A Legacy Game for Project Management in Software Engineering Courses

Jefferson Seide Molléri, Javier Gonzalez-Huerta, Kennet Henningsson
Software Engineering Research Lab Sweden, Blekinge Institute of Technology
Karlskrona, Sweden
{jefferson.molleri,javier.gonzalez.huerta,kennet.henningsson}@bth.se

ABSTRACT

Background: Software project management courses are becoming popular for teaching software engineering process models and methods. However, in order to be effective, this approach should be properly aligned to the learning outcomes. Common misalignments are caused by using a correct degree of realism or an appropriate instruction level.

Objective: To foster students to acquire knowledge (theoretical and practical) that enables them solving similar challenges to the ones they will face in real-world software projects.

Methods: We prototype and validate a legacy game that simulates the software development process. Students are required to plan and manage a software project according to its specification provided by the teachers. Teachers act as both customers and moderators, presenting the challenges and guiding the students' teamwork.

Results: Both students' and teachers' perception suggest that the proposed game has potential to motivate the knowledge acquisition through problem-solving. The feedback also suggests that some measures must be taken to ensure the pedagogical alignment and a fair game.

Conclusion: The lessons learned provide suggestions for adopting this or similar games in the context of project courses. As further work, we plan to describe and extend the game rules based on the results of this application.

CCS CONCEPTS

• **Social and professional topics** → **Software engineering education**; • **Software and its engineering** → **Software development methods**;

KEYWORDS

Education, Gaming, Software Development Process, Project Management Course

ACM Reference Format:
Jefferson Seide Molléri, Javier Gonzalez-Huerta, Kennet Henningsson. 2018. A Legacy Game for Project Management in Software Engineering Courses. In *ECSEE'18: European Conference of Software Engineering Education 2018, June 14–15, 2018, Seeon/ Bavaria, Germany*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3209087.3209094

ECSEE'18, June 14–15, 2018, Seeon/ Bavaria, Germany

ACM ISBN 978-1-4503-6383-9/18/06…$15.00
https://doi.org/10.1145/3209087.3209094

1 INTRODUCTION

Project management courses (PMC) are intended to provide students with a dynamic environment that mirror the real-world challenges [5]. This problem-based learning (PBL) approach is supported by the constructivist theories for knowledge acquisition and student-focused approach to teaching [10, 14].

A common issue related with the PMC approach is how to find a balance between the level of realism and the relevance of the contents students will learn [5, 10]. Another issue is whether the students receive the appropriate guidance for problem-solving [14].

From our experiences in previous project courses, we would also add another risk: which is that students are more focused on the technical challenges of the project tasks [5]. Thus, they are sometimes willing to "hack" the solution instead of focusing on the software development practices and models they are intended to be following. This permissive behaviour might prevent them from experience and reflect about the intended learning outcomes (LOs).

Therefore, at Blekinge Institute of Technology (BTH), we are currently developing a legacy game to expose students to a software development project and the different software process models. The game represent some of the challenges to be faced when dealing with the planning and management of a real-life software project without requiring students to develop the software system.

We piloted this game in the context of a software engineering course at BTH, so as to first illustrate the development models, but also aiming at gathering empirical evidence on how effective it can be as a means for the students to learn and reflect about those development models. This paper presents preliminary results after piloting the legacy game in a Software Engineering course, as well as students' perception regarding the legacy game approach.

2 BACKGROUND & RELATED WORK

2.1 Pedagogical Philosophies

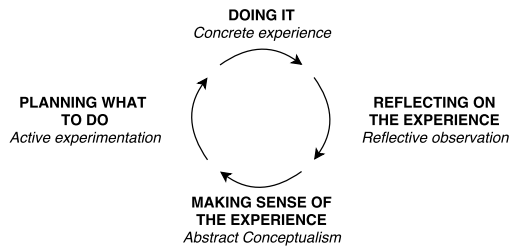
Problem-Based Learning (PBL) promotes active learning and knowledge acquisition through group work [14, 23]. Students are presented with a situation that requires a solution, whereas the teachers act as supervisors (and sometimes simulated customers), stirring the group toward a potential solution. The PBL takes several meetings, and in the time between meetings, students should look for deepen their knowledge regarding the problem.

The concepts above are connected to the **"learn by doing" philosophy** [8] and early stages of **design-implement experiences** [24]. These views favor relevant and practical learning and are particularly useful for skill-oriented engineering courses/programs. A further step on the design-implement framework comprises the implementation of the potential solution in a "hands-on" approach.

PBL is a teaching practice related to the **constructivist theories**, in which the learned is directly responsible for the knowledge construction [9, 13], a.k.a. **student-focused approach**. It aims to produce knowledge by connecting the students' prior knowledge to new facts and understanding. PBL extensively uses reflection, critical thinking and experimentation as learning facilitators.

Kolb [17] describe those stages in an experiential learning cycle (illustrated in Figure 1). In the first stage (i.e. doing it) the student is faced with a new experience, herein the game challenge. In a group, the student is foster to reflect on the challenge (stage 2) and make sense of a candidate solution (stage 3). Finally, the student applies the solution and gather its results. The cycle starts again, as the student progresses into a deeper understanding of the topic.

Figure 1: Learning cycle by Kolb [17]



2.2 Project Management Courses

PMC has become increasingly popular in software engineering education, particularly at the end of undergraduate and subsequent graduate programs [4, 22]. Ideally, PMCs should be aligned to problem-based learning and related pedagogical philosophies (see Section 2.1) but this is not always the case [10, 22].

The learning activities usually employed in PMCs are either small toy examples that lack the real complexity of a real-world project, or extensive real-client projects that are likely to overload students with technical issues [5, 18]. Thus, there is a strong need to achieve an ideal balance between realism and abstraction.

Flener [10] presents some of the risks and provide recommendations for teachers willing to use this approach. Common issues include i) ensure that students have the prior knowledge needed, ii) enforce a limit on the size of groups and projects, and iii) align the project activities to the LOs.

Moreover, some of the issues in the implementation of PMC are likely due to an inconsistent relationship to the course's learning outcomes [4], and an inappropriate degree of supervision by the facilitators [14]. Thus, there is an explicit need to align the LOs to these pedagogical aspects.

2.3 Game-based Learning

Serious games have been used for teaching and learning in SE programs, in particular topics such as project management [20, 21]. According to the reviews, this gaming approach contributes to motivation and learning due to the relevance of the content presented into the game. However, the validation of these game strategies is mostly obtained through the students' perception. Thus we can not assume a significant impact on the learning performance.

Baker, Navarro and Hoek [1, 2] proposed an educational card game that simulates a software engineering process. Problems and programmers exemplifies some of the phenomena occurring in the real-world through a high-degree of abstraction. It fits the limited time constraints of a course session, thus allowing the students to play the game several times facing different challenges each time. It is a competitive game, thus poorly reflect the real-world environment in which project members are expected to work together.

RCAG, SimSE and SERPG are digital games that simulate the software development process [3, 12, 19]. The interface provides a comprehensive view of different aspects of the development process, e.g. stakeholders, resources, project schedule. However, students are limited to interact with the computer. In the real-world, we expect the project members to interact and make decisions based on collective reasoning.

A third alternative is an inwards-class game that integrates the course's teaching-learning activities (TLAs) and assessment tasks (ATs). Jaramil [15], Klassen & Willoughby [16] proposed a game as learning activity to teach the management aspects of software development. These games use analogies that do not represent real aspects of the software process, e.g. project planning

2.4 Legacy Games

All the games presented in the section above are intended to run in a single class session. Thus, this setup is not entirely aligned to the PBL approach, which requires the students: i) to search for candidate solutions to a problem, ii) to critically analyze the candidates in order to make a decision, iii) and finally to reflect on the impacts of such potential solution. These are required skills to be trained in Engineering education [6].

The newly forged legacy game concept [7] describes a board game mechanic that is designed to change dynamically over the course of a series of sessions. It takes key elements from role-playing games, such as campaigns and storytelling. New game rules and contents can be introduced during the execution of the game, and similarly, old content may get overridden or removed. [11].

Therefore, a legacy game experience is likely to help integrating the concepts of PBL into a project management course. We opted to design such experience prototype and to pilot it in the context of a Software Engineering (SE) course at BTH.

3 RESEARCH PROCESS

3.1 Objective

In previous instances of a PMC, instead of a game, the students faced a real development task. This task was technically too difficult for them, as most of them lack deep development experience. Thus, through the development tasks, students sometimes did not achieve the intended learning outcomes, as they were too focused on solving the technical challenges.

Our main objective is, therefore, to propose a legacy game for a PMC that represent challenges of a real-life software development process. Students should handle tasks such as project planning, effort estimation, and manage unexpected situations and uncertainty. From a pedagogical perspective, by the end of the game we expect the students to demonstrate the knowledge acquired with this gaming experience.

3.2 Context

The game was designed as part of a PMC on a Software Engineering course, comprising of 7.5 European Credit Transfer System (ECTS). The overall goal of the course is to give the student basic knowledge of software engineering and the software development process, introducing the main phases of the development process, the different software development models and practices and its impact at product, process and organizational levels.

The course provides both theoretical knowledge and its application in practical situations. Theoretical knowledge is provided in a series of lectures covering themes such as requirements elicitation and management, testing, architecture design, project planning and project follow-up. The practical application requires the student to participate in the planning of a small project. This project is an interesting opportunity for designing and implementing the game.

We implemented a prototype legacy game during the period of Fall 2017. 9 students divided into two teams took part of the game. They were registered in a five years integrated engineering program (300 ECTS) in Industrial Economy[1], pursuing the specialization on Software Engineering and IT. All of them are native Swedish (although the teaching and the legacy game were conducted in English), between 22-27 years old, 1/3 female and 2/3 male.

3.3 Game Design

We designed a prototype for the legacy game to take part in the second half of the course once the theoretical contents had been covered. In this way, the game challenges matches the knowledge the students are going to be exposed to. The main teaching approach for the legacy game is problem-based learning. Each game turn takes part in a 2-to-4-hour workshop session and uses the students' assignment deliverables as input.

During the workshop sessions, the teachers also explain the game objectives and rules[2]. Teachers also present the project description and the goal of the game to all teams. The project description also includes the project resources (e.g. budget, workforce) and constraints (e.g. time to deliver, business rules).

3.4 Gameplay

Each turn comprises five steps:

1. Each team submit a project plan. The team should maintain a document describing a project plan in accordance with the learning objectives of the course. During the first round, the team submits an initial version of this plan. On each subsequent round, an updated version is submitted.

2. The teachers assess the project plan according to the learning objectives. If the students demonstrate a sounding project plan, that adheres to the good practices and matches the needs for the project, they are awarded bonuses for the upcoming game turn. Students can also be penalized if their project plan omits certain required development activities (e.g., testing) or practices (e.g. sprint retrospective). We have designed a set rubrics to assess both bonuses and penalties[3].

3. Each team rolls for uncertainty. During a workshop session, the game mainly consists of the students rolling dices that represent the uncertainties of a software development project. Teachers guide the students during the whole step, relating these uncertainties to real examples, e.g. a penalty in the implementation could be caused by a non-updated design or lack of requirements traceability. The uncertainty values are then added to the bonuses/penalties scored by the team's project. This result represents how much the actual process deviates from the original plan.

4. The teachers inform the teams about new events taking place. They represent changes that are likely to occur in a real-world software process, such as new requirements or resource limitations. Teachers play the role of customers or product owners, fostering students to negotiate some of the new challenges.

5. Each team updates their project plan. Finally, the teams are asked to report the changes to project plan according to the deviation and new events (Step 5). They are also suggested to make any improvements they think are needed for the success of the project.

Prior to the execution of the game, we defined a number of turns based on the course schedule. If there are turns remaining, a new turn begins. Otherwise, the game ends, and then the students are asked to provide their reflections on the project together with feedback about the gaming experience and lessons learning.

3.5 Execution

We have executed the game twice, for two different software development processes, i.e. plan-driven, and agile software development. The assignments and rubrics were slightly different in each case. During the plan-driven exercise, the students were expected to create a work breakdown structure (WBS) and detail the project schedule. The agile instance included different challenges, such as estimating based on story points by using planning poker, calculating velocity, managing technical debt, and planning releases. In the agile instance, the students had to make informed decisions about the different agile practices they adopted for their project.

It is important to note that the outcome of the game itself does not relate to the course evaluations. So, the students were not penalized if their project did not get any particular bonuses. In each session, lectures provide feedback to students, expecting that they could reflect on their choices and learn from the mistakes. The assessment task was based on a written report in which the students had to reflect about the pros and cons of each development model, as well as a *post-mortem* analysis of both projects, in which they had to reflect about what was the main cause of the eventual deviation to the original plan.

4 RESULTS

4.1 Students' Formative Assessment

During the game prototyping, the teachers conducted formative assessments to collect opinions from the students. Both verbal inquire and electronic forms were used. Eight out of the nine participants answered electronic form's open questions as follows:

1. What have you learned from the project planning game? The responses enumerate aspects such as: how to design a software project plan (mentioned by 4 students), how to manage the

[1] Civilingenjor i Industriell Ekonomi
[2] The rules are available to download at https://goo.gl/1oUvvB
[3] The rubrics are available as an appendix to the game rules in https://goo.gl/1oUvvB

deviations that can occur in a project due to uncertainties (2 mentions), trade-offs of the plan-driven method (1), project schedule (1) prioritize tasks (1), design tools, e.g. class diagrams (1).

2. Please describe positive aspects of the game. Students considered the game a fun experience (3 mentions), that mirrors real-world cases (2) and their uncertainty aspects (1). Other answers also mention the game as an alternative to theoretical lectures (1), and an approach to learning-by-doing, mainly by addressing the project mistakes (1).

3. Please describe negative aspects of the game. Students described a few difficulties, such as: A) deviations are presented too late during the game, make them hard to address; B) their project mistakes are likely to increase the game difficulty; and C) the game challenges are non-intuitive, thus difficult to relate to students with few or no practical experience.

4. Do you think the game represents the challenges of a real project? Most of the students answered that they could relate the game to real cases, although they acknowledged lack of experience with real projects. One participant answered that *"This is the most real exercise I have done so far"*. Another one mentioned that real-world projects are likely to be even more complex.

5. If you were attending this course next year, would you be willing to repeat the experience? Please justify your answer. All the students answered yes. Some of them made clear their expectations for further applications, e.g. A) more personalized / flexible project options; B) run the game in parallel with a "hands on" coding activities; and C) clear objectives by the project description.

These responses mainly indicate that students learned from and also enjoyed the experience. The results mostly describe personal experiences within the small group, and further investigations of the learning performance when applied in a larger group are needed.

When inquired by the teachers, the participants expressed their views verbally. They mentioned that the game mechanics were hard to understand when first presented, but this issue does not affect the course's learning outcomes. We hypothesize that this problem is mainly due a focused approach to present the rules, i.e. just the part of the rules[2] related to the current session was presented. This was an intentional design decision, since we wanted to avoid "defensive" planing if the students knew the rules beforehand, but also to allow students to experience uncertainty.

The students' feedback also suggest that short time was given between some of the workshop sessions. In particular, a series of 3 turns took place within a week (25, 27 and 29/Sept). This scenario is likely to hinder the gameplay, as it does not provide enough time for students update their project plan (Step 5). We plan to reschedule the course to provide one-week turns for the legacy game.

4.2 Teachers' Perception

After the game execution, the teachers met together to discuss the students' impressions and their performance about to the course objectives. This was aiming to collect the teachers' perceptions of using the game in the forthcoming course instances. The main results are:

For the proper execution of the game, it is necessary that the students produce and deliver the course assignments in time. A similar preparation was required from the teachers to assess the project plan deliverables before each session.

After a few turns of the game, teachers reported that the students start to improve their assignment's deliverables motivated by achieving a better score on the game task. Sometimes they were frustrated by not achieving bonuses, but overall the problem-solving aspect seems to foster them to review the course material looking for solutions.

The teachers acknowledge that some of the good results in some occasions might be caused purely by chance, not due to better or worse quality on their project plans i.e., the penalties were mitigated by good dice rolls. The teams also had the opportunities to experiment different alternatives. Moreover, during some of the workshop sessions, both teams worked side by side. This configuration introduced a competitive aspect, but also let students notice and learn from the others' decisions.

The teachers' collaboration during the workshop sessions were absolutely necessary. They provided feedback for the teams, related the game challenges with the theoretical knowledge, and ensured that the learning environment was fair and positive to all. The alignment sessions (peer assessing the project plans by two or more teachers), was also considered as strongly positive.

Finally, the game environment allowed teachers to discuss better with the students the results of their planning and decision-making. One of the teachers also mentioned that the students are using the knowledge they learned in the game in further course activities.

5 LESSONS LEARNED

A number of lessons learned were drawn from the game prototyping. They are derived from our observations during the game and a reflection about the students' assessment, as follows:

Identify the learning outcomes. The main objective of the game is to provide the students with a simulated environment that represent the challenges of a real project. This challenges, plus the project plan they are expected to maintain should be aligned to the LOs of the course. Teachers are also encouraged to identify threshold concepts and make sure that they take part in the project description.

Teachers' participation. While students act as players, teachers and lecturers take the role of the game moderator as well as customers/product owners. This includes tasks such as organize the game schedule, mediate questions regarding rules, and role-play the product owner. In fact, they controlled most of the aspects of the game, except the actions of the players/teams.

Course schedule. It is important that the workshop sessions match the number of turns the game will be carry out. Teachers have to adjust the time to deliver and project releases according to that, e.g. a two-month project ends after four two-week sprint which translates to four turns. There should be enough time for students to create/update the project plan (see Step 5, Section 3.4), as well as time for teachers to answer questions with regards to the game dynamics (as moderators) and to the project (as customer/product owners).

Theoretical knowledge. To promote fair gameplay, teachers should plan lectures and provide extra course material addressing theoretical knowledge related to the game challenges. In case of

the lectures, they can be scheduled before the game execution, or alongside the workshop sessions. It is also important that the course covers the required knowledge and templates for the different deliverables required for the project plans.

Students' teams. Different approaches can be used to assign players to teams, e.g. complete randomization, matched pairs, flexible groups. We suggest that the teams have similar size, preferably 4 to 5 students. It is recommended that the teams do not change during the gameplay.

Project description. Teachers should prepare a detailed document containing the information about the project and details about the project plan. Ideally, it should describe the desired features, the project resources and constraints (e.g. budget and time to deliver). This artifact should be presented according to the learning objectives for the course.

Students' project assessment. We strongly suggest using a set of rubrics to assess the project plans. It is important to detail to the students the bonus and penalties resulting from their project assessment. Therefore, they feel encouraged to improve the plan (see Step 4, Section 3.4) to achieve new bonuses and/or remove the penalties for the forthcoming turn.

New challenges during the gameplay. It is suggested that teachers provide an appropriate set of challenges considering the team's current situation during the game. The challenges should be scaled down to overcome potential frustration when the students are accumulating penalties or deviations to the planning.

The teams should also receive new resources to help them to achieve the newly presented challenges, e.g. by shortening the time to deliver, the product owner agree on hiring an extra developer for the team (extra workforce). This rule is intended to keep the balance of the game.

Ending the game. A successful project in the game should not be a measure of a student's performance and should not be related to the student's grade, and this should be communicated to the students. A well-thought and critical reflection, however, is a good opportunity for the teacher to collect feedback and provide the students meaningful insights regarding their gameplay.

6 CONCLUSIONS

In this paper, we reported the prototype of a legacy game aiming to represent the challenges of a software development project. The game is proposed as a pedagogical tool for software project management courses using a problem-based learning approach.

We piloted the game with students registered for a project management course. The participant's assessment regarding this novel approach was collected through verbal inquire and electronic survey, further compared to teachers' perception. The legacy game approach was perceived as a motivational environment for PMCs.

The most important contribution given is on a set of lessons learned from the prototype game execution. They are an important guide to improve the current game rules based on teachers' and students' opinion. Alternatively, these lessons are applicable in similar contexts, whenever teaching and gaming integration is done within a project course.

As future work, we plan to extend the game rules to integrate the game into a design-implement experience [24] in which students may evolve the project plan into a real product. Further applications are intended to validate the impacts of the proposed game in this particular context.

REFERENCES

[1] Alex Baker, Emily Oh Navarro, and André Van Der Hoek. 2003. Problems and Programmers: an educational software engineering card game. In *Proceedings of the 25th international Conference on Software Engineering*. IEEE Computer Society, 614–619.

[2] Alex Baker, Emily Oh Navarro, and Andre Van Der Hoek. 2005. An experimental card game for teaching software engineering processes. *Journal of Systems and Software* 75, 1-2 (2005), 3–16.

[3] Fabiane Barreto Vavassori Benitti and Jefferson Seide Molléri. 2008. Utilização de um RPG no ensino de gerenciamento e processo de desenvolvimento de software. In *WEI-Workshop sobre Educação em Computação*. 258–267.

[4] David Broman, Kristian Sandahl, and Mohamed Abu Baker. 2012. The company approach to software engineering project courses. *IEEE Transactions on Education* 55, 4 (2012), 445–452.

[5] Bernd Bruegge, John Cheng, and Mary Shaw. 1991. *A software engineering project course with a real client.* Technical Report. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst.

[6] Edward Crawley, Johan Malmqvist, Soren Ostlund, and Doris Brodeur. 2007. Rethinking engineering education. *The CDIO Approach* 302 (2007), 60–62.

[7] Rob Daviau. 2017. Legacy Games: From 'Risk' to 'Pandemic' to 'SeaFall' & Beyond. (2017). https://www.gdcvault.com/play/1024259/Legacy-Games-From-Risk-to Keynote at GDC 2017 - Game Developers Conference, San Francisco, California, US [Accessed: 2018 01 12].

[8] John Dewey. 1897. *My pedagogic creed.* Number 25. EL Kellogg & Company.

[9] Maja Elmgren and Ann-Sofie Henriksson. 2014. *Academic teaching.* Studentlitteratur.

[10] Pierre Flener. 2006. Realism in project-based software engineering courses: rewards, risks, and recommendations. In *International Symposium on Computer and Information Sciences*. Springer, 1031–1039.

[11] Board Game Geek. 2017. Glossary. (2017). https://boardgamegeek.com/wiki/page/Glossary#toc120 [Accessed: 2018 01 12].

[12] Thomas Hainey, Thomas M. Connolly, Mark Stansfield, and Elizabeth A. Boyle. 2011. Evaluation of a Game to Teach Requirements Collection and Analysis in Software Engineering at Tertiary Education Level. *Computers & Education* 56, 1 (2011), 21 – 35. http://miman.bib.bth.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=eric&AN=EJ902310&site=ehost-live

[13] Graham D Hendry, Miriam Frommer, and Richard A Walker. 1999. Constructivism and problem-based learning. *Journal of further and higher education* 23, 3 (1999), 369–371.

[14] Woei Hung. 2011. Theory to reality: A few issues in implementing problem-based learning. *Educational Technology Research and Development* 59, 4 (2011), 529–552.

[15] Carlos Mario Zapata Jaramillo. 2014. Teaching software development by means of a classroom game: The software development game. *Developments in Business Simulation and Experiential Learning* 36 (2014).

[16] Kenneth J Klassen and Keith A Willoughby. 2003. In-class simulation games: Assessing student learning. *Journal of Information Technology Education: Research* 2 (2003), 1–13.

[17] David A Kolb. 2014. *Experiential learning: Experience as the source of learning and development.* FT press.

[18] Supannika Koolmanojwong and Barry Boehm. 2009. Using software project courses to integrate education and research: An experience report. In *Software Engineering Education and Training, 2009. CSEET'09. 22nd Conference on*. IEEE, 26–33.

[19] Emily Oh Navarro and Andre Van Der Hoek. 2004. SimSE: an educational simulation game for teaching the Software engineering process. In *ACM SIGCSE Bulletin*, Vol. 36. ACM, 233–233.

[20] Giani Petri, Christiane Gresse von Wangenheim, and Adriano Ferreti Borgatto. 2017. Quality of games for teaching software engineering: an analysis of empirical evidences of digital and non-digital games. In *Proceedings of the 39th International Conference on Software Engineering: Software Engineering and Education Track*. IEEE Press, 150–159.

[21] André Raabe, Eliana Santos, Lauriana Paludo, and Fabiane Benitti. 2013. Serious Games Applied to Project Management Teaching. In *Enterprise Resource Planning: Concepts, Methodologies, Tools, and Applications*. IGI Global, 1427–1451.

[22] Paul Ralph. 2018. Re-imagining a Course in Software Project Management. In *Proceedings of 40th International Conference on Software Engineering: Software Engineering Education and Training Track, Gothenburg, Sweden, May 27-June 3, 2018 (ICSE-SEET'18)*.

[23] Henk G Schmidt. 1994. Problem-based learning: An introduction. *Instructional science* 22, 4 (1994), 247–250.

[24] PW Young and S Hallström. 2007. Design-implement experiences and engineering workspaces. In *Rethinking Engineering Education*. Springer, 102–129.