Chapter # - will be assigend by editors

ENHANCEMENT OF EXPERIENTIAL LEARNING IN SOFTWARE FACTORY PROJECT-BASED COURSE

Muhammad Ovais Ahmad – Karlstad University, Gdansk University of Technology, University of Oulu,
Kari Liukkunen - University of Oulu

Abstract: To better prepare next generation of software professional, it is important to provide opportunities for them to work on real software project along with real customer during their studies. This is the reason universities around the world offer project-based capstone course. Such courses help students to understand what they will face in the industry and experience real customer interaction and challenges in collaborative work. In regards, University of Oulu, Finland offers a software factory (SWF) course to enhance the learning and experience multicultural teamwork. This paper presents the design of the SWF course and student and teacher experiences. It discusses the importance of reflective learning diaries and serious games. Additionally, this paper examines factors in the SWF learning environment that affect student learning in the SWF course. Survey data were collected from the last six years of SWF projects. The results show that students consider the SWF to be a good collaborative learning environment that helps them achieve academic triumphs and enhances various professional skills. The learning diaries are effective for increasing students' learning experiences as well as providing an opportunity for teaching staff to monitor students' progress and offer better facilitation. These results are helpful for academic institutions and industry when developing such a learning environment

Key words: Collaborative learning, project-based learning, software engineering curricula, teamwork, software engineering

1. INTRODUCTION

In todays, digital age, the software industry is moving with a fast pace as highlighted by Marc Andreessen in his essay Why Software is Eating the World (Andreessen, 2011). This shift has also pushed educational institutions to train graduates with various skills and competencies. This shift pushing the universities boundaries related to their offerings in term of courses and programs. The major push is to create software engineering (SE) technology agnostic courses that prepare graduates that fulfill the industry needs and demands. One way to train SE graduates is to provide project-based learning through capstone courses (Howe, 2010; Walker, 2015; Erdogmus & Peraire, 2017).

Project-based learning (PBL) is common in various engineering programs and courses such as software engineering and information systems. It helps students to achieve a range of high-level goals and dealt with real life industry oriented problem or activities. In such project based courses student play active role whereas, teacher play role of facilitator and more passive. The students take active responsibility of their learning and have more control on how to solve a given problem. According to Barg and Barg (1999), project-based learning shown increase student motivation and interaction while working on real-world problem and improve social and team-working skills. Along with the technical knowledge the students also require to know the principle, methods and process while developing software project.

Educational institutions must train students in real-life practical projects where students engage in a collaborative teamwork environment and develop project management skills (Erdogmus & Peraire, 2017; Palacin-Silva, Khakurel, Happonen, Hynninen, & Porras, 2017; Wijnia, Loyens, & Derous, 2011). Universities around the world already include such capstone courses (ACM Joint Task Force, 2014). However, various transversal capabilities—leadership, decision making, negotiation, self-reflection, and the infusion of design thinking—receive little or no attention in these courses.

To provide such competencies, the University of Oulu established the software factory (SWF) learning environment/laboratory in 2012. The Oulu SWF is an infrastructure platform that serves multiple purposes to support SE research, education, and entrepreneurship. The SWF is a test bed for SE ideas and a source for original research on the development of basic scientific software (Ahmad, Liukkunen, & Markkula, 2014). It is an educational vehicle for the university where the artifacts produced in the factory serve to improve learning and provide teaching materials in close collaboration with industry (Ahmad et al., 2014; Fagerholm, Oza, & Münch, 2013). The Oulu SWF is part of a European Union SWF network (Taibi et al., 2016). The aim of the SWF is for students to share their experiences, learn in a collaborative environment, and grow to compete in the fast-growing ICT domain. The Oulu SWF laboratory is equipped with latest computers, software development tools, interactive projector, and wide range of gadgets available for student projects (i.e. smart phones, smart watches, tablets, cameras and so on).

This chapter describes the design of the SWF project-based course, assessment techniques, student perception, and teaching experiences, and discusses the importance of reflective learning diaries. The paper also aims to identify factors in the SWF learning environment that affect learning, in terms of exploring i) student achievements in term of skills gained, ii) students' perceptions of the SWF learning environment, as measured with the computer laboratory environment inventory (CLEI) (Newby & Fisher, 1997), and iii) students' attitudes toward the SWF project course, as measured with the attitude toward computers and computing courses questionnaire (ACCC) (Newby & Fisher, 1997).

This chapter is organized as follows: Section 2 sheds light on related work, and Section 3 discusses the SWF course under the pedagogic lens and presents the SWF course learning objectives, mode of delivery, overall structure, student team formation, mentoring, and student assessment. Section 4 reports students' perceptions, and Section 5 discuss the importance of student reflective learning diaries. Section 6 discusses lessons learned from the teacher perspective in the context of teaching and managing similar project-based courses. Finally, Section 7 concludes the paper and sheds light on future research work and improvement to the SWF course and alike.

2. RELATED WORK

Project- and problem-based courses prepare student for real work in the industry (ACM Joint Task Force, 2014; Palacin-Silva et al., 2017). Project-Based Learning (PBL) is based on the constructivist paradigm, which is student-driven learning approach (Yilmaz et al. 2020; Bell, 2010). PBL supports the development of social and cognitive aspects of learners (Yilmaz et al. 2020; Hung et al. 2012; Land 2004). The students as learner develop the questions, which are investigated under the supervision of teachers. In this process, teachers closely follow every step and approve before the students embark in a direction. Here, the key element is the student choice. According to Bell (2010), *PBL is a key strategy for creating independent thinkers and learners*. Lee et al. (2014) highlighted the important component for PBL is community partnerships, where students collaborate with professionals. Problem-based learning originated in medicine that subsequently adopted in other disciplines (Mann et al. 2020). According to Kolmos and de Graaff

(2014) problem-based learning approaches combine the learning approach, the social approach, and the content approach. It is basically student-centred approach which engage them in complex real-life problem with open-ended answers (Chang et al. 2018). According to (Jabarullah and Hussain, 2019) problem-based learning exposed students showed a greater inclination towards deep and strategic learning rather than surface learning. It is necessary for SE students to have hands-on experience and a glimpse of real software industry work during their studies. To prepare students for a software industry, universities around the world offer various capstone courses. The SWF project is based on capstone course concept.

Since 2010, the SWF and SWF-based courses have been offered at various universities around the world, such as the University of Helsinki, the University of Oulu, the University of Eastern Finland, the Free University of Bozen-Bolzano, Tampere University of Technology, the Free University of Cagliari, the Technical University of Madrid, Montana State University, the Catholic University of America, and Bowling Green State University (Ahmad et al., 2014; Chao & Randles, 2009; Fagerholm et al., 2013; Taibi et al., 2016; Tvedt, Tesoriero, & Gary, 2002). These SWFs aim to provide students with practical experience in software development projects and help the students to gain business experience in a collaborative environment, as well as polish their technical expertise. However, research on SWF projects and course curricula is scarce. Most studies report success stories, students experimenting with processes, and positive experiences of students' motivation in such courses or projects. We did not find any studies that reported on SWF course design or mode of delivery or that gave a detailed assessment of students' techniques.

3. THE SOFTWARE FACTORY COURSE

The SWF laboratory offering a 10 ECTS (290 h of work) advanced-level course for Information Processing Science master's degree program students at the University of Oulu, Finland. The purpose is to expose students to real-life software development projects in a multicultural collaborative environment. The focus is learning by doing—that is, managing authentic, resource-limited project work and integrating the practices of an academic expert in a unique project assigned by a software company. Each year, in the spring semester 15-20 students take the SWF course.

The SWF course is based on various learning theories or approaches—behaviorism, cognitivism, and constructivism—are taken into consideration (Anderson, 2008). The blended learning approach in SWF course goals was to maximize students' learning outcomes from three angles. First, behaviorism school, we observed how teaching staff behavior affects students' learning, e.g., teacher approval of certain items required by the course. In such cases, the teacher acts proactively to respond quickly. In this way, we avoid unnecessary wait times from the students' perspective. Second, cognitivism approach, the students are encouraged to have a mental map of their project and processes. Such encouragement is important, especially in the context of software development. The students need to have a map for a specific goal, which boosts motivation and reduces stress. Third, as constructivists, we do not push students to memorize the concepts taught during the lectures. Various serious games, learning diaries, and discussions enable students to develop their knowledge.

In summary, the SWF course is more inclined toward constructivism due to a student-centered model that focuses on learning by doing in a collaborative environment and problem-based learning. Such a collaborative course and environment have a significant impact on learning (Khine & Lourdusamy, 2003; Fagerholm et al., 2013; Ahmad et al., 2014; Taibi et al., 2016).

3.1 SWF course learning outcomes

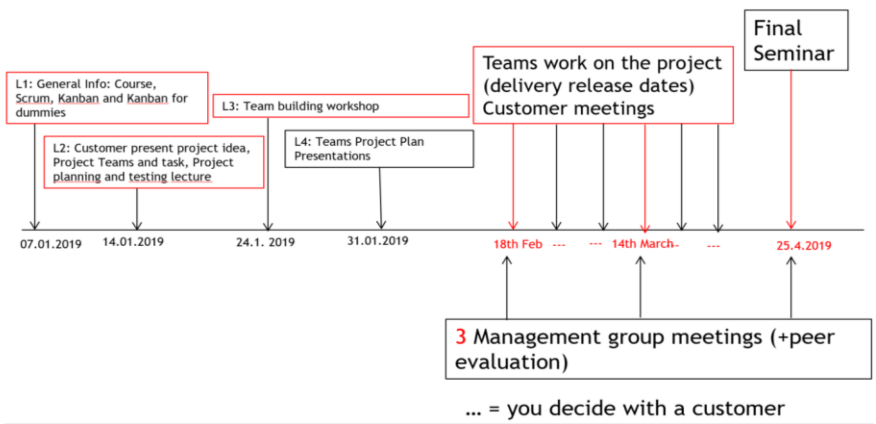
After completing the course, the students should demonstrate their ability to work on a challenging ICT project. Students learn to acquire and apply professional expertise in the topic of the project. One example of a project is a pathfinder using the Robot Operating System for an autonomous electric car. Students should be able to:

- Act as independent professional members in an ICT project; a team member collectively produces, monitors, and updates the project plan (a project with a fixed deadline and human resources).
- Search up-to-date scientific literature on the subject matter of the project to build professional expertise in the topic and apply to the project work.
- Develop analytical and creative skills for successful completion of the project and monitor and communicate the status (time and human resources used) of the project in real time within the team.
- Develop skills to communicate with customer in a professional context and manage a successful project review with the steering group/project team.
- Report and explain the status (progress, results, and future estimations of the project) to the steering group to support decision making and problem resolution concerning the project's future.
- Work as a project team member with people from different technical and/or cultural backgrounds, produce a realistic outcome in relation to the project deadline and human resources (ok, good, or excellent), and reflect on the relationship between the selected process models (evolutionary, agile, lean, etc.) and management practices followed in the project.

3.2 Mode of delivery

The SWF course adopted blended teaching or mixed-mode instruction to boost collaborative learning (Martyn, 2003). This approach has become popular in SE because it helps develop critical thinking and improves understanding of various concepts (Palacin-Silva et al., 2017). There are four major components of the SWF course: classroom lectures, serious games workshops, weekly customer meetings and monthly progress reports, and individual project work. All SWF project–related communication, materials, and deliverables are stored in the Optima workspace[1].

Even though the course had strong learning by doing roots, it has evolved more and more towards practical skills and way of working. During the first implementations there was some theoretical content and students did more written documents. Changes were based on the feedback from students and



software companies. A typical course execution structure is presented in following Figure 1. The SWF course consist of three main parts, lectures, workshops, project, and mentoring meetings.

Figure 1. Example of software factory execution plan.

Introductory lectures and first workshop provide the guiding steps for carrying out the course and discuss relevant course/project information. The students already have a background in project management and software development tools, processes, and techniques. However, in the first three introductory lectures, these concepts are briefly revised. The attractive aspects of the introductory lectures is that they are delivered with the help of software professionals from the local software industry. Oulu is a smart city, and many big ICT companies, such as Nokia, Ericsson, and Bittium, have offices in the

[1] Optima is a learning management system used at the University of Oulu, Finland. https://www.discendum.com/references/?q=optima

city. The invited professionals share their experiences and provide insights into running projects efficiently and adopting a software development and management method or technique based on project needs. Students are divided into multiple groups during the fourth lecture; each team has three to four members. Additionally, interested software companies are invited to present the project ideas to the student teams. These software companies and entrepreneurs are the real customers of the SWF projects. Such real customer involvement helps students to learn more about the dynamics of software projects and enhance their soft skills. Each team has the opportunity to select a project based on their interest.

Workshops goal is that students present their initial understanding of the project, problems expected, and possible solutions. The students present their project plan, development process, and management practices. The students break down the customer requirements and needs, as well as discussing the delivery of the project to the customer. In this phase, the teaching staff act as facilitators and guide students to successfully implement their projects. The aim of the workshop is to facilitate students through experiential learning to understand group dynamics and the software development methods, tools, and techniques used in the software industry. Various serious games are played, such as the Marshmallow Challenge, Draw Toast, Scrum Simulation with LEGO, and Poker Planning. For example, the Marshmallow Challenge is an instructive design exercise that engages students to work in teams. Such activities help students experience the dynamics of collaborative teamwork, the importance of analyzing each other's perspectives, and iteration planning.

The actual software development starts at the end of the first workshop. The students work on their projects for three months. The project team(s) experiment and select development methods, tools, techniques, and related technologies based on the project requirements and customer demand.

Frequent communication inside and outside the team is important. The project team meets the customer based on need, but the course recommends organizing a weekly meeting. Additionally, there are monthly management group meetings, where students present their overall progress, project and team challenges, and the status of the work hours. The students must also write a learning diary once a month. The teaching staff provide formal feedback during management group meetings.

3.3 Team formation and mentoring

We used a team formation tool called CATME (Hrivnak, 2013), which restricts each team to four or five students. The students' characteristics, such as ethnicity, gender, leadership preferences, specific technology skill level, and relevant knowledge, play a critical role in the formation of diverse teams.

The teaching staff separately mentor each team and monitor their performance and dynamics. The teaching staff act as facilitators, help students prioritize their tasks, provide feedback on the development process, and discuss the reflective learning diaries to enhance student learning. To track students' progress and facilitate efficiently, teachers use various techniques, such as daily stand-up meetings, agile retrospectives, burn-down charts, cumulative flow diagrams, and Kanban boards.

3.4 Assessment methods and criteria

Assessing project work and then grading individual team members is always challenging. It is the teacher's duty to fairly assess each team member and present the criteria clearly and understandably. In the SWF course, we used a rubric-based assessment. It is mandatory for students to attend all the lectures and workshops. The distribution of the SWF course assessment is summarized in Table 1.

Table 1. Software factory (SWF) course evaluation and grading.

Criteria	**Points**	**Criteria**
Final Software Product (Group Evaluation)	25	Final evaluation by the customer. The product must fulfil the customer requirements and meet functional and non-functional requirements. Additionally, customer evaluate the groups communication and meeting deadlines.
Supporting Documents (Group Evaluation)	25	Supporting documents and evidence regarding the entire development process, including planning, management, implementation process and the planned and actual work hours. The project plan is updated with each sprint, which lasts for two weeks.
Supervisor Evaluation (Group Evaluation)	10	Supervisor's evaluation of group performance. The teaching staff observe each team's progress from the beginning until the final presentation of the project. The teacher provides feedback after every weekly customer meeting and guides the students to improve their preparation for the next steps'. The teachers also consider how the teams prepare their presentations, handle technology during the meetings, and work together internally.
Individual Reflective Biweekly Learning Diary	20	Individual three reflective reports (3 × 5 points = 15) Final lesson learned report (5 points).
Peer Evaluation (Individual Evaluation)	20	Peer evaluation of individual performance by other members of the team.
Total	100	Final grade.

The peer evaluation and reflective bi-weekly learning diaries are out of SWF course long journey. Each year, we learnt new things and revised these two evaluation parts. For example, the weekly learning diaries were very exhaustive for the students and hindering the sprint planning and execution. The peer evaluation brings more visibility and transparency to each individual student contribution in the group project. Below we will explain the mentioned two assessment parts in detail.

It is mandatory for every student to write four diary entries during the project. Reflective diaries are core elements of self-regulated learning that promote the development of metacognitive strategies (Fulwiler, 1986). Fulwiler described the rationale for introducing reflective diaries as follows: “In the academic world, where we teach students to gain most of their information from reading and listening, we spend too much time telling our students how to see or doing it for them. That’s not how I would encourage critical, creative, or independent thinking. Our students have good eyes; let’s give them new tools for seeing better: journal writing is, of course, one of those tools.”

Peer assessment is a powerful meta-cognitive tool, which has been advocated in various studies (Kaufman, Felder, & Fuller, 2000; Layton & Ohland, 2001; McGourty, Dominick, & Reilly, 1998). According to McGourty et al. (1998): “In a cooperative learning environment, students themselves are often in the best position to provide one another with meaningful feedback regarding both their technical and interpersonal performance.” A number of peer assessment tools and advised reducing the possibility of a student intentionally “damaging” his or her peers’ scores and ensuring that students do not feel that they are “ratting” on their peers (McGourty et al., 1998; Nicole, Pamela, & Rebecca, 2005; Wilkins et al., 2000). In the SWF course, students are required to fill out a form to report aspects of their team members’ contribution and behaviour characteristics. The Oulu SWF project peer assessment is based on Sanders (Ohland & Layton, 2000; Sanders, Dean, Sanders, & Dean, 1984). Examples of teamwork characteristic statements include attending team meetings, contributing to the discussion at the meetings, completing tasks on time, and the team member’s ability to work with the other team members. The students were asked to score the characteristics using the following scale: Always (2 points), Usually (1), Sometimes (0), Rarely (-1), and Never (-2). Furthermore, students have the opportunity to express their feedback in answer to an open-ended question and report their concerns in detail

4. STUDENT PERCEPTION SURVEY

The objective of survey is to identify factors affect students learning in a SWF and explores the relationships between student achievement, and their perceptions regarding SWF learning environment. The teaching staff conduct a voluntary online survey annually at the end of the SWF course (2012–2018). In six years, 50 of 90 students participated in the survey. The students' perceptions of the SWF course and facilities are collected based on the CLEI and the ACCC (Newby & Fisher, 1997)(see Table 2). Survey questions use a five-point Likert scale, ranging from 1 (strongly disagree) to 5 (strongly agree). Studying the learning environment (i.e. the SWF laboratory) is one way to explore student perception (Kolb & Kolb, 2005; Newby & Fisher, 1997). The following Table 2 shows that the reliability of the factor measurement is high; Cronbach's alpha varies between 0.597 and 0.951. These values show that the CLEI and ACCC constructs are internally consistent and reliably measured.

Newby and Fisher (1997) developed the CLEI to measure students' perceptions of their learning environment. The CLEI has five constructs: student cohesion, open-endedness, integration, technology adequacy, and laboratory availability. In the Oulu SWF context, the laboratory availability construct is not relevant, as the SWF laboratory is assigned to SWF project students 24/7. Furthermore, with the help of the ACCC (Newby & Fisher, 1997), students' attitudes toward computers and computing courses are assessed. The ACCC consists of four constructs: anxiety, enjoyment, usefulness of computers, and usefulness of course. In the Oulu SWF context, we removed the usefulness of computers construct because the targeted students are in the final year of the Information Processing Science master's degree program. These students are proficient in the use of computers.

Table 2. Student perception survey based on CLEI constructs.

Measures	**Constructs**	**M**	**SD**	**α**
Computer Laboratory Environment Inventory (CLEI)	Student Cohesion	4.24	.63	.846
	Open-endedness	3.84	.75	.694
	Integration	3.41	.61	.527
	Technology Adequacy	3.28	1.07	.951

Students' perceptions about SWF laboratory is measure using the CLEI; their perception is quite positive. The highest mean scores are for "student cohesion" and "open-endedness" (mean = 4.24 and 3.84, respectively). "Technology adequacy" had the lowest mean score (mean = 3.28). The students feel confident and support each other in their project work. The students also have a positive response for "open-endedness" (average mean = 3.84), which might be due to the close industrial collaboration. The SWF project is developed iteratively which helps them to obtain frequent feedback from customer. The students collectively work toward the same goal and seek a solution for the given problem. The students are encouraged to use and put their theoretical knowledge into practice in the SWF project. The teaching staff facilitate throughout the project life. The SWF laboratory is fully equipped with the latest technology, which is very important for executing such student projects. For example, the students have access to the latest computers, various types of tablets, smartphones, and smartwatches. This access is why students provide positive responses to the "technology adequacy" construct (mean = 3.28). This positive perception shows that CLEI constructs play an important role in student learning. Various studies reported a positive association between environmental and attitudinal constructs (Al-Qahtani, 2012; Pyatt & Sims, 2012; Saadon & Liong, 2012).

The ACCC constructs in Table 3, show that student have positive attitudes toward the SWF course. The constructs "usefulness of course" and "enjoyment" score the highest (mean = 3.53 and 3.21, respectively). The students found the course useful because they work with a real software company project, develop the product or services based on customers' requirements, and manage and monitor their activities. Further, the students experience first-hand encounters with a real customer, which helps them to learn negotiation skills and develop for future jobs. The "anxiety" construct received the lowest mean score (mean = 2.64), which indicates that the SWF course is exciting and make them bore or create bad experiences. This is also evident in the "enjoyment" construct, which received a mean score of 3.21.

Table 3. Student perception survey based on ACCC constructs.

Measures	**Constructs**	**M**	**SD**	**α**
Attitude Toward Computers & Computing Courses Questionnaire (ACCC)	Anxiety	2.64	.83	.674
	Enjoyment	3.21	.55	.597
	Usefulness of Course	3.53	.72	.812

In an open-ended question, the students expressed some of their anxious encounters as:

- "*Groups [one project assigned to one team] have huge gaps and differences because some students are very modest, and they felt shy to say that they are good at programming, and it is also hard to say and to evaluate if one person is good at programming. While the course really works, and I learned so many things from this course, and this is more like a practical course.*"
- *"Confusion and sense of competition kept me on my toes for the whole length of the project. I like the course concept, but the variety of cultures among students brings variation in multiple aspects of how the projects flow. After understanding how human the students all are, I was able to forgive and work in a way I felt comfortable with."*
- *"Working in a group where people do not listen or understand what you are talking about when discussing web architectures, object-oriented programming, and other topics makes working in the project depressing."*

- *"When dividing the team, it is better to do it based on the required technical skills distinction, rather than culture differences. It could involve more interaction of the other teams and customers."*
- *"It is a 10 credit course, which means it has a big grade, and all the time, I think about not getting a bad grade rather than learning. It has a bad feeling to think that failing this course might really mess up ones study time in the school."*
- *"I think customers should be more involved in the project."*

The students work collaboratively on the SWF project, which helps them gain or improve various competencies as shown in Figure 2.The students rate themselves highly against the achievement of various competencies gained during the SWF project. Building positive relationships with multicultural team members received the highest value, which contributes to developing a shared mental model, managing tasks effectively, solving complex problem, and better negotiating inside and outside the group.

Figure 2. Competencies Gained During the SWF Project Course

The students expressed their positive experiences and competencies gained during the SWF project as follows:

- *"The [SWF] project allowed me to develop my project management skills as a whole, from planning to scheduling, task allocation to having formal meetings, and collaborating with a steering group. I gained a good insight into what duties a project manager has and what kind of personal traits are needed for the successful management of a project."*

- *"My teamwork and communication skills improved in project."*
- *"I learned about interesting technologies and software tools in usability activities in practice."*
- *"Management has also increased my ability to deal with the different nature of people, how to motivate them and drive or steer them to do the work. Everyone has his own working style to do the work."*
- *"From this course, I found my direction for future work, and I found things that I like to do. Thanks to the teachers, customers, and also my group members!"*

Overall, with these competencies, the students clearly worked collaboratively enable them to generate ideas, solve complex problems, and offer opportunities to form supportive networks in pursuit of improved outcomes. These results are in line with existing literature (Saadon & Liong, 2012; Jane Burdett, 2003).

5. REFLECTIVE LEARNING DIARIES IMPORTANCE

Writing reflective diaries is the core element in many medical, nursing and teacher education programs (Tang, 2002). The reflective diaries can help students to document and redirect their learning to better prepare for challenge position in the software industry. In SWF, reflective learning diaries enable students to see how they can better prepare themselves for the challenging SE profession. Every student writes biweekly learning diary to answer three main questions (*What tasks did you do in this sprint? As a learner, what did you improve or learn compared to the last sprint? What were the issues and challenges you faced in this sprint?*).

It is an opportunity for students to reflect on their individual and group experiences for each sprint. Thus, they can identify their own learning, polish existing skills, and seek improvement opportunities. It also help students to adapt to individual project needs by understanding software development methods, practices, and tools and their application during software development projects and experiments. For teachers it is provide opportunity to observe each student progress and make the necessary arrangements to assist. It is evident that students reflect on each sprint practice, seek improvement, and adapt development practices based on their experiences. The following are example statements from the students' learning diaries:

- *"We use Trello [online tool; www.trello.com] for task management like Kanban board; our understanding of the Kanban process was minimal. Moving our process to the white board in the Software Factory Lab, we had a deeper course to interact with one another, gather feedback from* *the supervising teacher, etc. Petty issues, such as missing work-in-progress limits on our board, were quickly raised by the supervising teacher. We thus had to move our work process from Trello to the white board in the Software Factory Lab just to not to repeat ourselves but use a common board and approach."*

- *"Being a multicultural team, sometimes it was hard understanding other people, which affected our work output to an extent. We tried to do the best, however, working schedule of team members was quite problematic. While there is no one to blame, we need to work together, find more consensus, interact on Slack more, be as productive as possible during the few minutes we have together, and persevere in the given task."*
- *"We come from different cultures, different languages, and possess different accents; it was quite difficult for team members to understand each other sometimes during conversations. However, when communicating on Slack, these issues were not present. I believe this was one of the reasons why communication on Slack was more frequent"*

In summary, reflective diaries offer opportunities for students to think critically, look back on the learning activities, help identify what they have learned throughout the practical software development project.

6. TEACHERS' PERSPECTIVE

As mentioned earlier, course evolved during the years towards more practical direction. It was learned that planning of the project course with real customers in advance has to take count on following issues:

- *Different universities in different areas around the world give very different practical skills set.*
- *Most of the students have not been working in the company environments before or during the studies. So, worker skills and good examples of professional working ways are missing. These are e.g. formal meeting procedures and customer presentations.*
- *Project management is not an easy issue for student groups.*

The teaching staff implement the SWF project in relation to the results and adapt teaching techniques to optimize learning outcomes. In this regard, the following lessons were learned:

- *The SWF laboratory involves the local software industry for real projects and customers for the university's SE students. However, non*-disclosure *agreements must usually be signed. This requirement must be communicated very clearly to students to avoid breaches. Similarly, the message should be communicated to the companies that the students might not be aware of the seriousness of confidentiality, and the companies should be careful when assigning confidential tasks.*

- *The company and real customers are invited to the final project presentation. However, their evaluation should not focus only on the final product. The evaluation criteria should be clearly communicated to these external stakeholders to avoid confusion and promote fair evaluations.*
- *The SWF project course design is very effective for motivating students and plays an active role throughout the project. For example, the SWF project has an almost zero dropout rate, despite requiring intense work over one semester. This is why it is important to include serious games and consider the gamification approach to improve students' motivation and active participation and increase collaboration (Glover & Glover, 2013; Sanmugam et al., 2016; Sheth, Bell, & Kaiser, 2012).*
- *The SWF project course is also demanding from the teaching staff perspective due to the frequent communication and mentoring. Each SWF project team requires a teaching assistant to provide technical support, monitor their progress continuously, and facilitate throughout the SWF journey.* This technique is very effective in a SE project-based environment (Palacin-Silva et al., 2017; Walker, 2015).
- *Creating balanced teams is a challenge with multicultural and heterogenous students. An unbalanced team with inadequate skills or cultural conflicts can create difficult situations during the long and intensive work period. The teaching staff must proactively oversee the teams' work and communication.*
- *Almost all universities around the world have strong policies regarding the installation of computer laboratory software. Such policies affect students when they need to urgently install software packages. The solution is to install a virtual machine on all students' laboratory computers. This enables students to freely install and update the required software, applications, and tools.*

7. CONCLUSION

This chapter provides a detailed description of a project-based SE graduate course, insight into course delivery, course assessment, peer evaluation, and the use of tool support for forming teams. Additionally, we also documented our six years teaching experiences, students' perceptions of the SWF laboratory, and the SWF course.

The SWF laboratory is an innovative learning environment that offers a graduate-level project-based course, where students learn SE processes and building software development products or solutions with real industrial customer. Our SWF experience shows that it is very important to maintain a balance between coding and SE development processes. The students can easily ignore development processes and start focusing on their own individual coding tasks. The main findings from the six years SWF course execution is that allow student to experiment with the software developments methods and process. Learning and mastering these processes and worker skills are essential to compete in the competitive job market.

The student survey results show that the majority expressed a positive view in two ways: (i) The SWF course is appreciated as an important course in their master's degree curriculum. The SWF project is a good blend of theoretical and practical training that enhances students' enjoyment, and they find the course useful for achieving the required competencies for future jobs; (ii) The SWF laboratory makes it possible to provide opportunities for students to interact with real software industry customers and work collaboratively in a multicultural environment. These findings are aligned with the results of several researchers (Al-Qahtani, 2012; Pyatt & Sims, 2012; Ahmad et al., 2014; Cico et al. 2020).

Fair assessment is a very important and complex activity for teaching staff in such courses. We developed a matrix that considers various aspects of teamwork. Along these lines, free riders and hardworking students in the project can be identified and treated fairly. Peer assessment and individual reflective learning diaries play an important role in encouraging and motivating students in this collaborate course. Reflective learning diaries is an excellent technique to boosts student engagement, assess their epistemological beliefs and conceptions of learning. On the other hand, it provides an opportunity for the teaching staff to create and update strategies for monitoring and regulating learning. For a teaching staff this kind of course require a good understanding how SW development projects are executed in company environments. The peer assessment findings are align with other studies (Li et al. 2020; Leenknecht et al. 2020).

In future work, the SWF course could be run in a geographically distributed context in cooperation with other SWFs and other universities or even as a part of companies' development teams However, this might require much greater technical competencies among teaching support staff and strong collaboration between universities and participating companies. It would be interesting to investigate such courses in different academic and cultural settings.

REFERENCES

ACM Joint Task Force. (2014). *Software Engineering 2014 - Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. Retrieved from https://www.acm.org/binaries/content/assets/education/se2014.pdf

Ahmad, M. O., Liukkunen, K., & Markkula, J. (2014*). Student perceptions and attitudes towards the software factory as a learning environment.* In IEEE Global Engineering Education Conference, EDUCON.

Al-Qahtani, M. F. (2012). *Students' perception and attitude towards computer laboratory learning environment.* Educational Research, 3(4) pp. 402-411.

Anderson, T. (2008). Theory and practice of online learning. Athabasca University Press.

Andreessen, M. (2011). *Why Software Is Eating The World*. The Wall Street Journal. Retrieved from http://www.djreprints.com.

Barg, M., & Barg, M. (1999). Problem Based Learning for Foundation Computer Science Courses. Basser Department of Computer Science, University of Sydney.

Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. The clearing house, 83(2), 39-43.

Cico, O., Jaccheri, L., Nguyen-Duc, A., & Zhang, H. (2020). Exploring the intersection between software industry and Software Engineering education-A systematic mapping of Software Engineering Trends. Journal of Systems and Software, 110736.

Chao, J., & Randles, M. (2009). *Agile Software Factory for Student Service Learning*. In 2009 22nd Conference on Software Engineering Education and Training. pp. 34–40. IEEE.

Chang, J.C., Hsiao, Y.D., Chen, S.C. and Tsung-Ta, Y. (2018), "Core entrepreneurial competencies of students in departments of electrical engineering and computer sciences (EECS) in universities", Education+ Training, Vol. 60 Nos 7/8, pp. 857-872.

Dybå, T., & Dingsøyr, T. (2008). *Empirical studies of agile software development: A systematic review*. Information and Software Technology, 50(9–10), 833–859.

Erdogmus, H., & Peraire, C. (2017). *Flipping a Graduate-Level Software Engineering Foundations Course.* 39th International Conference on Software Engineering: Software Engineering Education and Training Track. pp. 23–32.

Fagerholm, F., Oza, N., & Münch, J. (2013). *A Platform for Teaching Applied Distributed Software Development: The Ongoing Journey of the Helsinki Software Factory*. 3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development. pp. 1-5.

Fulwiler, T. (1986*). Seeing with journals*. The English Record, 32(3), 6–9.

Glover, I., & Glover, I. (2013). *Play as you learn: gamification as a technique for motivating learners.* Proceedings of EdMedia 2013--World Conference on Educational Media and Technology. EdMedia + Innovate Learning. Vol. 2013. AACE.

Howe, S. (2010). *Advances in Engineering Education Where Are We Now? Statistics on Capstone Courses Nationwide.* Advances in Engineering Education.

Hung, C. M., Hwang, G. J., & Huang, I. (2012). A project-based digital storytelling approach for improv-ing students' learning motivation, problem-solving competence and learning achievement. Journal of Educational Technology & Society, 15(4), 368–379.

Hrivnak, G. A. (2013). *CATME Smarter TeamworkCATME* Smarter Teamwork (www.CATME.org). Academy of Management Learning & Education, 12(4), 679–681.

Jane Burdett. (2003). Making Groups Work: *University Students' Perceptions*. International Education Journal, 4(3), 177–191.

Jabarullah, N. H., & Hussain, H. I. (2019). The effectiveness of problem-based learning in technical and vocational education in Malaysia. Education+ Training.

Kaufman, D. B., Felder, R. M., & Fuller, H. (2000). *Accounting for Individual Effort in Cooperative Learning Teams*. Journal of Engineering Education, 89(2), 133–140.

Khine, M. S., & Lourdusamy, A. (2003). *Blended learning approach in teacher education: combining face-to-face instruction, multimedia viewing and online discussion.* British Journal of Educational Technology, 34(5), 671–675.

Kolb, A. Y., & Kolb, D. A. (2005). *Learning Styles and Learning Spaces: Enhancing Experiential Learning in Higher Education.* Source: Academy of Management Learning & Education (Vol. 4).

Kolmos, Anette, and Erik de Graaff. (2014). Problem-Based and Project-Based Learning in Engineering Education–Merging Models. In Cambridge Handbook of Engineering Education Research (CHEER), edited by A. Johri and B. M.Olds, 141–160. New York: Cambridge University Press.

Layton, R. A., & Ohland, M. W. (2001). *Peer Ratings Revisited: Focus on Teamwork, Not Ability*. In Proceedings of the 2001 American Society for Engineering Education Annual Conference & Exposition.

Land, S. M. (2004). The design and evaluation of a CSCL tool to support reflection and revision of design projects. Journal of Computing in Higher Education, 16(1), 68–92.

Lee, J. S., Blackwell, S., Drake, J., & Moran, K. A. (2014). Taking a leap of faith: Redefining teaching and learning in higher education through project-based learning. *Interdisciplinary Journal of Problem-Based Learning*, 8(2), 2.

Leenknecht, M., Wijnia, L., Köhlen, M., Fryer, L., Rikers, R., & Loyens, S. (2020). Formative assessment as practice: the role of students' motivation. *Assessment & Evaluation in Higher Education*, 1-20.

Li, H., Xiong, Y., Hunter, C. V., Guo, X., & Tywoniw, R. (2020). Does peer assessment promote student learning? A meta-analysis. *Assessment & Evaluation in Higher Education*, 45(2), 193-211.

Martyn, M. (2003). *The Hybrid Online Model: Good Practice*. Educause Quarterly, 26(1), 18–23.

Mann, L., Chang, R., Chandrasekaran, S., Coddington, A., Daniel, S., Cook, E., ... & Dohaney, J. (2020). From problem-based learning to practice-based education: A framework for shaping future engineers. *European Journal of Engineering Education*, 1-21.

McGourty, J., Dominick, P., & Reilly, R. R. (1998). *Incorporating student peer review and feedback into the assessment process*. Frontiers in Education Conference. Moving from "Teacher-Centered" to "Learner-Centered" Education.

Newby, M., & Fisher, D. (1997). *An Instrument for Assessing the Learning Environment of a Computer Laboratory.* Journal of Educational Computing Research, 16(2), 179–190.

Nicole, C., Pamela, D., & Rebecca, S. (2005). *Self and peer assessment in software engineering projects.* In Proceedings of the 7th Australasian conference on Computing education - Volume 42 (pp. 91–100). John Garratt Publishing.

Ohland, M. W., & Layton, R. A. (2000). *Comparing the Reliability of Two Peer Evaluation Instruments The impact of integration on student's persistence. View project Optimizing Student Team Skill Development using Evidence-Based Strategies View project Comparing the Reliability of Two Peer Evaluation Instruments*.

Palacin-Silva, M., Khakurel, J., Happonen, A., Hynninen, T., & Porras, J. (2017). *Infusing Design Thinking into a Software Engineering Capstone Course*. 30th Conference on Software Engineering Education and Training. 212–221.

Pyatt, K., & Sims, R. (2012). *Virtual and Physical Experimentation in Inquiry-Based Science Labs: Attitudes, Performance and Access*. Journal of Science Education and Technology, 21(1), 133–147.

Saadon, S., & Liong, C.-Y. (2012). *Perception of Students on Services at the Computer Laboratory: A Case Study at the School of Mathematical Sciences, Universiti Kebangsaan Malaysia.* Procedia Social and Behavioral Sciences, 59, 117–124.

Sanders, D., Dean, Sanders, & Dean. (1984). *Managing and evaluating students in a directed project course*. ACM SIGCSE Bulletin, 16(1), 15–25.

Sanmugam, M., Mohamed, H., Zaid, N. M., Abdullah, Z., Aris, B., & Suhadi, S. M. (2016). *Gamification's Role as a Learning and Assessment Tool in Education*. International Journal of Knowledge-Based Organizations.

Sheth, S., Bell, J., & Kaiser, G. (2012). *Increasing Student Engagement in Software Engineering with Gamification.*

Taibi, D., Lenarduzzi, V., Liukkunen, K., Lunesu, I., Matta, M., Fagerholm, F. Ahmad, M. (2016). *"Free" Innovation Environments: Lessons learned from the Software Factory Initiatives*.

Tang, C. (2002). *Reflective diaries as a means of facilitating and assessing reflection*. In Proceedings of the 29th HERDSA Annual Conference. pp. 7–10.

Tvedt, J. D., Tesoriero, R., & Gary, K. A. (2002). *The Software Factory: An Undergraduate Computer Science Curriculum.* Computer Science Education, 12(1–2), 91–117.

Walker, A. Andrew E. (2015). *Essential readings in problem-based learning : exploring and extending the legacy of Howard S. Barrows*.

Wijnia, L., Loyens, S. M. M., & Derous, E. (2011). *Investigating effects of problem-based versus lecture-based learning environments on student motivation*. Contemporary Educational Psychology, 36(2), 101–113.

Wilkins, D. E., Lawhead, P. B., Wilkins, D. E., & Lawhead, P. B. (2000). *Evaluating individuals in team projects.* ACM SIGCSE Bulletin, 32(1), 172–175.

Yilmaz, R., Yilmaz, F. G. K., & Keser, H. (2020). Vertical versus shared e-leadership approach in online project-based learning: a comparison of self-regulated learning skills, motivation and group collaboration processes. *Journal of Computing in Higher Education*, 1-27.

AUTHOR INFORMATION

Complete name: Muhammad Ovais Ahmad

Institutional affiliation: Karlstad University

Institutional address: Department of Mathematics and Computer Science

Complete mailing address: Universitetsgatan 2, 651 88 Karlstad, Sweden

Telephone number: +358503482291

Email address: ovais.ahmad@kau.se

Dr. Muhammad Ovais Ahmad (Ph.D.) is a Senior Lecturer at Karlstad University, Sweden. He received his PhD in Information Processing Science from the University of Oulu. Previously, he worked as European Masters in Software Engineering program coordinator at M3S research unit at University of Oulu, Finland and Visiting Professor of software engineering at Gdansk University of Technology, Poland. His research interests are software development methodologies, process assessment and improvement, software engineering curriculum and pedagogy, optimization techniques, technology and digital transformations.

Complete name: Kari Liukkunen

Institutional affiliation: University of Oulu

Institutional address: M3S

Complete mailing address: Pentti Kaiterantie 1, 90014, Oulu, Finland

Telephone number: +358405949975

Fax number (optional):

Email address: kari.liukkunen@oulu.fi

Dr Kari Liukkunen (Ph.D., M.Ed.) has over 30 years of management and R&D experience in industry and academy. He has worked at University of Oulu since 2001. Now he is working as an Adjunct Professor and the head of the Software Factory laboratory at M3S research group. He has worked as a Visiting Professor in Fraunhofer IESE, Germany. He was also nominated as a Visiting Professor to Nanking Institute of Technology 2019. He has published over 60 papers in international conferences and journals and served as chair and committee member in organizing numerous international conferences. His main research interests are service design, SW development processes and educational technology.