Addressing the Double Challenge of Learning and Teaching Enterprise Technologies through Peer Teaching

Richard Glassey
glassey@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Olle Bälter
ob1@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Philipp Haller
phaller@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

Mattias Wiggberg
wiggberg@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

ABSTRACT

Students face difficulties when transitioning from introductory programming to using more complex enterprise technologies, such as libraries, software frameworks and development kits. Whilst much literature has been devoted to how to teach introductory programming effectively, less attention is devoted towards managing the transition towards more complex technologies. This work presents the design, experience and evaluation of a module that engages students with a range of enterprise technologies. The module uses peer teaching to transfer the responsibility to students for teaching each other about the technologies. As such, this reduces the need for the teacher to invest time in preparing materials, and it is feasible to cover more technologies depending upon the number of teaching teams. The evaluation, conducted on a cohort of 34 students studying six enterprise technologies over the course of one week, revealed overwhelmingly positive experiences with this approach. For the teacher, effort for preparation and delivery was minimal, and the feedback on the module was highly positive.

CCS CONCEPTS

• **Social and professional topics → Computing education**.

KEYWORDS

Computer Science Education, Software Frameworks, Peer Teaching

ACM Reference Format:
Richard Glassey, Olle Bälter, Philipp Haller, and Mattias Wiggberg. 2020. Addressing the Double Challenge of Learning and Teaching Enterprise Technologies through Peer Teaching. In *Software Engineering Education and Training (ICSE-SEET'20), May 23–29, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3377814.3381718

1 INTRODUCTION

Modern software development typically involves developing applications with the help of various enterprise technologies (frameworks, libraries, development kits and tools). Part of the development of a novice developer towards mastery involves the ability to adopt enterprise technologies (ET) and use them effectively to develop applications. Especially in specific domains, such as web and mobile development, software frameworks and development kits simplify common tasks and increase developer productivity [18, 19]. As such, it is important for the future generation of developers, students in software engineering and related fields, to have opportunities to be able to learn about a variety of ETs during their studies [3, 6].

However, it is challenging to learn about ETs due to their associated learning curves [9]. The transition from basic programming to using ETs is a phase shift in complexity. Typically, a teacher helps to guide students through this transition. In this context there is the opportunity to delve into the details. Yet this leads to a second challenge, in that the teacher must invest time into preparing teaching materials about a particular ET, and switching to an alternative ET would incur a similar time cost over again [16]. This is a real risk as the development of existing ETs and emergence of new ETs is continuous and students may miss out on encountering a broader variety of ETs [4]. Taken together, this is the double challenge of learning and teaching ETs.

One potential solution to this double challenge of learning and teaching enterprise technologies is to use peer teaching [11], that is, transfer the responsibility of teaching onto the students. The rationale in this context is that if students could form teaching teams, focus on different ETs, then the teacher could be freed from the cost of developing teaching materials, and instead focus on choosing a set of related ETs that could then be covered more efficiently. A further benefit is that it would allow students to compare learning experiences across multiple ETs and find common issues and their resolution, with the support of their peers.

To evaluate this approach, a module on ETs was created and embedded into an intensive education programme, the Software Develoment Academy (SDA), where 34 students train to become junior developers within three months. This special programme was created in response to the 2015 migration crisis to provide education to adults attempting to integrate into the local job market in Sweden, and is a collaboration between KTH Royal Institute of

Technology and Novare Potential, a recruitment consultancy, both based in Stockholm. This context is the ideal test as students begin with a zero-assumptions module on programming foundations (object-oriented programming with Java) and later on encounter a mobile development module that is based around a popular mobile development kit (Android SDK). The goals of this research are to discover if peer teaching is an effective and economical approach to addressing the double challenge of learning and teaching ETs. The initial offering of the module was evaluated by the experience of the students enrolled (n=26).

The main contribution of this work is a novel approach to learning and teaching ETs using peer teaching. The ET module is simple, flexible and economic in terms of the effort required for the course responsible teacher. The initial findings indicate that the transfer of responsibility of teaching onto the students is overwhelmingly positive and the dual roles of teacher and student throughout the module creates an interesting dynamic for the students. Ultimately, this approach helps underline how students can take control of their education in this area.

The remainder of the paper is as follows. The background section expands upon the double challenge of learning and teaching ETs and the opportunity of peer teaching to provide a viable solution. The ET module is then described in terms of its context, structure and evaluation. The results of the evaluation of the initial offering of the ET module are presented and the paper concludes with a discussion of these findings and the overall experience.

2 BACKGROUND

This section describes the double challenge of learning and teaching about enterprise technologies, and how peer teaching might act as an economical and efficient solution. The first part of the challenge concerns the difficulties that students face when learning about ETs. There is a significant leap in complexity from learning the basics of programming to mastering the various technologies that might be useful in their future career.

2.1 Challenges in Learning Enterprise Technologies

Whilst there is a large body of ongoing research addressing the challenges and misconceptions that students face with introductory programming [13], innovative techniques like pair programming [20] and peer instruction [22], and different ways of teaching programming [7],[1] it is harder to find equivalent focus on the transition to more complex technologies. Ali et al. highlighted this trend, noting that novice software developers are able to apply good design principles in theory, yet this rarely extends into their professional practice [1]. When they are asked to design software intended to run inside a software framework they tend to abandon good design practices and focus on simply "making it work."

Research on software frameworks does provide some insights into this challenge, whilst not necessarily focused on university students. Software frameworks are recognised as important contributions towards developer productivity [18, 19]. However, Fayad & Schmidt highlighted the associated learning curve, were it may take a developer 6 to 12 months to become highly productive (with appropriate training and hands-on mentoring targeted for effective use) [9]. Furthermore, they note that this effort only becomes valuable when amortized over many projects, and the ability to truly judge the suitability of the framework only emerges once the learning curve has flattened.

Shull, Lanubile & Basili also reflected on the high learning curve associated with software frameworks, noting that the effort in developing a framework-based application is closer to maintaining an existing application, than creating an application from scratch [23]. They went on to report that very little literature can be found on using frameworks, as the focus has instead been on how framework designers should design and document their frameworks to support reuse. The common approaches they identified for supporting the use of frameworks were: patterns and recipes, formal specifications of behaviour, architectural approaches or tutorials. They concluded that an example-based approach was the most effective strategy with students in their study.

Another speculative explanation for the lack of focus on this challenge could be that various ETs, like databases, mobile development kits, or web application frameworks tend to be encountered in courses dedicated to a particular domain, such as learning how web applications are constructed, function and so on. As a result, students encounter a complex technology as a consequence of taking a particular course, and their difficulties may be attached to the course, rather than the complexity of the technologies used. The result of this learning challenge is that students struggle with these technologies without realising the underlying reasons, and they do not have the opportunity to discover better approaches to learning or to develop deeper insights.

Whilst some students may benefit from taking an advanced software engineering course (covering patterns, frameworks, components, APIs and so on), many will not have this opportunity to develop a deeper understanding of how many ETs are implemented in theory and in practice. This concern was echoed by Caspersen, Christensen, & Bærbak, that frameworks should have a stronger role in teaching curriculums as they represent a successful approach to software reuse and are already in widespread use in industry, also that frameworks represented a good opportunity to reinforce learning of design patterns and delegation-based designs, and to gain a deeper insight into the nature of software patterns [3]. Christensen, Bærbak & Caspersen in a separate work went as far to suggest that CS1 level courses should incorporate software frameworks in order to encounter the transition in complexity as early as possible [6]. However, this may not be possible in all programmes of education, where there is not enough time or resources, and the focus is not oriented towards mastery of software engineering.

2.2 Challenges in Teaching Enterprise Technologies

The second part of the double challenge concerns the difficulties in teaching enterprise technologies from the teacher's perspective. In general, the number of technologies and the pace of development is such that it becomes a near impossible challenge for a teacher to both keep pace and choose the most relevant technology for now

[1]These four works are part of the top 10 papers of the last 50 years awarded by the ACM SIGCSE chapter. Of the ten papers, 7 are directly related to CS1 and similar introductory courses.

(and for at least a sufficient amount of the future). Rather, there must be a compromise, a focus towards a carefully chosen subset, and hope that the version that course materials developed remain relevant for a good amount of time.

Take for example the area of web application development. From the early days of simple CGI and PHP scripts executing on a web server, modern web development has completely changed and is now dominated by complex client- and server-side frameworks. Lancor & Katha report on their process of choosing between the top six PHP web application frameworks in terms of contrasting needs [16]: common functionality in previous class projects, framework complexity for those new to frameworks (learning curve), and developer friendliness (availability of documentation and online resources). They ultimately decided upon the most relevant framework by implementing a web application, in two of the frameworks as well as the equivalent plain PHP web application, before arriving at a result.

In a similar effort, on recognising the rise of web frameworks and the need to embed them within students' education, Chao, Parker & Davey described their efforts in following best practice in selecting software technologies for inclusion in education [4, 21], and applying this to which web application framework to include in their course. They eventually selected Yii, a PHP based framework,[2] but went on to say: "It bears repeating that this does not mean that Yii is the best PHP framework for every situation; rather that Yii is the preferred tool based on our given criteria." In both cases, the academics involved invested a serious amount of effort into evaluating frameworks, and this may not be a luxury that all can easily afford, especially with the pace of framework development easily rendering previous efforts redundant.

Whilst it might be easy for the teacher to see the functional equivalence between technologies, students may feel they have learnt about X and not Y, and it might be the case that employers are interested not only in Y, but also A, B and C. Attempting to convey this to students is difficult, as they lack experience in learning about multiple equivalent technologies and seeing that there are important differences, but also many similarities. Thus, despite all the advances and developments, there still remains the double challenge of learning and teaching ETs in an efficient, effective and economical way.

2.3 Opportunities of Peer Teaching

One possible approach to this double challenge could be to transfer the teaching responsibility to the students instead of the teacher, sometimes referred to as learning by teaching [8], peer teaching [26], or peer tutoring [12]. An immediate consequence here is that by definition there are more students than teachers. This would remove the effort and bottleneck of choosing a single representative technology. Instead, sub-groups of students could take responsibility for teaching a particular technology to the remaining class. Thus students would be actively engaged in both teaching and learning activities. Depending upon the course goals, varying configurations of groups and technologies could be arranged, allowing either for broad coverage of many technologies, or less with a more in-depth focus on particular technologies.

[2]https://www.yiiframework.com

The increase in enrollment in computer science classes creates challenges for higher education. However, Vygotsky concluded that students learn as much from their peers as their teachers [25], so one remedy is to hire more undergraduate teaching assistants [10]. A recent literature review of undergraduate teaching assistants in computer science observed that there are many reports of their benefits, but the evidence for this seems anecdotal [17]. There is therefore a need to study not only the advantages, but also disadvantages with using students as teachers. There are tools developed to support this, such as My Digital Hand, an online tool to support scaling of one-to-one peer teaching, and at the same time give researchers the possibilities to collect data in order to improve the processes [24].

However, as a solution to the challenges described above, we would like to take the student as a teacher concept a step further by using them not only as teaching assistants, but as teachers for their peers. Peer Teaching (PT) has been used for a long time and Goldschmid & Goldschmid's review praises the potential for both the student teacher and the student learner [11]. PT contributes significantly to socio-psychological needs among students: active learning, increased cooperation, motivation, self-confidence and self-esteem. PT when introduced into introductory CS led to dramatically reduced drop-out, failing, and low grades [5]. The Goldschmids identified five types of peer learning (tutorials, proctor, group, learning cell and student counseling), where learning groups are most relevant for this particular work. Finally, group work has been repeatedly shown to encourage learning transfer between individuals in the group and prepare students for real-life work environments [2, 15].

The remainder of this work will investigate the opportunity to use PT within the context of enterprise technologies. The critical questions to be answered are: (1) does peer teaching have a positive effect for students in their experience of learning about enterprise technologies; and (2) does peer teaching have a positive economical outcome for the course responsible teacher. To help answer these questions, the next section will describe the design and evaluation of a module using PT in the context of an intensive training for newly arrived adults aiming for a career in the IT industry.

3 ENTERPRISE TECHNOLOGIES MODULE

3.1 Context within the Software Development Academy

The Enterprise Technologies module was added to the 5th iteration of the Software Development Academy (SDA), an intensive education that aims to train non-programmers into junior developers in three months. SDA was created by KTH Royal Institute of Technology in collaboration with Novare Potential, a recruitment agency, as a response to a societal demand concerning migration. The central idea was that many newcomers to Sweden had prematurely abandoned their education and careers elsewhere, and lacked the profile and networks required to find meaningful careers that matched their training and interests. Since early 2017 the programme has been delivering an intensive education and helping students find their way towards careers within the local IT industry within Stockholm. Each iteration of SDA accommodates 35 students on average.

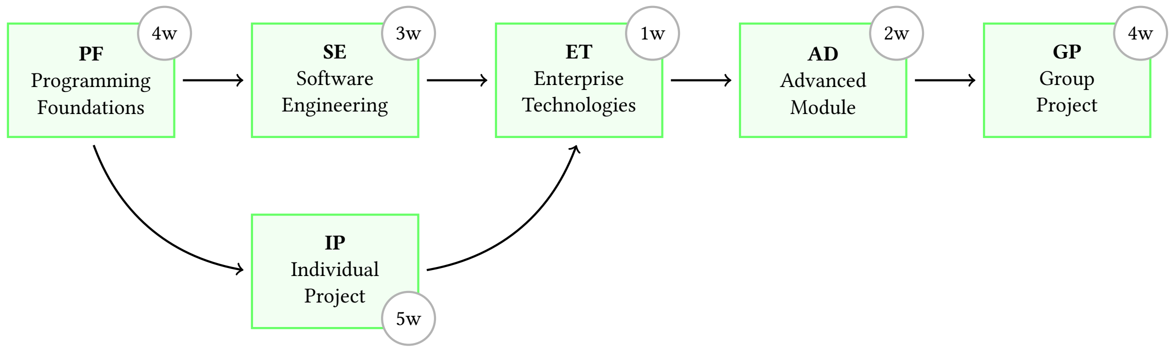

Figure 1: The SDA programme is organised into a series of six modules. The individual project (IP) runs in parallel with both programming foundations (PF) and software engineering (SE). Numbers in the white ovals indicate the duration of each module in terms of weeks.

The programme of education is organised into a series of 6 modules, as shown in Fig. 1:

Programming Foundations (PF - 4 weeks) — introduces basic programming, the Java programming language and thinking in terms of object-oriented design.

Software Engineering (SE - 3 weeks) — covers the broad topic areas from testing through to requirements engineering and project methodologies.

Individual Project (IP - 5 weeks) — Runs in parallel with PF and SE, were students are tasked with developing a specified application by themselves.

Enterprise Technologies (ET - 1 week) — introduces students to a range of important software frameworks and tools that are generally desirable to the IT industry.

Android Development (AD - 2 weeks) — focuses upon mobile application development using the Android SDK.

Group Project (GP - 4 weeks) — Students work in teams and follow the Scrum methodology to develop applications on a given domain (e.g. Fintech).

These modules are delivered over a compressed time-schedule. Typically a week of course material normally delivered at KTH Royal Institute of Technology is condensed into a single day. Each module is targeted towards topics that are both essential and desirable within the IT industry. Given the tight schedule, this creates a problem where students must rapidly advance their programming foundations to mastering a mobile application framework (e.g. Android SDK) within a matter of weeks, which leads to a lot of pressure on the students ability to transition.

At first, this was attributed to the nature of mobile development itself, or not finding the appropriate mode of delivery for the course material. However, it became apparent that students lacked the experience of rapidly learning a new framework for a particular application domain. This more general issue generated the motivation to create a module that specifically gave students the experience of getting to grips with multiple software frameworks in a short space of time. Instead of mastering a single framework, students instead should notice their ability to learn about multiple frameworks, identify what was the general learning experience and be able to apply that going forward in their education.

3.2 Overview of Module

Students received the following description of ET:

> *Enterprise Technologies (ET) covers a broad range of important frameworks that are used within modern software engineering and development of industry projects. The aim of the module is to cultivate independent learning and transmission of knowledge. Rather than have all students learn one ET, groups will research and develop different ETs and train the rest of the class.*

Furthermore, to complete the module successfully, students are required to:

(1) Research and gather training materials for a specified ET.
(2) Develop a training workshop for your peers on this ET.
(3) Deliver a workshop training for an audience of your peers on this ET.

3.3 Structure of Module

The ET module is the fourth in the SDA programme, after Software Engineering (SE) and before Android Development (AD). The high level structure of the ET module is shown in Fig. 2. Within the context of SDA, the ET module was allocated a timespan of five full working days. This was divided into two phases: two days of preparation and three days of delivery.

3.3.1 Phase 1: Preparation. The first two days were devoted to preparation. Students received a one hour session describing the ET module, including the rationale, the phases and the forms of evaluation. During this session students were divided into six teams, with up to six members per team. Students had free choice of who they wanted to work with. Once the teams had been decided, the six ETs were briefly announced and randomly allocated to the teams. There was then a five minute period of discussion to ensure that teams were happy with their allocation. For SDAv5 the following technologies were selected:

Docker - Operating-system-level virtualization — Allows all the requirements that an application needs to be bundled as a container, so that the application can run anywhere.

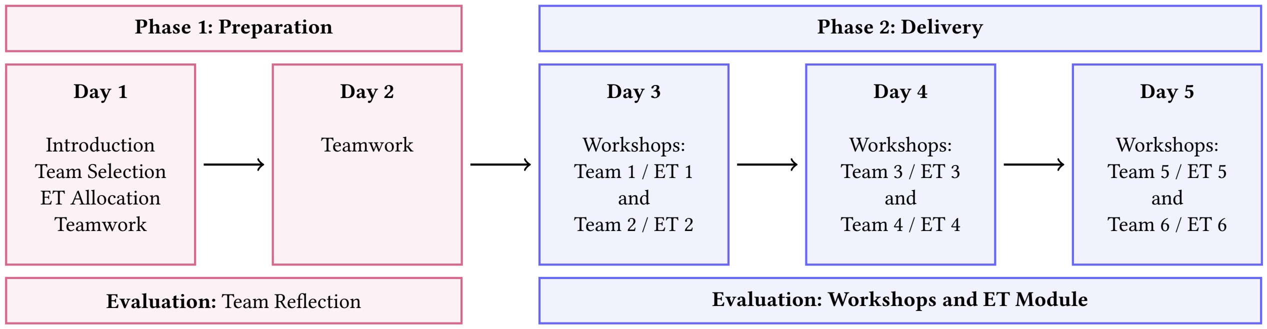

Figure 2: The breakdown of days and phases for the enterprise technology module (ET) within SDA.

Maven - Build automation tool — Allows the management of a project's build, reporting and documentation from a central configuration file.

MongoDB - Document-oriented database — Stores data using JSON documents, and is both scalable and flexible as a database solution.

Hibernate - Object-relational mapping framework — Provides a framework for mapping an object-oriented domain model to a relational database.

Play Framework - Web application framework — Creates scalable web applications following the model-view-controller pattern.

Deeplearning4j - Deep learning library — Assists with data preparation tasks and provides implementations of deep learning algorithms.

The selection of technologies was guided by the following rationale:

(1) Technology is both modern and desirable to industry.
(2) Documentation and tutorials suitable for students were available.

There was no expectation that the course responsible teacher needed to have expertise in any of the selected technologies. By following this approach, there was no need for the teachers to feel comfortable with the technology, only to be aware that it fitted into a set of possibly interesting alternatives for employers in the IT industry. The main safety net, and teacher effort, was to determine that the documentation and introductory tutorials were both findable with minimal effort and also suitable for the students given their level of training.

Once teams and technologies had been allocated, the students were left by themselves to research their specific technology and develop the following items for the delivery phase:

(1) Presentation of enterprise technology (10 min).
(2) Guided workshop, using bespoke and online materials (100 min).
(3) Feedback on workshop via a standard survey (10 min).

3.3.2 Phase 2: Delivery. As shown in Fig. 2, the delivery phase lasted for three days, and two teams per day would take control of the classroom and deliver their workshop. In the interests of setting the correct tone, and helping with any technical setup issues, the teacher was present for the first 15 minutes, allowing teachers to listen to the presentation and ensure there was an orderly transition into the workshop. After this, the teacher left the classroom and the responsibility was on the student teaching team to complete their workshop on time. In order to understand what happened during the teacher absence, and to better understand the effectiveness of this approach, the module was evaluated in three separate ways described in the next section.

3.4 Structure of Evaluation

The key idea underlying both phases was transfer of responsibility via peer teaching. Would the students be able to work as a team in (1) teaching themselves about their given technology, and (2) teaching of others in the class. On both points, very little direction was provided on how this should be conducted. The final aspect of interest was whether students would recommend this approach to be used in future iterations of the module.

The first instrument of evaluation was a survey that was circulated at the end of the preparation phase (see Fig. 2). The aim of this survey was to evaluate students experience of learning in order to teach others. The questions in the survey were organised into the following themes and all question items responses were captured on a five-point Likert scale: Strongly disagree, somewhat disagree, neutral, somewhat agree, strongly agree:

- Prior knowledge of the technology (3 statements)
- Motivation to learn to teach others (2 statements)
- Feelings about collaborating within their teaching team (7 statements)
- Reflections (open response)

The second instrument of evaluation was a survey that was circulated after the delivery phase had completed, at the end of the week. The focus here was the experience of both delivering their workshop (the teaching experience) and participating in the other five workshops as a student. The questions in this second survey were organised into the following themes:

- Leading a workshop as a teacher (6 statements)
- Experiencing workshops as a student (6 statements)

The final instrument surveyed students' general perceptions of this intensive introduction to multiple ETs (9 statements), whether they would recommend its inclusion in future iterations, and their overall reflections (open text). In addition, the module concluded with the course leader taking an hour to openly discuss the student experience of the module as well as how students felt about learning technologies in the future. All of the statements are summarised in Table 1.

4 RESULTS

Of the 34 students actively participating in the ET module, 26 answered the delivery phase survey. Whilst students were encouraged to complete this activity immediately after each of the phases, it was also explained to be voluntary and several reminders were issued, but that was as far as the encouragement went. For the sake of overview, the Likert scale was mapped to a scale from -2 to 2.

4.1 Evaluation of Preparation Phase

Few of the students had used the technology in question before (mean -1.5, STD 1.2). The teaching teams clearly did a good job of selling the ETs to their students: all would like to know more about their specific ET (mean 1.5, STD 0.5). In terms of confidence, prior to delivering their workshop, students were ambiguous about whether they could describe their ET to another classmate (mean 0.7, STD 0.7). However they were slightly more positive that their class would benefit from their workshop (mean 0.5, STD 0.9). However this had no correlation with the stress they felt about teaching their classmates (Pearsons 0.07).

When it came to the seven statements on collaboration within the teaching team they were in general slightly positive (mean 0.6, STD 1.0). No statement stands out here, but two citations from the comments illustrates both the benefits and drawbacks:

> *I was very surprised at how easy it is to learn together with my classmates. The theme that [] seemed like [a] dark forest turned out [to be] easy st[u]ff. I am grateful to my classmates and KTH Royal Institute of Technology/Novare Potential for this experience :)*
>
> *As this module is pretty open to every individual learning at their own pace. I feel that there are people in teams who are nothing more than extra baggage and they are slowing down everyone else as well. Every one has to devote their time to learn something new but if people just come in with the mindset that they would get a brief from other team members this kills the whole concept of team work. So in my opinion there has to be some kind of accountability for every team member to be able to show what they have learned.*

4.2 Evaluation of Delivery Phase

The six statements on the topic of Leading a workshop had a mean of 1.3 (STD 0.7), that is slightly better than "Somewhat agree". The two statements with most positive answers were "I was happy working in a teaching team during my workshop" (mean 1.6, STD 0.6) and "I am motivated to make improvements to my workshop after delivering and evaluating it" (mean 1.5, STD 0.5). On the other end of the scale was "Resolving student challenges made me feel more like an expert in the ET" (mean 1.0, STD 0.9). One illustrating quote from the open question is:

> *It was [a] very good opportunity to try how much I'm able to lead or help with a learning part! I feel more confident about the topic I teach and I'm very motivated about my topic! I wish if we can take more than one week with the new ET module!*

The five statements on the topic of Experiencing the workshops also had an average of 1.3 (STD 0.7). The two statements with most positive answers were "I felt comfortable asking my teaching team for help if I needed it" (mean 1.5, STD 0.6) and "I got ideas about how to improve my workshop from experiencing others" (mean 1.5, STD 0.5). The lowest ranked statement ranked was "I feel more confident learning about other ETs after the workshops" (mean 1.1, STD 0.8). One illustrating quote from the open question is:

> *I enjoyed the workshops and different styles of teaching. It was engaging and fun!*

4.3 Evaluation of ET Module

The nine statements on the topic of General ET aspects had a mean of 0.6 (STD 1.2), slightly worse than "Somewhat agree". The statement with most positive answers was "I feel this was an engaging way to learn about several ETs" (mean 1.4, STD 0.7). The most negative one was "The time allocated was sufficient to understand each ET" (mean -1.0, STD 0.9). On the question "Would you recommend that [next year's] students should also experience the new ET module?", one answered "No", four "Maybe" and 21 (81%) answered "Yes". We also got some suggestions how to improve the module. As always, students wished they had more time, but also to install the necessary software ahead of the workshops, hints on teaching, and suggestions for other ETs (to replace Deep Learning, which some thought was too much).

5 DISCUSSION

We have conducted a study of using peer teaching to address the double challenge of learning and teaching enterprise technologies. From the outset, this work has been guided by the questions of (1) does peer teaching have a positive effect for students in their experience of learning about enterprise technologies; and (2) does peer teaching have a positive economical outcome for the course responsible teacher?

In terms of (1) the student experience, the findings were mostly positive. The students were slightly positive that they both would be able to explain their ET to their classmates and that their classmates would be able to understand it. Some students were stressed about the delivery, some not, but that is probably a natural attitude when teaching, for some, probably for the first time. Furthermore, given the intensive nature of the programme this module belongs to, and the various pressures that are being experienced by the students, this was a recurring feeling due to the compressed timeframe.

Collaboration in teams is complex, and this experience was no different. Some students complain about others not pulling their weight, some praise the effectiveness of the group-based learning in the preparation phase. More instructions on how to work in teams would possibly be useful, but would also take time from other content preparation. That said, and in line with the recommendation to use groups in peer teaching [11], the notion of a teaching team

Table 1: The evaluation was divided into three surveys, delivered during the preparation, delivery and module reflection phase respectively. Each consisted of a series of statements which the students could indicate their level of agreement. Responses were recorded using a five-point Likert scale from strongly disagree to strongly agree

Preparation Phase	
Prior Knowledge	I have used this enterprise technology before this module
	I would like to learn more about this particular enterprise technology
	I feel that I can describe this enterprise technology to another class mate
Motivation	I am confident that my class will benefit from my workshop
	I feel stressed by the prospect of teaching my class mates
Collaboration	I liked working as a teaching team when learning
	Working in a teaching team was more engaging than learning by myself
	I would prefer more time alone to focus on my own learning
	Working in a teaching team felt more slow than if I had worked independently
	I found that I could help my team develop their understanding
	I found that my team helped me develop my understanding
	The collaborative aspect helped to overcome individual challenges
Delivery Phase	
Leading a Peer Workshop	I felt we delivered the learning objectives that were planned during preparation
	I was happy working in a teaching team during my workshop
	Taking the role of teacher increased my confidence in discussing the ET
	Resolving student challenges made me feel more like an expert in the ET
	The workshop evaluation provided constructive feedback that I could act on
	I am motivated to make improvements to my workshop after delivering and evaluating it
Experiencing a Peer Workshop	I liked having my fellow SDA students as teachers
	I felt comfortable asking my teaching team for help if I needed it
	I feel more confident learning about other ETs after the workshops
	The consistency of the workshops might have been better with tighter requirements
	I got ideas about how to improve my workshop from experiencing others
	After each workshop, I felt motivated to learn more about the ETs that were presented
Reflection upon ET Module	I feel this was an engaging way to learn about several ETs
	The intense format was an efficient use of time
	The presentation - workshop - evaluation format was a good fit
	The level of detail of information was generally about right
	The time allocated was too short to understand each ET sufficiently
	It might be better to focus on less ETs
	I would have preferred an expert rather than my peers as teachers
	If someone outside of SDA mentioned that they one of the ETs, you feel prepared to engage them
	I'm surprised by how much I could learn about different ETs in a short time

helps to further divide the labour within the group and acts as an important safety net to ensure that no one feels uncomfortable with the teaching responsibilities.

There were some complaints (as illustrated in the quote above) on the team working skills on some of the students. Teamwork "remains a widely reported problem in collaborative learning" [14]. As the students come from all over the world we should have anticipated this challenge and attempted to reduce it, or at least evaluated this aspect thoroughly to identify if this was a major problem, but coming from cultures where teamwork is omnipresent in schools, we did not.

However, the concerns students had before delivery seemed unfounded. When it comes to the entire concept of peer teaching within the ET module, we think the approval rating of 81% for our first attempt is encouraging. We, as teachers, learned some things that we will improve until the next iteration (e.g. advise students to install technologies in advance of workshops, work in pairs in the workshops to reduce the number of students/technical issues further, provide some guidance on how to manage a classroom and support students). When it comes to the time aspect, it is natural to want more time to do anything, but as teachers we have both time constraints and content demands to consider.

In terms of (2) the economics for the course responsible teacher, the experience was very positive. In summary, the teacher contact time was absolutely minimal. For a week of intensive teaching, only 5.5 hours were required: 1hr to introduce the concept and form the teams, 15 mins (passively) attending each workshop presentation, and 1hr for the general reflections at the end of the week. Little time was spent investigating the technologies, other than discussing the choices with other teachers as a sanity check and making sure that quality tutorials could be found. Finally, the evaluation data generated by the students during preparation and delivery phases was not overly demanding, taking about 2 hours to review. Given that this module delivered a full week of teaching activities for the students, of which the experience was overwhelmingly positive, this is a nice result for the amount of effort required. Future course offerings will incur no extra teacher cost to update materials, as they will simply be generated as a consequence of the peer teaching activity in each offering.

Taking a more critical perspective from the teaching point of view, there is the concern that whilst the time cost is very economical, considering this is an instensive week of teaching, the depth of engagement may be quite shallow. This is a consequence of targetting a range of technologies, but more time perhaps could be used to go beyond the initial hurdles of installation and sense-making of a new technology. However, the value in demonstrating students' own ability to take control of their learning, develop a basic understanding rapidly and convey this to their peers, and then apply this when required in their future careers would have more lasting value than investing in only one technology. One fair argument is that a teacher would not struggle themselves to collect basic tutorials from online sources and disseminate them, however the transfer of control is still considered to be vital for students who hope to very soon enter the IT market were they will be exposed to a lot of different technologies and may have to be agile and adaptive.

We believe the most important takeaway for the students is that they are now aware that they can learn anything in a relatively short time. During the feedback session at the end of the week, there was a clear sense of energy and enthusiasm, rather than exhaustion and overload. Students remarked about their surprise that they had managed to not only cover six technologies in such a short time, but that they felt they could now at least be conversant about them. The typical learning curve issues encountered with software frameworks [9, 23] were not observed, or at least the ET module created a more comfortable atmosphere to explore a framework with your peers. Of course, this does not suggest that we have resolved the learning curve, rather lowered the entry barrier to several within a short space of time. In the context of this specific module, in-depth mastery was never the intention, rather the broader skill of being able to encounter the fast changing world of technologies that they will have to learn as part of their future training and career development.

In terms of implications for practice, the approach can easily be modified for different course constraints. In the case presented here, one week is a consequence of a very compressed programme of education in order to rapidly make students ready for the local IT job market in Stockholm. Different configurations of teaching teams and technologies can be imagined in order to deal with different domains and class sizes. One variation could focus on a set of technologies that serve the same purpose. Rather than spend the time evaluating six web frameworks as reported by [16], six teaching teams could prepare training materials on each, teach the remaining students, and then make a comparison of them as an additional form of evaluation. Repeating this each year would easily keep pace with changing technologies without incurring any significant costs for the teacher in terms of preparing new materials. The other obvious dimension to experiment with is what happens when more time is given for futher in-depth workshops after the initial introduction has taken place. There would not appear to be much of a limit here in terms of stretching the time allocated to continue studying the technologies in parallel, although perhaps after the novelty has worn off from the initial contact with peer teaching, the impacts may not be so obvious.

For future work, the immediate concern will be to validate these positive results by repeating the course with a new cohort of students. The SDA project will run a parallel offering at Lund University in Fall 2019 and Spring 2020 with a different teaching team and environment, so there will be an excellent opportunity to compare the findings of this initial experience with the similar technologies, but in a different context. Beyond this, more attention needs to be devoted to what follows the introductory nature of the ET module and build upon the students sense of confidence in taking control of their own learning.

6 CONCLUSION

This paper has presented the design and evaluation of an approach to teaching students about enterprise technologies, such as libraries, software frameworks and development kits. The use of peer teaching had the double benefit of engaging students in the module, having the chance to assume the role of both student and teacher, but crucially in reducing the preparation and delivery effort for the course responsible teacher. Whilst the results presented here are preliminary and need to be validated through repeated course offerings, they are mostly positive towards this approach. We believe that the open and flexible nature of this module means that other teachers could easily mould it and shape it to their own situations, whether it be a broad coverage of useful enterprise technologies as presented here in the ET module, or a tighter focus on a survey of related technologies, such as web application frameworks. Either way, releasing control of course materials, avoiding the need to have mastery of a technology, and giving students the responsibility to learn and then teach others points towards a promising approach in reducing the learning curve associated with more complex technologies.

ACKNOWLEDGMENTS

The Software Development Academy (SDA) was founded by Mattias Wiggberg (PI, KTH Royal Institute of Technology), Philipp Haller (KTH Royal Institute of Technology) and Farzad Golchin (Novare Potential). SDA is funded by the private fund Marianne och Marcus Wallenberg stiftelsen, by European Social Fund (ESF) and KTH Royal Institute of Technology. The authors want to thank all of the SDA team members, regular staff and part time students, and other colleagues devoted to make the project happen.

REFERENCES

[1] Zoya Ali, Joseph Bolinger, Michael Herold, Thomas Lynch, Jay Ramanathan, and Rajiv Ramnath. 2011. Teaching object-oriented software design within the context of software frameworks. In *Frontiers in Education Conference (FIE)*. IEEE.

[2] Judy Brown and Gillian Dobbie. 1999. *Supporting and evaluating team dynamics in group projects*. Vol. 31. ACM.

[3] Michael E Caspersen and Henrik Bærbak Christensen. 2008. Frameworks in teaching. In *Reflections on the Teaching of Programming*. Springer, 190–205.

[4] Joseph Chao, Kevin Parker, and Bill Davey. 2013. Navigating the framework jungle for teaching web application development. In *Proceedings of the Informing Science and Information Technology Education Conference*.

[5] Joe D Chase and Edward G Okie. 2000. Combining cooperative learning and peer instruction in introductory computer science. In *ACM SIGCSE Bulletin*, Vol. 32. ACM, 372–376.

[6] Henrik Bærbak Christensen and Michael E Caspersen. 2002. Frameworks in CS1: a different way of introducing event-driven programming. In *ACM SIGCSE Bulletin*, Vol. 34. ACM, 75–79.

[7] Stephen Cooper, Wanda Dann, and Randy Pausch. 2003. Teaching objects-first in introductory computer science. *ACM SIGCSE Bulletin* 35, 1 (2003), 191–195.

[8] David Duran. 2017. Learning-by-teaching. Evidence and implications as a pedagogical mechanism. *Innovations in Education and Teaching International* 54, 5 (2017), 476–484.

[9] Mohamed Fayad and Douglas C Schmidt. 1997. Object-oriented application frameworks. *Commun. ACM* 40, 10 (1997), 32–38.

[10] Jeffrey Forbes, David J Malan, Heather Pon-Barry, Stuart Reges, and Mehran Sahami. 2017. Scaling introductory courses using undergraduate teaching assistants. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on computer science education*. ACM, 657–658.

[11] Barbara Goldschmid and Marcel L Goldschmid. 1976. Peer teaching in higher education: A review. *Higher education* 5, 1 (1976), 9–33.

[12] Sinclair Goodlad and Beverley Hirst. 1989. *Peer Tutoring. A Guide to Learning by Teaching*. ERIC.

[13] Lisa C Kaczmarczyk, Elizabeth R Petrick, J Philip East, and Geoffrey L Herman. 2010. Identifying student misconceptions of programming. In *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 107–111.

[14] Edward Kapp. 2009. Improving student teamwork in a collaborative project-based course. *College Teaching* 57, 3 (2009), 139–143.

[15] Herman Koppleman, Charles van der Mast, G PA, Elisabeth van Dijk, G MA, and Gerrit C van der Veer. 2000. Team projects in distance education: a case in HCI design. In *ACM SIGCSE Bulletin*, Vol. 32. ACM, 97–100.

[16] Lisa Lancor and Samyukta Katha. 2013. Analyzing PHP frameworks for use in a project-based software engineering course. In *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 519–524.

[17] Diba Mirza, Phillip T Conrad, Christian Lloyd, Ziad Matni, and Arthur Gatin. 2019. Undergraduate Teaching Assistants in Computer Science: A Systematic Literature Review. In *Proceedings of the 2019 ACM Conference on International Computing Education Research*. ACM, 31–40.

[18] Maurizio Morisio, Daniele Romano, and Corrado Moiso. 1999. Framework based software development: investigating the learning effect. In *Proceedings Sixth International Software Metrics Symposium (Cat. No. PR00403)*. IEEE, 260–268.

[19] Simon Moser and Oscar Nierstrasz. 1996. The effect of object-oriented frameworks on developer productivity. *Computer* 29, 9 (1996), 45–51.

[20] Nachiappan Nagappan, , Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, and Suzanne Balik. 2003. Improving the CS1 experience with pair programming. In *ACM SIGCSE Bulletin*, Vol. 35. ACM, 359–362.

[21] Kevin R Parker. 2010. Selecting software tools for IS/IT curricula. *Education and Information Technologies* 15, 4 (2010), 255–275.

[22] Leo Porter, Dennis Bouvier, Quintin Cutts, Scott Grissom, Cynthia Lee, Robert McCartney, Daniel Zingaro, and Beth Simon. 2016. A multi-institutional study of peer instruction in introductory computing. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. ACM, 358–363.

[23] Forrest Shull, Filippo Lanubile, and Victor R Basili. 2000. Investigating reading techniques for object-oriented framework learning. *IEEE Transactions on Software Engineering* 26, 11 (2000), 1101–1118.

[24] Aaron J Smith, Kristy Elizabeth Boyer, Jeffrey Forbes, Sarah Heckman, and Ketan Mayer-Patel. 2017. My Digital Hand: A Tool for Scaling Up One-to-One Peer Teaching in Support of Computer Science Learning. In *Proceedings of the ACM SIGCSE Technical Symposium on Computer Science Education*. ACM, 549–554.

[25] Lev Semenovich Vygotsky. 1980. *Mind in society: The development of higher psychological processes*. Harvard university press.

[26] Stephanie Wright and Emory L Cowen. 1985. The effects of peer-teaching on student perceptions of class environment, adjustment, and academic performance. *American Journal of Community Psychology* 13, 4 (1985), 417–431.