 (per team)	6-8	Unknown
Transition of organization structure	Yes	No	Yes, be-tween two courses	No	No	No	Yes, in learning steps
Development method	Iterative/S CRUM	XP/SCRUM	Iterative	XP	Spiral model	Incremental	Incremental
Time budget/ reports	Individual Examined	No	No	No	Yes	No	Detailed schedule
PBL	Used	Used	Used	Some	Leading star	Leading star	Used
Competition	Yes	Yes	No	No	No	No	No

companies, *transition of organization*, and *grading with individual time budgets*. The approach has been evaluated using questionnaire-based surveys, with the following main findings.

A large majority of the students believe that the approach of having a simulated company makes their experience industry relevant. The same pattern is also observed when analyzing the answers from a large minority of the students who have more than six months of full time work experience (26%). The size of the simulated company (30 students) was, by the majority, considered to be good for learning about practical aspects of software engineering. In the current course implementation, a simulated (role-played) customer was used. However, according to the evaluation, a majority of the students believe that they had learned more if the customer would have been a real external customer. Having an external customer increases the risk associated with the outcome the project and it is therefore important that whoever is responsible for the course has a very close collaboration with such a customer, and regularly follows up the progress of the customer interaction. Evaluation of such an extended approach is considered as future work.

The main findings regarding the transition of the organization from a line organization to an agile organization were generally positive. The students express a certain frustration at the beginning of the project, especially when defining and understanding roles. However, the evaluation shows that an agile approach with cross-functional teams both makes students more motivated to do a good job and proves to be a better way of working than that of a line organization with project management.

The evaluation results indicated that numeric grading was considered to be very important as an incentive to work hard in the course. Grading criteria also strongly influenced their way of working. A potential problem with the current implementation is that grades are mainly group based, i.e., one grade for each company. A suggested approach, that can be used for future evaluations, is to extended the current individual time report with another individual self-assessment report. The hypothesis is that combining company grades with individual grades would provide a more fine-grained grading that would still maintain the incentive for the students to work together as one simulated company.

ACKNOWLEDGMENT

The authors would like to thank all the students who participated in the software engineering course during years 2009 and 2010. This research was funded by the Department of Computer and Information Science, Linköping University, Sweden.

REFERENCES

[1] Interim Review Task Force, "Computer Science Curriculum 2008 - An Interim Revision of CS 2001," 2008.

[2] Joint Task Force on Computing Curricula, "Software Engineering 2004 - Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," 2004.

[3] R. H. Todd, S. P. Magleby, C. D. Sorensen, B. R. Swan, and D. K. Anthony, "A survey of capstone engineering courses in north america," *Journal of Engineering Education*, vol. 84, pp. 165–174, 1995.

[4] A. Goold, "Providing process for projects in capstone courses," in *Proceedings of the 8th annual conference on Innovation and technology in computer science education*, ser. ITiCSE '03. New York, USA: ACM Press, 2003, pp. 26–29.

[5] S. Karunasekera and K. Bedse, "Preparing Software Engineering Graduates for an Industry Career," in *Proceedings of the 20th Conference on Software Engineering Education and Training*. IEEE Press, 2007, pp. 97–106.

[6] D. A. Umphress, D. Hendrix, and J. H. Cross, "Software process in the classroom: the capstone project experience," *IEEE Software*, vol. 19, no. 5, pp. 78–85, 2002.

[7] J. Biggs, "Enhancing teaching through constructive alignment," *Higher Education*, vol. 32, no. 3, pp. 347–364, 1996.

[8] J. Biggs and C. Tang, *Teaching for Quality Learning at University*, 3rd ed. Open University Press, 2007.

[9] M. Ben-Ari, "Constructivism in computer science education," *Journal of Computers in Mathematics and Science Teaching*, vol. 20, no. 1, pp. 45–71, 2001.

[10] K. Schwaber and M. Beedle, *Agile Software Development with Scrum*. Prentice Hall, 2001.

[11] C. Robson, *Real world research: a resource for social scientists and practitioner-researchers*, ser. Regional Surveys of the World Series. Blackwell Publishers, 2002.

[12] B. Glaser and A. Strauss, *The discovery of grounded theory: strategies for qualitative research*. Chicago: Aldine, 1967.

[13] F. Marton and R. Säljö, "On qualitative differnces in learning: I - outcome and process," *British Journal of Educational Psychology*, vol. 46, pp. 4–11, 1976.

[14] M. Lejk and M. Wyvill, "A survey of methods of deriving individual grades from group assessments," *Assessment & Evaluation in Higher Education*, vol. 21, no. 3, pp. 267–280, 1996.

[15] F. Meawad, "The Virtual Agile Enterprise: Making the Most of a Software Engineering Course," in *Proceedings of the 24th IEEE Conference on Software Engineering Education and Training*, 2011, pp. 324–332.

[16] M. Blake, "A student-enacted simulation approach to software engineering education," *IEEE Transactions on Education*, vol. 46, no. 1, pp. 124–132, Feb 2003.

[17] D. Coppit and J. M. Haddox-Schatz, "Large team projects in software engineering courses," in *Proceedings of the 36th SIGCSE technical symposium on Computer science education*, ser. SIGCSE '05. New York, USA: ACM Press, 2005, pp. 137–141.

[18] Z. Chaczko, D. Davis, and V. Mahadevan, "New perspectives on teaching and learning software systems development in large groups," in *Information Technology Based Higher Education and Training, 2004. ITHET 2004. Proceedings of the Fifth International Conference on*, 2004, pp. 409–414.

[19] V. Razmov, "Effective pedagogical principles and practices in teaching software engineering through projects," in *Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, 2007. FIE '07. 37th Annual*, Oct 2007.

[20] L. Jing, Z. Cheng, J. Wang, and Y. Zhou, "A spiral step-by-step educational method for cultivating competent embedded system engineers to meet industry demands," *Education, IEEE Transactions on*, vol. 54, no. 3, pp. 356 –365, Aug 2011.

[21] D. F. Wood, "Abc of learning and teaching in medicine problem based learning," *British Medical Journal*, vol. 326, no. 7384, pp. 328–330, Feb 2003.

[22] T. Clear, M. Goldweber, F. H. Young, P. M. Leidig, and K. Scott, "Resources for instructors of capstone courses in computing," in *Working group reports from ITiCSE on Innovation and technology in computer science education*, ser. ITiCSE-WGR '01. New York, USA: ACM Press, 2001, pp. 93–113.

[23] D. Broman, "Should Software Engineering Projects be the Backbone or the Tail of Computing Curricula?" in *Proceedings of the 23th IEEE Conference on Software Engineering Education and Training*, Pittsburgh, USA, 2010, pp. 153–156.

[24] R. Lawrence, "Teaching Data Structures Using Competitive Games," *IEEE Transaction on Education*, vol. 47, no. 4, pp. 459 – 466, Nov 2004.

[25] L. M. Regueras, E. Verd, M. J. Verd, and J. P. de Castro, "Design of a Competitive and Collaborative Learning Strategy in a Communication Networks Course," *IEEE Transaction on Education*, vol. 54, no. 2, pp. 302 – 307, 2011.