# Providing a Baseline in Software Process Improvement Education with Lego Scrum Simulations

Jan-Philipp Steghöfer
Department of Computer Science and Engineering
Chalmers | University of Gothenburg, Sweden
jan-philipp.steghofer@gu.se

## ABSTRACT

A critical aspect of software process education in general and software process improvement (SPI) education in particular is to give students the chance to experience processes and issues associated with process at first hand. This is, however, often difficult in an educational setting since providing a meaningful project in which to apply a process can take away time and focus from the intended learning objectives. Instead, miniatures and simulations can be used to create an environment in which students can interact with processes directly without taking up large parts of the curriculum.

In this paper, we report on our experience of using Lego Scrum simulations in an SPI course at the Bachelor level. The simulations are used both to introduce a baseline for the students to let them experience process issues directly, create an improvement plan that addresses observed issues, and to apply and evaluate the plan in a second simulation. This allows students to engage with SPI methods practically, instead of purely theoretically, and allows the teacher to refer to the shared experience throughout the course.

The collected data shows that the approach is suitable, but that students struggle with the demand of putting an improvement plan into practice. We show which issues commonly occur in the simulations and thus allow teachers who adopt the practice to scaffold it and react accordingly, in particular to empower the students to take on responsibility for the improvement of the process.

## CCS CONCEPTS

• **Social and professional topics** → **Computing education**; • **Software and its engineering** → *Software development process management*; • **Applied computing** → Education;

## KEYWORDS

Software Process Improvement, SPI, Software Engineering Education, Scrum

**ACM Reference Format:**
Jan-Philipp Steghöfer. 2018. Providing a Baseline in Software Process Improvement Education with Lego Scrum Simulations. In *ICSE-SEET'18: 40th International Conference on Software Engineering: Software Engineering Education and Training Track, May 27-June 3 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3183377.3183378

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*ICSE-SEET'18, May 27-June 3 2018, Gothenburg, Sweden*
© 2018 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
ACM ISBN 978-1-4503-5660-2/18/05…$15.00
https://doi.org/10.1145/3183377.3183378

## 1 CHALLENGES OF SOFTWARE PROCESS (IMPROVEMENT) EDUCATION

The main issue with teaching software processes and their improvement is that students have no or limited experience with working in a structured and organised way, do therefore not appreciate processes, and do not see a need to improve them. A culture that emphasises coding and a focus on the product exacerbates this [22]. However, industry not only demands that students are able to program, but also that they can work in teams within the confines of a development process, are able to communicate and adapt, and to continuously improve this process in their daily work [2].

It is therefore essential to equip students with knowledge and skills about software processes and software process improvement (SPI). However, it is difficult to relate the need for SPI to the experience of the students and to allow them to apply SPI methods in a practical setting. In courses in which learning objectives contain the higher levels of Bloom's taxonomy [16] (application, analysis, synthesis, and evaluation), it is necessary to provide the students with a meaningful baseline in which they can experience and analyse process issues and for which they are able to apply SPI techniques, potentially synthesising different ones into a coherent improvement plan and evaluating the different available techniques with respect to their merits for this baseline.

The contribution of this paper is a detailed description and analysis of using a simulation as a baseline and as a way to apply an improvement plan. In particular, we show the importance of anchoring SPI education in a practical example, but also discuss limitations of such an approach. We thus address the following questions:
**RQ 1:** Which real-life improvement issues are revealed by Lego Scrum simulations and how are they addressed by the students?
**RQ 2:** Which benefits and which limitations does the use of Lego Scrum simulations have when applied in a software process improvement course?

We are using Lego Scrum simulations in which students apply the Scrum methodology to build a Lego city [17]. Such simulations have shown promising results in software processes education (see, e.g., [19, 21]). We have applied the method described in this paper over a period of three years in three course instances with a total of more than 240 students. We have collected data from interviews with students, questionnaires, observations, and reflection reports written by the students. These data sources have been analysed to improve the way the course is taught and allow us to reason about the different course elements and their evolution.

The structure of the paper is as follows: Reported experiences with SPI education are discussed in Section 2. We then outline the course setup in Section 3. A short description of the research methodology is given in Section 4. Creation of the baseline using a Lego Scrum simulation is described in Section 5. The definition of an improvement plan in class is summarised in Section 6. How this plan is applied in a second simulaton is then discussed in Section 7. Based on the empirical data collected, we analyse our efforts in Section 8 and discuss our lessons learnt. We conclude with some advise to other teachers and an outlook on future work in Section 9.

## 2 RELATED WORK

The importance of educating students about the software engineering *process* and the related issues of *process improvement* has been acknowledged by many authors (see, e.g., [4, 14, 18]). We focus here on papers that specifically address the latter.

One of the earliest reported attempts to establish SPI in teaching is related to Personal Software Process (PSP) [11]. Also called a "self-improvement process" [12], it is based on the idea of continuous measurement and improvement and thus in the spirit of Scrum and other agile methods as used in our course. Students apply improvements directly to their own work, gaining practical experience as they go along. The simulations discussed in this paper serve a similar purpose, albeit in a more condensed form.

The use of a case study in an SPI course is advocated in [8]. Instead of observing process issues in a simulation, local IT industry was used to provide the baseline. In contrast to our approach, the case study is purely informative, but does not give the students a chance to apply SPI techniques themselves. On the other hand, the exposure to industry makes the relevance of the course very clear to the students.

Following a similar philosophy as ours — that practical work needs to be a part of SPI education — von Wangenheim and Hauck suggest to complement an SPI course with "practical course work within a simlated software project" [30]. Students are asked to assess a project and to define a software process based on theoretical lectures. A capstone project completes the learning sequence. The evaluation shows that the combination of theoretical and practical work is promising. However, such a setup requires commitment on the program scale and can not be achieved in single courses.

## 3 A BACHELOR-LEVEL SPI COURSE

The teaching approach described in this paper is general enough to be applicable in SPI courses in different stages of the education and with different scope. The particular course in which it was applied is a small course with 3 higher-education credit points and a total expected workload of 80 hours that are distributed over a 10 week span. It is taught to second year students and requires a successfully passed course on Software Processes as a pre-requisite. Relevant intended learning outcomes include:

- evaluate a project effort and suggest process improvements based on the evaluation;
- create a plan for such an improvement initiative;
- select between the different approaches, methods and frameworks that exist in the software process improvement field;

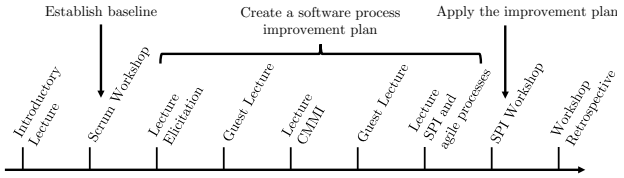


**Figure 1: Structure of the course. The different activities are distributed over 10 weeks.**

In order to achieve constructive alignment [3], it is thus necessary to give students access to a project effort and to scaffold the teaching so that it is possible for them to plan their improvement initiative using "the different approaches, methods and frameworks that exist in the software process improvement field". This will be the focus of this paper. To align the course with the intended learning outcomes, the course structure, shown in Figure 1, is designed to mimic an SPI effort in the real world following the four common SPI steps [29]:

(1) A process instance (the development of a product using a defined process) is first evaluated.
(2) Based on the evaluation, improvement needs are identified and an improvement plan is created.
(3) This improvement plan is then implemented in a second process instance.
(4) Finally the effects of the improvement are evaluated.

To address item 1), the students undergo a simulation of a software development effort. They then analyse the simulation and their experiences and derive improvement needs. To address item 2) these are used to define improvement goals and associated metrics using Goal/Question/Metric (GQM) [27]. The goals are used as the input to select relevant process areas, generic and specific goals and practices from CMMI [24] to define a CMMI roadmap [6]. This roadmap is then used to address item 3) and applied in a second simulation. The students are asked to take measurements using the metrics they defined and then analyse whether or not they achieved their improvement goals. Based on these measurements, item 4) is addressed with an evaluation of the improvement outcomes.

The course is examined with a written essay in which the students are asked to describe their experience in the first simulation, detail the improvement needs and the improvement plan they came up with, and report on the implementation of the plan in the second simulation. In addition, they are asked to describe the different theoretical frameworks of SPI, report on the guest lectures, and connect both course moments to their own experience.

## 4 RESEARCH METHODOLOGY

We follow the iterative approach of *action research* [15] to reflect on the situation in the course, plan course improvements, and act accordingly in each iteration [7]. Following Schön's reflective practitioner [20], this reflection-on-action takes place after the course based on collected empirical data, while reflection-in-action is used to react to emerging situations while the course is ongoing.

We have collected material about the course in three course instances in 2015, 2016, and 2017. This material includes extensive field notes by the teacher, student essays that constitute the examination, questionnaires the students were asked to fill in, and

interviews conducted with students by a researcher who is not a teacher in the course. The essays and interviews were analysed by the author who also acted as the instructor of the course. The questionnaires were analysed by the same researcher who conducted the interviews. To avoid bias, the results were discussed with colleagues familiar with the course and the subject area. We therefore cover the four lenses proposed by Brookfield [5]: autobiographical, theoretical, student, and peer.

While the course has been developed within these three years (e.g., by introducing flipped classroom elements), the basic approach using Lego Scrum simulations to establish a baseline, creating an improvement plan for the baseline, applying the plan in a second simulation, and evaluating the outcomes remained the same. Data was collected and analysed from the following sources:

*Teacher's notes.* The teacher kept detailed notes of most of the teaching moments, in particular of the simulations, for all three course instances. In one case (the first simulation of the 2016 iteration) these notes have been peer-reviewed by another teacher who was present, in other cases (the first simulation in both the 2015 and the 2017 iteration), the notes also contain the results of the discussions with other teaching personnel that was present and acted, e.g., as Product Owners. Emergent coding was used to identify recurring issues and their reasons.

*Student essays.* We used examination essays as a data source to determine the commonly reported process issues, how students addressed them in their improvement plan, how the students felt about the effectiveness of the simulations, and which impression of SPI they had after the course. For this purpose, a random selection of 20 essays per year was used. We thus selected 60 of a total of 184 essays handed in this period, including re-examinations. They were not filtered by grade, but only essays with at least 5 pages were used (the recommended length was five to six pages) to ensure a sufficient level of detail. In addition, only essays that describe the implementation of an improvement plan were used. Essays for the 2017 iteration were available in anonymised form.

The essays for all three iterations follow the same structure that includes a reflection of the experiences in the first simulation, a proposal for an improvement plan, and a description of the actual implementation of the improvement plan. These three sections were analysed by identifying all issues that were encountered by the students and the solutions that were proposed or tested. Additional reflections, e.g., of symptoms caused by the encountered issues, were also recorded. From the summary of the essays, interesting quotes about the improvement effort or the course in general were extracted. The extracted information was then coded with emergent codes which were later harmonised.

Students described the observed issues in the first simulation individually, but defined the improvement plan and the final implementation of that plan as a group. Thus, some issues reported by individual students have not been addressed in the group effort.

*Questionnaires.* Students in the 2017 iteration were asked to answer three questionnaires for a parallel evaluation of the flipped classroom element of the course that was spearheaded by a different researcher not involved in the course. At the beginning of the course, we checked for student attitudes and expectations; a mid-course questionnaire checked how these attitudes had developed and how the expectations were achieved in the course; and an end-of-course questionnaire gauged the students attitudes about the different learning moments after the last session. The questionnaires were not anonymous which allowed us to correlate the student attitudes and additional data collected such as the time spent on the course and participation in the different learning moments with the final grade of the students. The data was provided to the author in anonymised form.

*Interviews.* The same teacher who analysed the questionnaires also performed three interviews with students from the 2017 iteration. While the interview questions were mainly focused on the flipped classroom aspect of the course, they also contained specific questions about the simulations. Thus, the answers provide insights into the Lego Scrum simulations and their effectiveness. The interviews were analysed using emergent coding in several iterations until a stable set of codes appeared.

## 5 USING LEGO SCRUM SIMULATIONS AS A BASELINE FOR IMPROVEMENT

The students were asked to plan and conduct a development effort applying Scrum to build a Lego city in groups. The main goal of this activity was to provide them with first-hand experience about the issues that can appear in a development process and thus provide a *baseline* for an improvement initiative. All students underwent a similar simulation in their first term in the programme and applied Scrum in at least one project course since then. In comparison to the first simulation (that followed the outline in [21]), the students needed to structure the work themselves. As they are already familiar with Scrum, they should be able to define a schedule as well as a Definition of Done. They should also be familiar with the general setting of the simulation, with the fact that resources are limited, and that it is a team effort to build one city.

The simulation was structured in three phases: a short introduction by the teacher clarified the goals of the simulation, introduced the few rules that exist and allowed students to ask questions (20 minutes); the students then proceeded to plan the process and prepared for the first sprint (25 minutes); finally, the process was applied and the city was built in several sprints (2.5 hours).

The teacher had prepared the classroom so that the building resources sat in one corner of the room. Two tables at the front of the room served as the "integration area" where the increment of each sprint has to be placed for review. A whiteboard had been prepared as a product backlog with ca. 15 user stories, sorted by priority. The user stories are provided by the teacher but are the responsibility of a product owner (PO), usually a teaching assistant, who is available to all teams.

### 5.1 Process planning and constraints

The students were asked to split up into groups of six to eight, resulting in eight to ten groups. They were free to plan their sprints and the distribution of time within the sprint any way they wanted. They were only asked to stick to a general framework schedule that indicated when the first sprint should start and when the final review should start. In addition, they were provided with the following constraints:

- There must be one Scrum Master per team.
- Sprints must at least contain Sprint Planning, Sprint Review, and Sprint Retrospective.
- Breaks must be scheduled if desired.
- A Definition of Done (DoD) must be defined.
- Product must be reviewed several times.
- Schedule must be visible on the board.

During the 25 minutes for planning the schedule and defining the DoD, the students were first asked to reach an agreement in the groups and then combine the individual proposals to a final, common consensus. The process was conducted without interference by the teacher. In fact, the teacher emphasised several times that the students are responsible for defining the process and for the planning. As such, little scaffolding was provided for this activity.

The students were asked to record the problems they encountered throughout the simulation, including the planning session. These notes were part of what forms the baseline. After the initial planning, the teacher gave feedback on the schedule and the DoD, but left it to the students to make updates.

## 5.2 Observations from process planning

The teacher notes and student essays show that there are several issues the students struggle with when planning the process in all three iterations of the simulation.

*Inclusiveness of the planning activity.* Most groups select a Scrum Master rather quickly who is sent to discuss the planning with the other Scrum Masters. However, it is rare that the students who are left actually occupy themselves. Instead of seeing which resources are available or planning for how the group should work internally, they choose to busy themselves with unrelated activities.

Once the Scrum Masters feel that their planning is done, a second recurring issue is that the results are not communicated back to the groups and that there is no discussion of their impact on the groups. This means that students are not able to transform the schedule and the rules into actionable tasks once the sprint starts.

*Scheduling and rules.* The students leave the schedule rather vague. They define a sprint length and often schedule for a break, but do not prepare a more fine-grained breakdown of the different activities within the sprint (planning, implementation, review, retrospective). This causes at least the first sprint to be quite chaotic, in particular since little to no planning is done and students realise too late that they have not scheduled a sprint review. Furthermore, students do not define rules to structure their work. In particular, what is missing are rules that define how the groups select the user stories and how the resources are going to be shared. The latter causes some groups to hoard Lego pieces while others spend a lot of time looking for the bricks they need.

*Definition of Done.* Students struggle with defining clear criteria for when a user story is done, i.e., when it can be demoed to the Product Owner (PO). Items included in the DoD are either too general ("Requirement must be fulfilled") or include post-done activities ("PO must approve product"). It also does not include some important details (such as a consistent color scheme or that the sizes of all products should fit). This causes the students to present partial products to the PO that often have holes in them, are structurally not sound, and do not match in terms of size.

## 5.3 Observations from executing the process

The issues during the planning phase have a severe impact on the performance during the sprints. The lack of rules and the lack of coordination causes conflicts when picking user stories. No estimations of the user stories are performed and teams fail to define a velocity. This causes some teams to select much more than they can handle and others to select too few stories. In addition, it is not guaranteed that the Product Owner's top priorities are selected.

Responsibilities and procedures are unclear as well. In one of the iterations (2017), the Scrum Masters decided to establish a "design committee" to ensure that models fit together and to establish general guidelines. However, it was never defined which students were to form this committee or where this committee was supposed to meet. This caused delays and frustrations. In some cases, the students ask the teacher to address these issues, which is refused by the teacher in the spirit that the process is the students' responsibility.

Students also fail to keep time. In the 2016 iteration, the teacher enforced the start of the sprint review since the students had broken down the sprint into its respective phases and asked the teacher to run a timer. When the review started, none of the students took charge. Only about one third of the models were reviewed. As a consequence of the experience, the students reduced Sprint Planning to 5 minutes while extending the Sprint Review to 8 minutes. When asked if this change was a consensus, the students evaded the question.

In the 2017 iteration, a fine-grained breakdown of the sprint did not exist at all and only a timer for the entire sprint length was set. Therefore, the review did not start until some of the students realised that there were only two minutes left in the sprint. That caused the sprint review to be extremely rushed and some models to not be evaluated by the Product Owner. Once the review was over, the students again rushed into the implementation without performing a retrospective or planning the next sprint.

In later sprints, the issues change slightly. Inter-team communication becomes problematic. In the 2016 iteration, one team disassembled another team's model and used the building material for their own purposes. In 2017, one of the teams started the sprint review with the Product Owner on time. However, not all teams were aware of this and students kept pouring in with models they wanted to have reviewed. Resource management also becomes a problem, in particular since some teams were hoarding resources or the models built in the first sprints were using up too many bricks. Estimations of the work load are, however, still an issue: in all iterations, students neglected to estimate user stories. As a result, many students were idle for the final minutes of the sprint.

In general, the Definition of Done and the schedule were not updated with new information from the sprints and the reviews. The statement of the Product Owner, e.g., that each of the models should have a roof and the walls should only have strategically placed holes for windows was never recorded in the DoD.

It is unavoidable that the simulations differ slightly from year to year. In the 2016 iteration, e.g., the Product Owner pointed out that students took low priority user stories. Consequently, this was

reported by many as a problem. In the 2017 iteration, the teacher even decided to skip the third sprint to give students a chance to retrospect. The process up to that point was extremely chaotic and due to very poor time management, students had not conducted any retrospectives up to that point. This threatened the success of the simulation and the teacher was forced to intervene.

## 6 CONSTRUCTING AN IMPROVEMENT PLAN

The improvement plan itself is constructed in three in-class sessions. Due to the relative short course, only selected SPI techniques can be used. If a course has more scheduled lectures, this could be extended or the plan could be elaborated further. The techniques themselves have been taught in different ways. In the first iteration of the course (2015), the teacher gave lectures and the students were asked to apply the techniques as homework. This proved to be problematic because the few students who actually attempted the homework got stuck early on (e.g., when trying to understand the operative layer of GQM) and could not proceed since the feedback of the teacher was too delayed. This was one of the reasons why the flipped classroom model was adopted for the second iteration of the course (2016). The techniques were now explained with videos that the students could watch at home and the time in the classroom was reserved for applying the techniques, allowing the teacher to give formative feedback directly when necessary and guide the students through the process. This also opened up possibilities for peer assessment [28]. The model was refined in the third iteration (2017) with additional quizzes the students could use to gauge whether they understood the material correctly and preparatory assignments that the students could use to prepare the classrooms sessions at home.

Students are invited to send in the milestones at the end of each step for feedback to the teacher. The final goal is that each group has an actionable improvement plan that can be coordinated with the other groups and enacted directly in the second simulation.

### 6.1 Eliciting needs and defining goals

The goal of the first in-class event is that each team has clearly defined goals, questions, and metrics that describe their improvement needs and how to measure if the improvement was successful. Therefore, the focus is on analysing the experience from the simulation, eliciting improvement needs, and deriving the improvement goals from the collected data. The students are familiarised with interview techniques, techniques to identify and select stakeholders and artifacts, and with analysis techniques for the information they collect. Goal/Question/Metric (GQM) [27] is applied to derive improvement goals and metrics.

The groups are split in half. Each sub-group has 20 minutes to prepare interview questions to allow them to understand the issues that occurred during the simulation. Once the questions are ready, the groups are mixed so that sub-groups from different groups interview each other. The interviews last about 20 minutes, split into two 10 minute chunks so that each sub-group asks and answers the questions once. Afterwards, the original groups reconstitute, now with two sets of answers from two different other groups. They combine their data by identifying the three most pressing concerns in a short 10 minute brainstorming.

The rest of the session is dedicated to defining goals, questions and metrics according to GQM. The first step is to define the goals according to the GQM format (purpose – issue – object – viewpoint). Based on these goals, the students then define questions that help them to understand whether the goals have been achieved. The final step is to define metrics that allow answering these questions along with a plan for how the necessary data can be gathered.

### 6.2 Defining concrete improvement steps

The goal of the second in-class event is for each team to have defined a CMMI roadmap [6] that is specific to their improvement needs. CMMI [24], the Capability Maturity Model Integration, is a well-known prescriptive approach to software process improvement. It consists of a set of best practices, categorised by process areas that describe how a development process should ideally address certain issues. Examples for process areas are Project Monitoring and Control, Project Planning, and Requirements Management. Each process area has specific goals it addresses and specific practices it recommends. It is assigned to a maturity level that is used to characterise how well an organisation manages its processes. The examples mentioned here are all on maturity level two ("managed"), the first level above the initial one that represents an unmanaged process. To facilitate the selection, students use a quick reference as a guide [25]. Selected practices are summarised in a CMMI roadmap that also contains the improvement goals it addresses and a rationale why these particular practices were included.

Examples of roadmaps (see, e.g., [6]) are rather vague since they only contain process areas, but not the specific goals and which action to take. The students are asked to select suitable specific practices and refine them to an actionable set of steps for the second simulation. For instance, the process area "Project Planning" contains "SG 2 A project plan is established and maintained as the basis for managing the project", a specific goal often selected by the students. One of the specific practices is "SP 2.4 Plan for resources to perform the project". If the students pick such a practice, they are asked to concretise it by defining the resources at their disposal, their role in the project, and a tangible plan for their use.

### 6.3 Tailoring the improvement plan

There are two goals for the third in-class event: 1) have a refined CMMI roadmap including specific and generic goals and practices; 2) have a concrete plan for the second simulation, in particular when and how SPI techniques will be applied and measured. For this purpose, a connection between CMMI and agile practices is established in order to allow the students to target specific issues with Scrum they have encountered. This connection is based on an additional aspect of CMMI — generic goals and practices — and how they relate to agile development. In particular, the work in [23] reformulates the generic practices and shows how they are applicable in an agile process and thus in Scrum. Combined with the connection to the Scrum lifecycle established in [13], the students are then asked to select suitable generic goals and practices to improve their roadmap.

The second part of the in-class event is dedicated to refining the plan by making the practices actionable as discussed above, in

particular the generic practices selected earlier. Again, peer assessment is used along with group discussions. This has the positive side effect that the different groups are aware of at least some of the plans of the other groups. The teacher is again available for formative feedback while the students work.

### 6.4 Observations from the Construction of an Improvement Plan

A main issue is the fact that the different parts of the improvement plan build on top of each other. If a group misses one of the sessions, it becomes quite difficult to catch up. Likewise, if the students come to class unprepared (e.g., without having familiarised themselves with GQM or CMMI through the material made available online beforehand), they have a very hard time being productive and contributing to the improvement plan. This is one of the reasons for reduced attendance in the second simulation and some of the issues described there. On the other hand, this also means that most of the teams do not have an improvement plan that is concrete enough. This is one of the reasons why there is additional time scheduled to concretise the plan at the beginning of the third in-class event and in the second simulation (see below).

Another issue is that the improvement plan stays fairly vague in many cases. Even if students refine the specific and generic CMMI practices and are reminded by the teacher on several occasions that the steps need to be actionable, students still struggle to formulate improvement steps that are concrete enough. As an example, a student who identified intra-team communication as one of the issues formulates the most concrete action in the improvement plan as "During the sprint, make sure everyone is being updated on what went wrong so adjustment could be taken." Unfortunately, it remains unclear how this is going to be achieved.

## 7 APPLYING THE IMPROVEMENT PLAN

The second simulation follows a slightly different structure to allow the students to implement their improvement initiative. The simulation starts with a discussion of the improvement plans within the groups in which students have the opportunity to make sure that the steps they want to take are clearly understood and are actionable. This phase is followed by peer assessment where groups are paired and explain their improvement goals and strategy to each other. The result of the peer assessment can be taken into account when the final improvement plan is created. This initial phase takes up about 80 minutes, including the introduction by the teacher.

The second part of the simulation starts with the planning of the project during which the teams together define the schedule, the rules, and the Definition of Done as in the first simulation. This phase takes up 30 minutes. A total of 100 minutes are available for the building phase. A final discussion of about 15 minutes to reflect on the experience is scheduled for after the project is completed.

Instead of a Lego city, the students are asked to create a different product: a human habitat on Mars. The main criteria for review are that all modules are connected, the habitat is hermetically sealed, and as few bricks as possible are used. This changes the technical challenges since more focus is on the interface between the modules of the habitat. The main motivation for this shift is that the students can not only reproduce the city and apply lessons learned from there (e.g., that color scheme is important), but need to think about new technical challenges for the product as well, thus strengthening the likeness to an actual product development effort.

### 7.1 Observations in the Second Simulation

One of the major concerns in the second simulation is that some of the groups have an incomplete or even no improvement plan. This not only jeopardises the success of the students who want to participate without preparation, but also of the others. In the 2017 iteration, e.g., a student with a partial improvement plan that lacked the concrete improvement steps, took over a leadership role and started instating "improvements" that she felt were sensible. This overwrote some of the efforts of other groups that had based their improvement steps on a more solid theoretical footing. In addition, the peer assessment aspect of the preparation of the simulation is compromised if some groups are unable to provide substantial feedback. However, the peer assessment phase can also help the teams with incomplete plans to catch up and fit the ideas of other groups to their own improvement needs.

Some of the issues that were evident in the planning of the process in the first simulation are evident in the second simulation again. A typical problem is that the Scrum Masters start discussing the schedule and the DoD as well as improvement ideas, but the rest of the students have not found a good way to occupy their time. In the 2017 iteration, this was detected by the students themselves and they assigned some to start working with the requirements and clarifying them with the Product Owner. Unfortunately, however, these students did not give feedback to those that were defining the Definition of Done, so that crucial aspects were missing from the DoD. Only feedback from the teacher made them realise this and coordinate better. Likewise, confusion remained about when the sprint starts and who is taking care of time management. In the 2017 iteration, two timers were started with a 90 second delay.

In general, the second simulation is usually much calmer and less stressful for the students. This has to do with the reduced number of students, but also with the fact that many of the improvement ideas show beneficial results. In the 2015 and 2016 iterations, e.g., a focus was on working with requirements, estimating them properly, and adhering to the priorities defined by the Product Owner. In addition, the schedule for the different activities within a sprint was changed to allow for more time to plan. These changes resulted in better final products and less confusion about what the Product Owner actually required. Unfortunately, these benefits were not visible in the 2017 iteration. The students did not estimate user stories or their velocity and were confused about the DoD and the acceptance criteria for the stories, resulting in an unsatisfactory sprint review with only two out of eight user stories accepted by the Product Owner. It is unclear what caused this, but the general hectic and confused mood could have been a contributing factor.

The fact that fewer students are present during the second simulation has an influence on the pertinent issues. Fewer user stories are tackled and thus, resources are plentiful. Sprint reviews are less hectic since fewer products are being reviewed and less students are involved. Since less students require access to the Product Owner, this role is also no longer a bottleneck. The impact of the reduced number of students is also mentioned in some reports who attribute

at least some of the improved communication between teams and with the Product Owner on the smaller group size.

Unfortunately, it can be observed that students rarely take measurements, even though the improvement plans are supposed to contain metrics and a measurement plan. Measuring does also not seem to be a part of the sprint planning or retrospectives.The reason for this oversight might be a perceived lack of time, but the teacher emphasises several times in the in-class meetings that everything that needs to be done in the second simulation should be planned for accordingly in the sprint planning.

## 7.2 Evaluation of the improvement effort

The evaluation of the improvement effort should be done by the students in their own retrospectives. It is also part of the examination to reflect on the results of the second simulation and provide next improvement steps. However, in the 2017 iteration, a dedicated in-class session was devoted to the simulation in order to allow the students to reflect as a group and provide feedback from the teacher's perspective. The event was planned and announced before the simulation.

The lack of measurements from both simulations makes it difficult for many students to see quantitative improvements of their process. All groups usually lack measurements from the first simulation and thus have no basis to compare the new numbers to. However, improvements over the sprints of the second simulation are sometimes visible. In addition, the students have a qualitative notion of the success of the improvement.

The discussion about these topics is based on questions posed by the teacher, including "What were you actually able to improve?", "What were you provably able to improve?", "Were your improvement plans concrete enough?" and "Do you feel that everybody in the room was aware about what was going on?". These questions are designed to help the students reflect on their own behaviour and to start a discussion about the reasons for some of the problems that could still be observed. In a second step, the students are triggered to reflect about how they could have gone about establishing and implementing the process improvement plan differently.

# 8 EXPERIENCES AND LESSONS LEARNED

This section summarises the observations reported previously and combines them with the quantitative and qualitative data from our other sources to address the two research questions: **RQ 1:** *Which real-life improvement issues are revealed by Lego Scrum simulations and how are they addressed by the students?* discussed in Section 8.1 and Section 8.2 and **RQ 2:** *Which benefits and which limitations does the use of Lego Scrum simulations have when applied in a software process improvement course?* discussed in Section 8.3 to Section 8.5.

## 8.1 Process issues addressed by students

Student reports show that communication and planning are the two major problems. These two categories account for more than two thirds of the reported problems. Communication is part of the category of organisational issues while planning belongs to the project issues category [1]. Both aspects are major SPI problems that are addressed in industrial improvement efforts as well [1, 9]. A

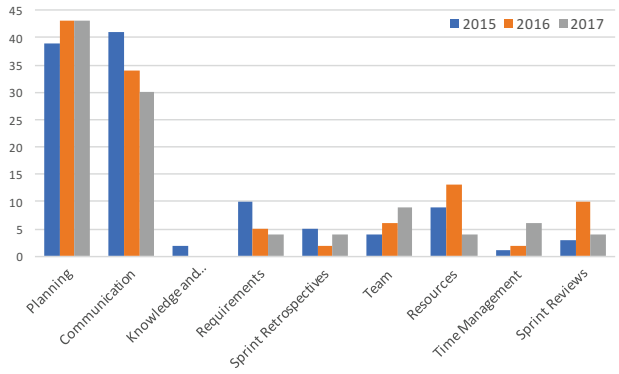


**Figure 2: Frequency of issues in different course iterations**

breakdown of the frequency of reported issues for the three course iterations can be found in Figure 2.

*Communication issues* can be categorised in three ways: 1) communication within the team 2) communication with other teams 3) communication with the Product Owner. The majority of issues reported are related to communication with other teams. Since the creation of the Lego city is a collaborative effort, the coordination between teams, e.g., to establish common standards, ensure that each user story is only built once, and ensure that dependent user stories work with each other (e.g., that the tractor fits into its garage), is essential. The first simulation often lacks this and many problems that appear during the first two sprint reviews can be attributed to communication problems. In addition, coordinating the use of the limited resources in later sprints also forces teams to communicate with each other better.

The students do realise that they are not able to *plan the development effort* effectively. During the first simulation, they are asked to structure the time available as they see fit and define a Definition of Done. Students put too little time on sprint planning, review, and retrospective in order to maximise the time available for building in each sprint. This might be a response to their experiences in the simulation in their first year where many students feel stressed due to the tight schedule imposed by the teachers. However, this issue can also be a learning opportunity if the students realise that their timing does not work (e.g., because not all products can be reviewed in time) and change their schedule accordingly. Unfortunately, this does not always happen. The definition of done is also problematic: the students often define vague terms or include erroneous items, such as "PO approved model during review". This makes it difficult for them to determine when their increment is ready for review.

Time and resource management, sprint reviews and retrospectives, and issues within the team beyond communication play a lesser role. It is possible that the communication and planning issues cover many of the problems that occur in other areas. On the other hand, a causal relationship might be at play as well: issues with requirements, e.g., manifest because the communication with the Product Owner is insufficient.

The incidence of the issues is very stable over the years. However, the focus shifts slightly, presumably due to communication within the class. Leadership, e.g., is an issue that has only been reported in the 2017 iteration (classified under team). Out of the 20 analysed

reports, six mentioned that leadership or management was missing. This has not been mentioned as a problem in any other iteration.

## 8.2 Quality of improvement plans

Unsurprisingly, most of the improvement ideas that students include in their plans are centred around communication issues and planning, with a focus on communication with other teams and the work with the user stories. Common suggestions are to introduce a Scrum of Scrums and to instate more principled task breakdown and estimation within the teams. Associated with the latter are changes in the schedule to allow more time for sprint planning. Depending on the concrete sequence of events in the first simulation, there can be specific issues that were encountered there that are addressed by the students. Since the sprint reviews were very chaotic in the 2016 iteration, e.g., steps to address this by imposing a specified process or by only sending the Scrum Masters are evident in the improvement plans for that year.

What is striking is that there is a substantial gap between the improvement plans, the theoretical foundations, and what actually happens in the second simulation. The first gap is mainly evident in the refinement of the CMMI practices to concrete, actionable steps. As discussed in Section 6.2, the students have a hard time making the leap from the relatively abstract practices that CMMI prescribes to concrete improvement steps that are applicable in the simulation. Even if the improvement steps are concrete and sensible, the connection to the specific goal from CMMI is often tenuous. One possible interpretation is that the students have the concrete improvement steps in mind and look for specific practices to map them to, rather than the other way around.

A confounding factor for the students and the effectiveness of their improvement plan is the fact that the SPI initiatives of the different teams do not need to be compatible with each other. A student from the 2015 iteration reports:

> "There was conflicting ideas from other teams SPI initiatives. Forced to follow standards implemented by others from majority decision, huge new problems were created. This includes global standards which were unachievable and ridiculous. The attempt to control recourses ended up creating a greater restriction." (Essay, 2015)

While according to the teacher's observations, the conflicts are not usually quite as drastic, it can not be denied that the different ideas sometimes clash and the process to establish the common schedule, standards, and process is not always inclusive.

## 8.3 Reported student experience

Students across all years repeatedly mention the *time pressure* in the second simulation as an impediment to the successful application of software process improvement methods.

> "The improvement goals were mostly not achieved, given that the context of the simulation, with its time limitations being too restricting." (Essay, 2017)

This is somewhat surprising, given that the students were aware of the time constraints and were tasked to prepare improvement plans that are concrete enough to be applicable in the simulation.

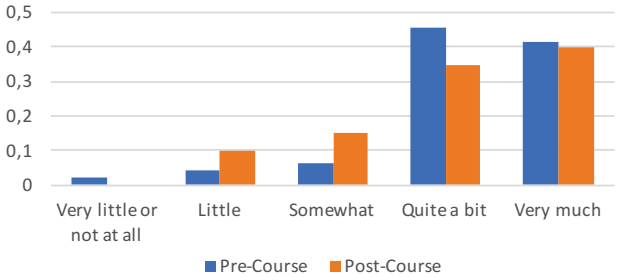


**Figure 3: Assessment of the relative expectation how much the simulations will enhance the students' learning (pre-course survey, n=46) and how much it actually enhanced students' learning (post-course survey, n=20).**

It also contradicts the fact that in the 2017 iteration, students felt the setup phase before the simulation was too long.

A major aspect that impacts the student experience is the *lack of preparation* for the second simulation during the course. As discussed in Section 7.1, the improvement plans of some groups are incomplete or not concrete enough, making it more difficult to join a concerted SPI effort. The students feel the impact of this poor preparation themselves. One of the interviewees reported the following:

> "But the second one was very stressful. Because of the problem of people not participating all the way through, suddenly showing up. Then also leaving in the middle of the simulation because they realize this is not working." (Interview #3, 2017)

When considering the overall experience in the SPI course, it is worth noting that the element of reflection in the examination seems to have a positive effect on the attitude of the students. Many students report in the summary of the report that they realised something valuable about software process improvement when writing up the report, e.g.:

> "During the second simulation I discovered how hard an SPI effort actually is to implement and the need for more rigorous planning and investigation into process issues and which process areas to focus on." (Essay, 2017)

When considering the expectations students had about the usefulness of the simulations, the data from the 2017 surveys shows that the teaching moments mostly fulfilled the students expectations (cf. Figure 3). The rather positive pre-course assessment might be due to the fact that the students already knew the simulation format and had positive experiences with it. The fact that a slight drop-off from the original expectation is visible indicates, however, that the factors discussed above might impact the perception of the learning of the students.

Interestingly, the expectation for the usefulness of the simulations does not correlate with the final grade. While the mean value is very high (4.05 on the 5-step Likert scale), students with a high expectation of usefulness do not necessarily fare better in the examination. In contrast, the expectation that the in-class learning activities are useful for student learning has a Pearson correlation of 0.5 that is statistically significant on the 0.05 level (2-tailed).

## 8.4 Student maturity and leadership

Students in general seem to have a very hard time to take responsibility for their way of working. They rely on the teacher to, e.g., keep the time or ask for permission to do certain things in the process (such as picking a new story from the product backlog when they are done with their work in the sprint). If they do take charge, it can be observed that there are several students who do that independently, leading to conflicts and confusion as described for the 2017 iteration in Section 7.1. A possible reason for this is that the students are not usually given this kind of freedom. Arguably, in most courses the guidelines are rather strict and controlled and enforced by a teacher or the process is not defined at all and the students thus do not follow one.

However, it is arguably precisely the freedom the students have in the simulation that causes the issues in the process to surface. In their essays, some students claim that their inexperience is the major cause of the observed process issues. While this might be the case, the simulations are the kind of environment in which experience can be created following the experimental model of work experience [10] and represent an environment that is closer to the reality students will find themselves in when transitioning into professional capacities.

Related to this, some students blame lack of leadership for the problems they encountered. The students mention in their essays that a leader would be able to impose a better structure and get people in line. Again, it becomes clear that students struggle with taking on responsibility:

> "Culture played a role in how everything was executed, since taking leadership is not something students often do in a classroom setting. It was quite awkward since no one really took command over the process, causing it to suffer." (Essay, 2017)

Some students address this perceived issue directly, by, e.g., suggesting to introduce a hierarchical team structure or a "leader for the Scrum of Scrums". This externalisation of responsibility is, however, no solution to the underlying problem that students have difficulties organising complex group work and structuring their tasks in a self-organising fashion.

## 8.5 Simulation design and teacher perspective

The first simulation is designed to force certain failures. The chaos that ensues when students are asked to structure the process is expected by the teacher and enforced by a weak scaffolding. The fact that the students identify the underlying causes for the resulting uncoordinated work, however, shows that the concept is viable. In particular when considering that the students went through one of these simulations before and applied Scrum in at least one project course, it is somewhat surprising that they are not doing better. They are taking over some lessons from that, but seem to have forgotten many aspects of structured work and demonstrate a lack of ability to self-organise.

One aspect that is also repeatedly mentioned by the students is the stress they feel in the simulations. Since the sprints are relatively short, students feel pressured and are prone to abandon the process aspects that they do not see directly contributing to the delivery of the product. This "product vs process" dilemma has been the subject of previous research [22] and is evident here again: planning activities and communication suffer first when students feel they need to use their time to the fullest. In contrast, however, students have the freedom to define sprint lengths, their velocities, and which stories they pick themselves. This should give them the ability to define a reasonable scope for each sprint that allows them to deliver the increment and have enough time for planning, review, and retrospective. Likewise, students also blame stress for problems in the second simulation, where they had the chance to instate changes to the process to reduce the stress level. Even though the teacher reminds the students, e.g., that breaks need to be a part of the schedule, the students opt to not include them.

Interestingly, the stress that is part of the design of the simulations could also be a detriment. Students in both the 2015 and 2017 iterations mention, e.g., that while the changes made to improve communication with other teams were successful, the stress of the simulation lets the team members forget to talk to each other, causing new issues that were not present in the first simulation. Stress and the overall conflict potential might also be one of the causes that students leave the event prematurely.

Overall, the main challenge for the teacher when using Lego Scrum simulations as a baseline in SPI education is to balance the fine line between giving the students freedom as well as the opportunity to fail and insufficient scaffolding. As discussed above, the former is an integral part of the learning experience. However, some aspects should be enforced by the teacher. In order to be able to reflect on their experiences, students should, e.g., collect the measurements they identified. Likewise, if the improvement plan is poor and does not work (as happened in the 2017 iteration), the teacher should interfere in order to give the students the chance to achieve the goals of the teaching activity. However, which concrete intervention is necessary or prudent in any situation must be left to the discretion to the teacher and her ability to reflect-in-action.

## 8.6 Threats to Validity

We discuss threats to the validity of our study and the methods used to minimize the threats, following the classification in [26]. Our main threat is *differential selection*, i.e., collecting and comparing data from different groups of students in the different iterations. This threat is closely associated with the *history* threat as data was collected at different points in time. We have addressed these issues by pointing out important differences and trying to attribute these differences to specific circumstances as recorded in the teacher's notes. The second most important threat is *contamination*. It is possible that the introduction of the flipped classroom in 2016, e.g., influenced the effectiveness of the simulations. We have tried to mitigate this effect by focusing on the simulations themselves and including any contaminating factors in the discussion.

Threats of *instrumentation*, i.e., influences of the data collection method, have been addressed by combining several data sources and triangulating the results. The use of examinations as a data source can have negative effects since students might write what they think the teacher wants to read. We have addressed this by using those parts of the examination that are free of valuation (reported issues in the simulations) and mostly critical remarks by the students. *Attrition*, i.e., the loss of participants while the

study was ongoing affects mostly the questionnaires used in the 2017 iteration. We attempted to obtain as many answers as possible within our available resources. A *Hawthorne effect* (i.e., participants perform better since they are given attention) was not observed.

Since *maturation* of the participants is an explicit goal of our teaching, it is not counted as a threat to validity. In contrast, we emphasise the maturation of the students and how the simulations contributed to it in our discussion. *Researcher bias* is not fully avoidable but has been mitigated by involving other researchers as observers and discussion partners. The effect of *testing* could have played a role in the questionnaires in the 2017 iteration. However, the questions were modified from the pre-course to the post-course survey and there was a gap of 10 weeks in-between.

## 9 CONCLUSION AND FUTURE WORK

This paper summarises our experience with using Lego Scrum simulations as a baseline for a software process improvement course. In answer to RQ 1, the simulations do produce relevant issues that are also visible in practice, even if they are mostly limited to planning and communication issues. The students respond mostly with common-sense approaches with tenuous connection to established SPI techniques, but do achieve substantial improvement in the second workshop. In response to RQ 2, benefits are the shared experience that allows eliciting improvement needs and creating a tailored improvement plan based on established SPI techniques. In terms of limitations, the deliberate progression from one teaching moment to the next poses strong requirements on continuous student engagement. In addition, the focus on planning and communication issues conceals some of the more interesting concerns in the process, e.g., resource and requirements management.

Stress associated with the short iterations in the simulation causes students to abandon process principles and — even after this has been pointed out to them — to not change their behaviour. This might be an effect of the way we teach in general: knowledge and skills are imparted without asking the students to create connections to their practical experience and to the relevance for their professional careers. Scaffolding is also an issue — how much should the teacher allow the students to explore and where should the teacher interfere?

While the simulations and the overall course they are embedded in will be further refined in future work, an alternative way of approaching SPI would be to embed it as learning sequences in project courses, similar to [30]. Students could be equipped with SPI tools in one of their first terms and then asked to repeatedly apply them as part of their reflection in the different projects. Instead of focusing all exposure to SPI methods and the associated reflection to a single course, it could become a general theme of a program with repeated application of the tools and support from a teacher.

## ACKNOWLEDGMENTS

The author would like to thank Christian Stöhr (Chalmers University of Technology) for his support in the analysis of the course, the conduction of the interviews with students, and the thorough analysis of the questionnaire data for the 2017 iteration of the course. The author is also thankful for Håkan Burden's (RISE Victoria) feedback on drafts of the paper.

## REFERENCES

[1] Sarah Beecham, Tracy Hall, and Austen Rainer. Software process improvement problems in twelve software companies: An empirical analysis. *Empirical software engineering*, 8(1):7–42, 2003.
[2] Andrew Begel and Beth Simon. Struggles of new college graduates in their first software development job. In *ACM SIGCSE Bulletin*, volume 40, pages 226–230. ACM, 2008.
[3] John Biggs. Enhancing teaching through constructive alignment. *Higher Education*, 32(3):347–364, 1996.
[4] Barry Boehm, Alexander Egyed, Dan Port, Archita Shah, Julie Kwan, and Ray Madachy. A stakeholder win–win approach to software engineering education. *Annals of Software Engineering*, 6(1):295–321, 1998.
[5] Stephen Brookfield. *Becoming a critically reflective teacher*. Jossey-Bass, San Francisco, 1995.
[6] Jan J. Cannegieter, Andre Heijstek, Ben Linders, and Rini van Solingen. CMMI Roadmaps. Technical Report CMU/SEI-2008-TN-010, Software Engineering Institute, CMU, Pittsburgh, 2008.
[7] Linda Dickens and Karen Watkins. Action Research: Rethinking Lewin. *Management Learning*, 2(30):127–140, 1999.
[8] Torgeir Dingsøyr, M. Letizia Jaccheri, and Alf Inge Wang. Teaching software process improvement through a case study. *Computer Applications in Engineering Education*, 8(3-4):229–234, 2000.
[9] Tore Dybå. An empirical investigation of the key factors for success in software process improvement. *TSE*, 31(5):410–424, 2005.
[10] David Guile and Toni Griffiths. Learning through work experience. *Journal of Education and Work*, 14(1):113–131, 2001.
[11] G. W. Hislop. Teaching process improvement in a graduate software engineering course. In *Frontiers in Education Conference (FIE)*, volume 1, Nov 1999.
[12] Watts Humphrey. *PSP: A Self-Improvement Process for Software Engineers*. Addison-Wesley Professional, 2005.
[13] Carsten Ruseng Jakobsen and Kent Aaron Johnson. Mature agile with a twist of cmmi. In *Agile, 2008. AGILE'08. Conference*, pages 212–217. IEEE, 2008.
[14] Mehdi Jazayeri. The education of a software engineer. In *ASE '04*, Washington, DC, USA, 2004. IEEE CS.
[15] David Kember and Lyn Gow. Action research as a form of staff development in higher education. *Higher Education*, 23(3):297–310, 1992.
[16] David R Krathwohl. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218, 2002.
[17] Alexey Krivitsky. Scrum simulation with lego bricks, October 2011.
[18] Emily Oh Navarro and André van der Hoek. Simse: An interactive simulation game for software engineering education. In *Computers and Advanced Technology in Education*, pages 12–17. ACTA Press, 2004.
[19] Maria Paasivaara, Ville Heikkilä, Casper Lassenius, and Towo Toivola. Teaching students scrum using lego blocks. ICSE Companion Proceedings, pages 382–391, New York, NY, USA, 2014. ACM.
[20] Donald A. Schön. *The Reflective Practitioner: How Professionals Think in Action*. Harper torchbooks. Basic Books, 1983.
[21] Jan-Philipp Steghöfer, Håkan Burden, Hiva Alahyari, and Dominik Haneberg. No silver brick: Opportunities and limitations of teaching scrum with lego workshops. *Journal of Systems and Software*, 131, September 2017.
[22] Jan-Philipp Steghöfer, Eric Knauss, Emil Alégroth, Imed Hammouda, Håkan Burden, and Morgan Ericsson. Teaching Agile – Addressing the Conflict Between Project Delivery and Application of Agile. In *ICSE Proceedings Companion*, ICSE '16, pages 303–312, New York, NY, USA, May 2016. ACM.
[23] J. Sutherland, C. R. Jakobsen, and K. Johnson. Scrum and CMMI Level 5: The Magic Potion for Code Warriors. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, pages 466–466, Jan 2008.
[24] CMMI Product Team. CMMI for Development, Version 1.3. Technical Report CMU/SEI-2010-TR-033, Software Engineering Institute, CMU, Pittsburgh, 2010.
[25] CMMI Product Team. Quick Reference Guide for CMMI® for Development. Technical report, CMMI Institute, Pittsburgh, PA, July 2015.
[26] Daniel R Tomal. *Action research for educators*. Rowman & Littlefield Publishers, 2010.
[27] Rini Van Solingen, Vic Basili, Gianluigi Caldiera, and H Dieter Rombach. Goal/Question/Metric (GQM) approach. *Encyclopedia of Software Engineering*, 2002.
[28] Marjo Van Zundert, Dominique Sluijsmans, and Jeroen Van Merriënboer. Effective peer assessment processes: Research findings and future directions. *Learning and Instruction*, 20(4):270–279, 2010.
[29] J.A. Calvo-Manzano Villalón, G. Cuevas Agustín, T. San Feliu Gilabert, A. De Amescua Seco, and M. Pérez Cota L. García Sánchez. Experiences in the application of software process improvement in smes. *Software Quality Journal*, 10:261–273, 2002.
[30] Christiane Gresse von Wangenheim and Jean Carlo R Hauck. Teaching software process improvement and assessment. In *EuroSPI 2010*, volume 99 of *Communications in Computer and Information Science*, pages 25–34. Springer, 2010.