Academic Education of Software Engineering Practices: Towards Planning and Improving Capstone Courses Based upon Intensive Coaching and Team Routines

Christoph Johann Stettina\textsuperscript{1,2}, Zhao Zhou\textsuperscript{2}, Thomas Bäck\textsuperscript{2}, and Bernhard Katzy\textsuperscript{2}

\textsuperscript{1}Centre for Innovation, Leiden University, Schouwburgstraat 2, 2511 VA The Hague, NL
\textsuperscript{2}LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, NL

Abstract

Academic education of professional processes is challenged by a necessary balance of practical activities with academic reflection. In this paper we address this issue by discussing our experiences with teaching software engineering practices and their continuous improvement. By designing a graduate course we embed an intensive coaching routine based upon agile practices with research activities to leverage knowledge of students and coaches. As a concrete example of an embedded research project we conduct an experiment on the impact of two different meeting routines on the teams satisfaction with information exchange. Our results show that the intensive coaching in individual teams is shorter in nature and more appealing to the students. Our findings suggest that software engineering education can benefit from the notion of team routines and process improvement practices contributing to maturity of students and educators.

1. Introduction

One of the biggest challenges in academic education of professional processes like software engineering is to balance students’ practical education with academic reflection. Software engineering graduates on one hand need to get prepared for managing the engineering process. On the other hand, they are academic graduates who need to be trained in reflection and research activities. Research suggests that agile methods stimulate learning and reflection [22]. As such they are not only software engineering practices but can be applied in education as well. They have attracted educators and coaches delivering positive results in industrial [21, 16, 15] and academic settings [10, 11]. Agile practices are concrete team-level routines such as frequent customer feedback loops, iterative delivery of intermediate results, and stand-up team coordination meetings. These routines further knowledge creation and faster feedback loops on organizational level. Such procedural knowledge is tacit [26, 14] and therefore cannot readily be acquired through simple methods such as lectures or books. Agile routines, in professional practice as well as in academic education, thus need be trained through executing the process [26]. In order to be able to transfer routines educators and coaches thus themselves need to have collected sufficient experience executing the steps. This is equally true for software engineering researchers, who need experience with agile methods to design and execute meaningful studies.

In this paper, we follow the call from literature [6, 27] to advance academic teaching in software engineering by discussing our approach to continuous planning and improvement of software
development capstone courses. Based upon intensive coaching and the notion of team routines we create a research-roadmap which enables professional education of students alongside with academic reflection. As a specific example we discuss a graduate course on *System Development and Project Management*. There, 30 students in 6 teams are part of an experiment while turning a created idea into a working demonstrator using agile practices. The paper aims at helping academic educators in their creation of research-based software engineering courses for agile methods. To software project managers and coaches in large organizations it provides understanding of the impact of inter-team stand-up meetings on team satisfaction and coaching success.

2. Background and Related Work

Learning and knowledge creation are closely tied to human action [14]. Literature on learning theories suggests that students progress through two major stages during the development of a cognitive skill, a declarative knowledge stage and a procedural knowledge stage [26, 1]. While the declarative knowledge (the knowing of what), can be acquired from traditional approaches such as lectures and text books, procedural knowledge (the knowing of how) is to be trained on the process [26]. From text books we use declarative knowledge about agile methods. Agile practices [20, 29] like stand-up meetings (e.g. time boxed, frequent or daily team meetings providing a status update), iteration reviews (e.g. demonstrations at the end of each iteration/sprint to present working functionality) or pair programming (e.g. two programmers writing and reviewing source code at the same workstation) are suitable to create procedural knowledge. They stimulate direct communication and learning through the repeated, iterative interaction of the participating individuals. Agile practices are organizational routines [19] that support both stages of the learning process and provide a comprehensible understanding of software engineering practices in a single teachable framework [6]. Industrial practice knows agile coaches, who guide teams through their projects and the iterative steps so that teams improve their agile practice while undertaking the development process [21, 16, 15]. In the industrial process like in the class room especially stand-up meetings and iteration reviews are useful.

Agile practices contribute a number of benefits to course programs and literature especially describes this for basic under-graduate capstone courses [6]. Agile practices contribute to professional routines from several domains such as project management [28, 18] and human-computer interaction [12] and have been applied in combination with supportive ICT environments [2]. Capstone course projects are often preferred as they provide a good opportunity for students to combine multidisciplinary knowledge acquisition through a coaching routine [5, 17]. However, such courses are no vocational training but include academic reflection.

There are few contributions on how to balance coaching routines with academic activities and how to enable their continuous improvement in education [4]. Critics have pointed out that the weak influence of research findings on academic educational practice is largely caused by 1) the low status of educational research, which is hampered by a lack of strict methodological rigor, frameworks, and norms [27], and 2) the fact that the produced knowledge is not applied in the field, because universities do not emphasize educational research [27]. In conclusion some call for a more evidence-based education where knowledge produced can move in either direction [24] with a greater involvement in the field based upon multidisciplinary and team-based work [27]. Agile methods provide access to a large research base through the fields of software process improvement and empirical software engineering with both rigorous methods evaluated in practice and large and active research communities. As such they could provide a stable theoretical ground for reflection on team coaching routines in education.
3. Objectives

Tierney and Holley [27] argue that educational research should be an interdisciplinary study focussing on solving problems and team orientation to move away from silo orientation of separate academic disciplines. Agile methods, as one possible approach, provide an interdisciplinary framework to address software engineering related problems at both learning stages [26, 1] while providing opportunities for evidence based reflection upon process improvement [22]. Considering the earlier work, we would like to explore our possibilities by asking the following question: *How can we plan software engineering courses so that using agile process improvement techniques we can improve education and contribute to research at the same time?*

To further our understanding and provide one concrete example of integrating software engineering practice and research in education we have embedded a quasi-experiment based on an agile team routine in our course design. Specially we look at stand-up meetings [20] which are integral to coaching in agile methods. Within agile capstone courses an extended coaching success has been reported [10], however, connected to a higher workload for students and instructors [10, 4]. Due to their iterative nature recurring stand-up meetings represent a big part of the workload. How can we justify this additional effort? There is little in-depth research on concrete team-level coaching routines and as a concrete example of a research activity within our course we thus would pose the following question: *What are the implications of individual intra-team stand-up meetings on coaching success and team satisfaction compared to bigger inter-team stand-up meetings?*

4. Study Context

The context of the study is the *System Development and Project Management* course with the goal to prepare students for the multidisciplinary challenges of ICT projects. Following our research questions we designed a 6 weeks practical assignment for the interdisciplinary course integrating project and stakeholder management and requirements engineering techniques with an agile approach. While the regular course lectures provide the necessary background and theoretical (declarative [26]) knowledge, the practical sessions aim to build up the students’ procedural knowledge. To do so the students have to develop a project from an initial project bid towards a working demonstrator and present it at a trade fair. The project is especially focussing at a project’s Front-End activities thus those when a project team is not entirely in control of the scope yet and ideas still need to be strengthened within an organization. To stimulate the learning progress of the students we employ two agile practices: Stand-up meetings and iteration reviews with customers [20, 29]. Stand-up in agile software development [20, 25] are daily team meetings providing a status update to team members. It facilitates information exchange among on potential challenges and enables coordination inside the team. The meetings are generally hold standing and are timeboxed to 5-15 minutes to frame its short and focussed nature. Each coaching session starts with a team stand-up where each group was asked the three common questions: *“What have you done since the last meeting?”*, *“What are you planning on doing until the next meeting”* and *“What issues and impediments are you facing that prevent you from accomplishing these things?”*. Iteration reviews/demonstrations are applied to involve the customer in the development process and gives the customer a structured way to steer the product development. In this course we use the reviews to advance the students’ learning outside the ”agile sweet spot” [7] as the scope of the projects needs to be worked out by the students. Due to the focus on the human aspects of software development, we pay particular attention on team building, teamwork and informal communications.
5. Method

In this paper we want to contribute to the understanding of the complex topic of planning and embedding experiments based on agile team routines into capstone courses, to improve education and contribute to research at the same time. As this research is connected to a variety of factors such as process, coordination, teamwork and perceptions in a social context we performed a single-case study [30] combining qualitative and quantitative elements. This allows us to explore the complex problem while developing rich and informative conclusions for further research.

5.1. Data Collection and Analysis

Agile methods put an emphasis on self-managing teams and the integration of individuals into the software development process [22]. We thus made use of individual perceptions of team members and linked qualitative process descriptions with quantitative questionnaires and the artifacts developed in course of the project. Questionnaires would allow us to collect the individual perceptions in a measurable manner while enabling the participants to state their opinions in an anonymous way. To create a deeper picture supporting our research we further used data from informal interviews, ethnographical notes, observations and analyzed the delivered artifacts. Observations and informal interviews were conducted by the first author during the coaching sessions every iteration. The coaching notes were used similarly to diaries [3], to track the implementation progress and the challenges of the students. This approach as presented earlier [23], enables to capture the development of selected perceptions of individuals and team throughout the entire project and it allows the comparison of these perceptions to the project outcomes.

To address our second research question embedded into the course, the 30 attending students formed 6 teams and were divided into two main groups: SUnited and SIndividual. While the 3 teams belonging to group SUnited would take part in the weekly stand-up meetings altogether, the 3 other teams belonging to the group SIndividual would take part in individual team stand-up meetings. This setup provides a suitable environment allowing the comparison of several teams working on the same project. It allows the analysis of patterns of action emerging in course of the experiment and their implications on the project results. The rationale for choosing intra and inter-team meetings for the experiment was that we wanted to understand the implications of the two different meeting routines on the information exchange and the emerging ideas among the participating teams. To do so we applied two types of questionnaires: First, a short questionnaire for a longitudinal collection of the individual team member’s satisfaction with the project, the amount of work and the documentation was used. This part of the questionnaire was collected every week during the project. The approach has been applied by us in a similar study setting based upon an undergraduate course [23] and is similar to the project satisfaction graph as presented by Moe [13]. The questions were: How satisfied are you with the project?, How satisfied are you with the teamwork in your team? and How satisfied are you with the information exchange in this project? Second, longer questionnaires were applied in the course of the development phase, they were used to collect perceptions on the usefulness of specific artifacts, the distribution of roles within the groups. The questionnaires also gave students the opportunity to anonymously comment on their projects. Sample questions can be found here below: How useful did you find the stand-up meetings? How useful did you find writing meeting minutes for the weekly stand-ups? Would you prefer to have stand-up meetings in bigger or in smaller groups?

We chose to administer the questionnaires during the coaching sessions on paper as the response rate was expected to be higher if compared to questionnaires administered online after the class. A consistent response rate was important for the validity of our data. The forms were anonymous
and we re-iterated to the students that the collected data was part of the research project and would by no means affect their grades.

5.2. Bias and Limitations

Several factors limit the generalizability of this study. Although our experience is based upon multiple iterations of capstone courses the here discussed data stems from a single course undertaken in spring 2012, within one university setting with a limited number of teams and students. To counter this we collected the participant’s perceptions over time and calculated the statistical significance. The extent to which graduate, undergraduate and doctoral students are representative of professional software developers and the threat of interaction of selection and treatment respectively, have been addressed in various other studies (e.g. [8]).

6. Results

In this section we will proceed to describe our findings according to the two research questions.

6.1. Course Execution

The course project started with an introductory session on February 2, 2012 and lasted until March 15. During the six week of the course, the students attended the class consisting of lectures on two consecutive days each week. The practical sessions in which the stakeholders were introduced and the teams were coached, took place once a week right after the lecture. The lectures were given by the third author and covered topics on software project management and the related technical, economic and organizational issues. At the end of the course a trade fair event represented the finish of the project with about 50 internal and external participants.

6.2. Research Experiment: The Stand-up Meetings

The theme of the embedded research project has been discussed with fellow colleagues and chosen according to state-of-the-art, applicability and contribution to research and practice. Questionnaire collection for the embedded experiment began together with the stand-up sessions during the second week and lasted throughout the entire project. The meetings were timeboxed to 15 minutes for the separated teams (SIndividual) and to 30 minutes within the big group (SUnited). The session times were adjusted in course of the project (originally 10 minutes and 40 minutes).

The initial observation was that it always took a little longer for the bigger group to gather. This remained constant till the end of the project and while the stand-ups for the separated groups seemed to become sharper in time, with more team members being on time as the time was scarce, the larger group was shrinking. At the last meeting while team A was complete, teams B and C of the bigger group was only partially present. This development is further confirmed by the satisfaction graph in Fig. 1a: While the teams belonging to SUnited are less satisfied the teams of SIndividual are more satisfied with their information exchange. In line with the observations we can see in our survey results that the teams prefer stand-up meetings in smaller groups and that the teams that participated in separated stand-ups, found the meetings more useful (Fig.1b).

To test the statistical significance of our data we conducted an independent-samples t-test to compare the satisfaction levels in the SIndividual and the SUnited groups. A P-value below 0.05 is commonly considered statistically significant, while a value of 0.05 or bigger indicates no difference between the groups. There was a significant difference in the satisfaction with information exchange of the SIndividual (M=4.42, SD=1.52) and SUnited (M=5.15, SD=1.28) groups
There was further a significant difference in the satisfaction with the project \( (t(112)=2.7365, p=0.0036) \) and innovativeness perceived \( (t(112)=3.3109, p=0.00125) \) and workload \( (t(112)=1.8397, p=0.0685) \). We did not find a significant difference in the satisfaction with teamwork \( (t(112)=2.0939, p=0.0385) \). These results suggest that the applied stand-up routine does have an effect on the perceived satisfaction with information exchange and innovativeness inside the teams. Specifically, our results suggest that in the individual stand-ups the team members are more satisfied with the exchanged information.

Most of the teams had developed (consciously or not) a member acting as a spokesman. This became visible early on and while the smaller groups seemed to invite more team members to participate within the discussion, in the bigger setting it was rather single and interested team members taking part in the discussion while remainder stayed quite or was distracted. This seemed partially caused by the impatience of the team members caused by the similar project backgrounds and the same repeating issues and progress in course of the project. The teams provided short answers as they almost seemed pressured waiting for the next group follow in the stand-up.

![Graph](image)

**Figure 1:** Satisfaction with information exchange and stand-up meetings. [Scale: -3=Completely dissatisfied, -2=Mostly dissatisfied, -1=Somewhat dissatisfied, 0=Neither satisfied or dissatisfied, 1=Somewhat satisfied, 2=Mostly satisfied, 3=Completely satisfied]

7. Discussion

In this section, we discuss our findings according to the two guiding research questions. We will first begin discussing the embedded research question and the experiment, and then proceed to discuss our broader objective and the experiences with planning and improving in our course.

7.1. How Can We Plan Software Engineering Courses so that Using Agile Process Improvement Techniques We Can Improve Education and Contribute to Research at the Same Time?

The outcome of the here embedded particular research experiment is that the next iteration of the class will be held in separate coaching sessions for each group with an iteration length of one week. The outcome of this particular course iteration is that the teaching staff can draw upon these findings improving the course during the next iteration and contributing to research. While coaching of 6 teams is education, it is a research setting through careful design of a study about the processes applied by the teams.
Fig. 2 depicts the feedback cycle based upon the course design. The initial research backlog, the accumulation of possible research items waiting to be done over time, is compiled from current state-of-the-art (e.g. from a literature study on team routines in SE). During course preparation the university staff or coaching team agrees upon a piece of research, a particular experiment, to be executed within the course. Possible items for such a research backlog would be: application of new technology in teams (e.g. new collaborative solutions such as Google Docs or Dropbox) or testing new team based technologies in context (e.g. pair programming or pomodoro techniques). During the course the students teams are coached and data for the experiment is collected. After course completion a retrospective is being held together with the students, data is evaluated and a report being written. Then the circle begins anew moving to the next research backlog item with the next iteration of the course.

The attention on intensive coaching in our study is driven by the need of students with little experience to acquire professional knowledge from multiple domains which can only be trained on the process [26]. Literature suggests that the understanding of such everyday action is necessary to understand capabilities of teams and organizations and that their study is particularly difficult [19]. Without explicit practice it is almost impossible to study agile routines. How are they established or learned? And how do they evolve in teams? Our quasi-experiment provides a good example on how software engineering routines as patterns of human action evolve in teams over time.

Our quasi-experiment provides a good example on how software engineering routines as patterns of human action evolve in teams over time. In that sense we present our case for software engineering education based upon observable and concrete patterns of action performed by the participants. Agile routines such as stand-up meetings are observable micro-activities and produce measurable outcomes to enable process improvement. Our course design is driven by its consideration of use in educational environments as well as it aides the fundamental understanding of routines in practice. Following the use-inspired basic science of the Pasteurs Quadrant [24], the workload of the training routine in our class is thus not only justified by its contribution to the maturity of the students, but also by its contribution to research, thus to the maturity of the tutors.

Crucial for such a course design is the transparency towards the students and sharing of learning outcomes while staying consistent with academic principles to avoid doubts among students. It is important to reiterate that such continuous learning is important in their everyday work life and that they should be aware of teamwork aspects. It is necessary is to pay attention to sufficient training of the coaches in this case, especially if new coaches are added to the team. The course design requires a careful preparation and discussion of the chosen research theme, however, contributes significantly to practical experience to research and coaching of the involved coaching staff.
7.2. What are the Implications of Intra-team Stand-up Meetings on Coaching Success and Team Satisfaction Compared to Inter-team Stand-up Meetings?

In our experiment the advantage of possible knowledge gain among the SUnited teams compared to the individual meetings of the Sindividual teams was overridden by the decreasing satisfaction in SUnited. Information had to be repeated by the coach during the individual sessions, however, this was justified by the advantages such as the higher attention level in the more personal individual meetings. An interesting observation was that during the 3rd stand-up session members of team B, while hearing of a specific need of another team, offered to implement a specific plugin. "if you pay us..", said a team member jokingly. There certainly has been an exchange of students outside the team boundaries, this however, remained the only visible example among the student teams.

During the stand-up meetings we found that the teams in SUnited provided shorter answers with little discussion if compared to the teams in Sindividual. This could be caused by the fact that the teams with similar goals, issues and challenges the teams felt under pressure knowing that in there is a next group is waiting. This emphasizes that the teams should feel comfortable for a good exchange and discussion. While applying the routine to coach student teams the coach thus needs to establish a relationship of trust emphasizing on his facilitating role. Establishing such a connection is much easier in the individual stand-ups, and allows deeper discussions. Further, the short meeting notes proved to be very efficient to follow up on each team’s progress, especially while coaching the six teams in parallel. However, by applying the described routine coaches by all means should avoid mechanically asking the three stand-up questions. Rather the team members should be encouraged to reflect on their planning, coordination and teamwork practices.

As inter-team stand-ups do not work in our experiment, how can we stimulate direct communication among teams on a frequent basis? In large organizations applying agile methods the distribution of inter-team information is often implemented through "Scrum of Scrums" [9], a type of stand-up meetings among ScrumMasters or other designated team members. Although a team lead, a committed spokesman, naturally emerged in all of the teams, it should be further researched how such a meeting could be introduced in a class setting. Maintaining an optimal learning experience for all students while omitting information asymmetries is important to consider.

Similarly to earlier reports [20] we have observed that development impediments could be identified and addressed early on during the project. Especially as the projects building on existing infrastructure, the stand-ups improved the communication in terms of communicating technical specifications and known limitations of the system. They proved very efficient to intercept communication gaps between the project and the stakeholders.

The iteration length of one week proved to be appropriate. According to our experience an iteration length of one week seems reasonable for capstone courses lasting up to three months, especially at the beginning and on bachelor level. In our eyes this is not so much related to the amount of work, which would be also influenced by the amount of courses taken by the students in parallel, but rather to the repetition rate necessary to absorb the process in a specific timeframe.

8. Conclusions

In this paper, we discuss our experiences with academic education of professional processes in software engineering, especially addressing the balance of practical coaching activities to academic reflection. We believe that students’ learning experience can be enhanced by frequent feedback loops of agile coaching routines with processes close to practice and theoretical reflection.

First, we advance the understanding of teaching in software engineering by discussing our approach to planning and improvement of software engineering capstone courses based upon inten-
sive coaching and the notion of team routines. We present a graduate course design based on agile practices and discuss its contribution to the students practical and academic maturity through an iterative coaching routine. The intensive coaching is shorter in nature, and thus more appealing to the students as our data shows. Process research and improvement is difficult, the education of professional practices takes time and needs to be taught on the process. By making academic courses more realistic it is possible to enable a better student experience and allows access for researchers to analyze routines. The application of a generic questionnaire throughout the course allows furthering the knowledge of educators and students by means of process improvement.

Second, we discuss a concrete example of our approach by conducting a quasi-experiment on the impact of two different stand-up meeting settings. Larger setting with three separate teams in one meeting and a smaller setting where each team holds a separate stand-up. Our results show that coaching per team costs little additional effort while providing much better information exchange and student satisfaction. Instead we found that the additional effort is not only justified by its contribution to maturity of the students but also by furthering the knowledge of educators and students by means of process improvement. This contributes to continuous improvement of the course design, as well as to improvement of agile methods in practice.

9. Recommendations for Future Research and Practice

The perspective on coaching based on team routines raises a number of questions for future research. With the increased demand for coaching, the collaboration amongst coaches becomes increasingly important. On the bachelor level we are currently experimenting on the application of multiple coaches in one course. Therefore, future research is encouraged to explore our results in different contexts. For example, in different in different courses with a professional background and group sizes. Another direction is regarding the applied research method. With the increasing importance of routines in creating knowledge, how can we improve the techniques to study routines and create experiments in-class?

Acknowledgment

This research project has been supported by the Living Lab The Hague project co-funded with support from the European Regional Development Fund of the European Union. We thank all the participants for generously contributing to this study.

References


178