A Multi-Level Didactical Approach to Build up Competencies in Requirements Engineering

Yvonne Sedelmaier, Dieter Landes
Faculty of Electrical Engineering and Informatics
University of Applied Sciences and Arts
96450 Coburg, Germany
{ yvonne.sedelmaier, dieter.landes }@hs-coburg.de

***Abstract*— Requirements engineering education at universities is a fairly difficult issue for various reasons. Among the most prominent causes is a lack of authenticity, i.e. too artificial settings that do not adequately mirror the complexity of real-world situations. We present an approach to requirements engineering education that tries to avoid some of these shortcomings, in particular by including requirements elicitation with real customers into an integrated didactic step-by-step approach. As it turns out, requirements engineering education is far more than assembling technical knowledge, but rather involves many non-technical skills that obtain a specific context-sensitive flavor in requirements engineering. Our didactic approach also addresses these skills, while resting on a sound pedagogical underpinning. Indications for the success of our approach are visible, e.g., in self-evaluations of the participants which are also summarized in the paper.**

***Index Terms*—requirements engineering, problem awareness, methodological skills, competencies, personal skills, real customers.**

I. Introduction

It is commonly accepted that requirements are top success factor in software engineering projects, but, conversely, also a major reason for project failures. Therefore, providing IT students with solid requirements engineering skills is of paramount importance. In practice, however, teaching and learning requirement engineering is not too easy. Part of the problem is the fact that good requirements are essential in complex real-life projects, but time and resource restrictions prohibit instructors from running many such projects – typically, there is only one such project during university education. Often, such a project comes late as a capstone project that ties together everything that should have been learned before. Unfortunately, learning requirements engineering only theoretically does not work well either. Students tend to view many important issues in requirements engineering as commonplaces and fail to see their importance.

It seems to be one of the big challenges for instructors to make requirements engineering education as descriptive as possible to make the matter more tangible for students. In particular, this encompasses mapping the complexity of real-world projects at least in part to a university context in such a way that the associated problems become evident for the intended audience.

In this contribution, we present the didactical approach that we developed for requirements engineering education at Coburg University of Applied Sciences. Core ingredients of our approach are a realistic and integrated setting, which includes writing a requirements document for a complex application and, as of late, eliciting requirements from real customers. In our specific setting, customers play a double role: in addition to simply providing requirements, they also act as external experts for communication issues. Another main characteristic of our approach is the extensive active involvement of students in the learning process. In particular the latter aspect has a solid theoretical underpinning in constructivist didactics. An additional characteristic of our approach is a strong emphasis on non-technical skills which are particularly relevant for requirements engineering, but also gain a very specific, context-sensitive shape in this particular domain.

The didactical approach that we conceived at Coburg seems to be quite successful since students value the importance of requirements engineering to a much higher extent and view themselves well equipped to deal with requirements engineering in practice. This finding is substantiated by a series of evaluations that we performed.

In the remainder of this paper, we will first analyze difficulties in teaching requirements in more detail before we characterize key features of our didactical approach and their pedagogical foundation. We discuss lessons learned both from the perspective of instructors and of students, before we give a short summary and an outlook to future work. Teaching Requirements Engineering

A. *Challenges in Teaching Requirements Engineering*

Although requirements engineering is a core ingredient of software engineering, it is fairly difficult to teach and to learn. Teaching and learning software engineering is generally restricted to small toy projects which mirror real world problems only to a limited extent. This is due to several reasons:

Early on, software engineering education often focuses on teaching and learning how to program a computer rather than

on requirements. Typically, programming assignments are small and clearly defined. Students design small pieces of software, often in small groups or even on their own. Assignments typically focus on a specific problem, e.g. specific element of the programming language such as arrays or loops, or particular algorithms. This means, software development assignments primarily focus on the technical aspects of programming and specific programming languages in a domain that is pretty familiar to students. As a consequence, requirements are supplied by the instructor, expressed clearly, and easy to understand since no unfamiliar terminology or unusual domain concepts are involved. This easily makes students believe that there is no need to bother with requirements since they generalize their programming experiences to real software development projects: There is no such thing like fuzzy requirements of stakeholders that use an unfamiliar terminology since students never came across such stakeholders. Consequently, there is a danger that students underrate the importance of requirements engineering since requirements engineering techniques do not solve any problem in their world. Often, students cannot even imagine problems that are rooted in insufficient requirements. And they do not believe instructors who report on their own practical experiences with what can go wrong with requirements. Students often think instructors exaggerate. Techniques they should learn in requirements engineering seem to be boring and useless since students mostly do not know why they need these techniques.

Furthermore, programming assignments tend to be isolated without relationship to other tasks. Even if there is more than one possible way to solve a problem the chosen approach will not have any consequences on following tasks. Students do not really need to balance reasons for or against alternative solutions. So, it would not matter if requirements were wrong or incomplete since students would not suffer from the consequences.

Even if, at a later stage, the focus shifts from programming to software engineering, and in particular to requirements engineering, the situation remains somewhat problematic: Due to time or capacity restrictions, the complexity of real world problems can hardly be reproduced in university education.

As one consequence, students usually do not perceive interdependences between requirements. They often suffer from the misconception that complexity scales up linearly. While a single use case is fairly easy to specify, dozens of use cases are not. If the number of requirements grows, so do the interdependencies between them.

In addition, students are in general given precise assignments and only need to apply known methods to solve a given problem. In university education students do not need to think about what the nature of the current problem actually might be. Students tend to take clear requirements for granted. For instance, it is quite easy for students to model a business process and extract use cases from it when the given example is very simple and clearly delimited. In "real" requirements engineering, requirements engineers first have to clarify the problem and then understand and solve it. They first have to elicit requirements before there is any point in thinking about technical solutions. Coming back to the example above, this means that there must be information on business processes in an organization before these processes can be modeled and taken as a basis for use cases. Frequently, this information is not readily available, but must rather be elicited from a range of appropriate stakeholders which frequently are not easy to identify. Students rarely face the problem of eliciting requirements from multiple groups of occasionally uncooperative stakeholders. Therefore, they do not see a problem in eliciting requirements.

All these challenges trace back to an insufficient match between scenarios in requirements engineering education and in real life. Given a restricted amount of time, it is quite difficult to expose students to examples which reflect real problems in requirements engineering. Requirements depend massively on the software that should be developed. Software engineering in university education mostly deals with developing a toy solution that will not be used in daily life. This also applies to requirements in university education: Requirements tend to be simple, and therefore requirements engineering seems to be unnecessary in students' opinions. Due to the lack of real customers students cannot imagine the complexity of and interrelationships between requirements within a large software engineering project.

B. *Didactical Approach in 2013*

At Coburg University, requirements engineering is a major issue in an elective course called "Software Modeling" which is offered in the second year of a bachelor program in informatics. Before enrolling into that course, students are required to take a compulsory introduction to software engineering in the preceding semester.

"Software Modeling" has been offered for several years and has been continuously evolving, including its didactical approach. The 2013 revision of the didactical approach aimed at improving students' understanding of requirements engineering and has been described in detail in [1].

For this approach we defined several intended learning outcomes in detail:

- Students shall acquire a more tangible impression of the term "requirements".
- Students shall understand the importance of requirements and shall be able to act accordingly.
- Students shall understand characteristic approaches to the specification of functional and non-functional requirements and their prioritization.
- Students shall understand the role of communication with other involved parties in requirements engineering.
- Students shall understand the role of business processes as a source of requirements.
- Students shall be able to collaboratively apply appropriate methods and notations in order so specify requirements for a sample software application.
- Students shall understand popular approaches to complexity and cost estimation for software systems.

Based upon the intended learning outcomes, the sequence of topics was restructured in order to focus on the given problem first before presenting solutions, relevant issues were illustrated on the basis of a continuous example, additional practical exercises were introduced, and predominantly passive learning settings shifted towards more active ones. The course emphasizes business process models as a source of requirements. Modeling processes puts software engineers in a position to extract requirements indirectly from an organizational workflow instead of, or in addition to, asking future users which functional and non-functional features the new software should exhibit.

This didactical approach includes the assignment to develop a requirements specification in teams of four or five students. To this end, students are exposed to a problem setting that they were sufficiently familiar with. For instance, the problem setting in 2013 was the derivation of requirements for a system to support the application, approval, and reimbursement for business trips. As a first step, students were required to develop business process models for the problem setting. The basic input for students consisted of an official leaflet which is used as a guideline for university members whenever they are about to go on a business trip. This brochure contains detailed rules for the application and reimbursement of a business trip and provides some details of how the process works. The teams of students extract distinct steps of the process before modeling them in a notation of their choice.

The business process models were subjected to a peer review. The process models of a peer group then served as a basis to extract use cases and fine-grained requirements.

Although this approach was successful from the perspectives of instructors as well as students, it still revealed some potential for improvement. Even though the assignment used a real world scenario, there is still no real customer from whom requirements must be elicited. And even though students obviously learned a lot in this course, not all teaching goals were completely achieved.

To sum up, in 2013 we applied several fundamental changes to our previous teaching approach in order to achieve our intended learning outcomes. Due to the fact that this new course design helped students to achieve these aforementioned goals, we decided to retain this didactical approach at large, and to refine it here and there.

So, in the 2014 iteration, we first refined our intended learning outcomes. So far, they were a little too abstract and we adapted the importance of some teaching goals again. While, for example, writing down given requirements is not the main focus any more, we now emphasize eliciting requirements form customers before writing them down.

C. *Intended Learning Outcomes*

The course "Software Modelling" aims at three main goals in addition to the existing intended learning outcomes:

- Students should understand the role and importance of requirements for their future careers. Students should develop problem awareness with respect to requirements engineering and recognize the importance of requirements and the difficulties in eliciting requirements. This teaching goal is assumed to be achieved if students are capable of eliciting requirements from future users, modeling business processes, and writing a requirements document.
- Students should enhance specific communication skills that are needed in requirements engineering. Students should be enabled to conduct a customer meeting in a goal-orientated way to elicit requirements. How can students elicit requirements which they did not "invent" themselves but are to be provided by a real customer? How can customers be prompted for information which may serve as a basis for requirements? How can requirements be documented and written down? How can students pass this challenge within a team (allocation of roles, etc.)?
- A third teaching goal is to strengthen self-reflection, self-organization, and self-responsibility of students. This is the basis for competence development [2].

As a consequence of the new prioritization of intended learning outcomes, a gap between them and the didactical design became evident so that some didactical fine adjustments became necessary.

II. Characteristics and Pedagogical Underpinning of a New Didactical Approach for Teaching Requirements Engineering

Based upon the experiences with the 2013 approach, we retained the structure of contents, activating learning elements, and a continuous example which culminates in writing a requirements document. Since active learning elements are commonly considered as a good approach for understanding abstract topics, we enhanced these aspects in the 2014 iteration. Students should play an active role in nearly every lesson instead of just listening to the instructor. During the lessons activation comes in by, e.g., small tasks that students need to deal with or by discussions between students and instructors.

One main weakness of the 2013 approach is the lack of eliciting requirements from a real future user or customer. So we refined our didactical setting mainly with respect to the following aspects.

A. *Eliciting Requirements from Real Customers*

One of the major drawbacks of our 2013 didactic approach was the fact that it did not address requirements elicitation. Several years back, we had tried to include this issue by having students elicit requirements from a peer team. Although this approach provided some insights with respect to difficulties of eliciting requirements, the whole setting was still artificial – students tended to be too cooperative in the role of a customer since they had no precise impression how real customers might act.

Therefore, we decided to bring in a real customer in 2014. Since we had chosen a system for managing offered training courses as application domain, we got in touch with a training provider in order to convince them to act as customers, a plan to which they happily agreed. We contacted a training and

consulting company with particular expertise in intra-project communication. This gave us an opportunity to include an additional aspect: Besides acting as a customer and reproducing typical behavioral patterns of customers in doing so, we had the chance to move to a meta-level right after the elicitation session. On this meta-level, the "customers", now in their role as communication experts, were to initiate a joint reflection with the student team on what had just happened in the elicitation session in terms of (un)successful communication.

In addition to being more realistic, students were expected to take the whole exercise more serious since they would not like to disgrace themselves in the face of externals. Furthermore, credibility was expected to increase since statements of external experts, based on their immediate practical experience, were deemed to have more weight than those of the instructor, who is latently alleged to exaggerate and, after more than ten years at university, to have lost immediate contact to what's happening in practice.

Students were split in two groups of approximately ten individuals and devoted a three-hour block for each team's elicitation session. About half of the session was planned for the actual elicitation of requirements from two customer representatives, and the other half, without the students knowing before, for an on-the-spot reflection of what went well and what did not. Students were asked to prepare for the elicitation meeting by pondering about good questions to ask, e.g. for identifying and clarifying business processes at the customers' site, and agree on an allocation of responsibilities and tasks within their team.

B. *Multi-level Teaching Approach*

When students enter this course, they already have some theoretical knowledge about specifying functional requirements through use cases [3].

We started the course with a first assignment that should be accomplished in teams of four students:

> *Exercise 1: Bidding for a software project*
>
> *A seminar provider intends to purchase a software system to manage his offered seminars. Imagine you as director of a software development company are asked to make an offer for such a software system.*
>
> *1. Think about your next four to five steps you would do, to prepare an offer. What would you do?*
>
> *2. How would you proceed? Give reasons why you decided for exactly this methods and approaches.*
>
> *3. Which problems might appear? What do you need to prepare that offer?*
>
> *Write down your results on a flipchart. (Working time: 30 minutes)*
>
> *Present your results in class.*

Students were supposed to take an active part in the course right from the start. This first exercise mainly aimed at raising awareness of requirements as an absolutely necessary prerequisite for bidding for a software project. Students should arrive at this insight by thinking about this exercise by themselves.

In a next step, students got an introduction to modeling business processes by using BPMN or event-driven process chains (EPCs).

Then students were split in two groups of, by and large, ten members each. Student teams were given a second assignment, namely they were supposed to elicit requirements from a real stakeholder, exchange their results, and build business process models on the information they received from the customer (see sec. III.A.). Process models were developed in a two-step approach: first, each team member developed an individual model before these individual models were merged and consolidated into a joint team model.

In the first exercise a lack of working techniques became evident. Therefore, we modified our second task by giving more precisely formulated briefings. For example, we added the following passage:

> *Exercise 2: Conduct a customer meeting*
>
> *[…]In preparation of the elicitation meeting with the customer, find an agreement on your intended course of action (among other things, your strategy to ask questions) and distribution of tasks. Clarify in the run-up the questions, you want to ask, the allocation of roles within your team, and the exchange of results at the end of the meeting. […]*

As an additional reaction to the two phases of the customer meeting (see sec. III.A.), which already included communication analyses on a meta-level, instructors decided to add a lecture session in order to further address communication and working techniques. In particular, this lesson put a focus on working techniques including allocation of roles and goal-orientation, approaches for preparing and conducting a customer meeting [4], question strategies, and communication techniques such as active listening [5]. This lesson was given in a pair-teaching format: the responsible instructor for this course with expertise in informatics acted jointly with an instructor with pedagogical background. As its main advantage, such a format offers the possibility to adapt and combine technical and non-technical knowledge and highlights inter-relationships between two disciplines to students. The customer meetings were analyzed again in a group discussion together with the students. Central questions were: "What went well? What would you do better next time?" Students realized by themselves that they should better prepare a meeting. Thus, they received information about structuring, preparing, and chairing a meeting. Furthermore, they learned about types of questions and question strategies to elicit needed information. This seems to be a good pedagogical approach because possible solutions are only presented after the need had actually arisen, i.e. students had already experienced a problem before they learned about possible solutions. Instead of teaching abstract and theoretical stockpiling knowledge, for which students typically do not know any use case, they could directly transform and apply the "newly acquired" knowledge.

As a preparation for the following session, students were also taught how to provide and to accept feedback, especially in a review process.

In parallel to the meta-analysis of the customer meeting, students got an assignment to model a business process in a notation of their choice. This task should be performed at home by each student individually. Following this, students should merge their individual business process models and derive a joint group model. The third exercise was to review their merged processes between teams of four or five students. To this end, they needed to remember and apply feedback rules. Without a-priori information about feedback and review processes students might feel accused and criticized.

Business process models are intended to serve as a source for requirements. Thus, students should now learn how to extract requirements from a business process model and write a requirements document. For this reason, a metaplan technique was used to activate students and collect contents of a requirements document as a first overview. Then several specific topics were worked out in class. During the following weeks, students were guided through several tasks which are necessary for writing a requirements document. Now that they know the context of single components they were gradually led to a complex document which contains all topics they learned before. Combining elements they develop over the time by themselves leads to a complete requirements document. Students had to work on individual and group exercises to repeat the learned contents in active work. Furthermore, they should apply theoretical learned knowledge and transform it into usable action knowledge.

In order to increase students' motivation, various exercises were associated with microcredits, i.e. a small bonus that may be used to improve the final grade in the exam.

C. *Pedagogical Underpinning*

There are indications that we learn

"- 10 % of what we read,
- 20 % of what we hear,
- 30 % of what we see,
- 50 % of what we hear and see,
- 70 % of what we say,
- 90 % of what we both say and do" [6] [7].

We choose this didactical multi-level approach to gradually build up students' competencies without overburdening them. Apparently, students are not used to structure their own working processes. With our approach we want to foster their self-organization and self-responsibility, and develop their communication skills and working techniques step by step. Students' previous knowledge and level of competence must be taken into account. This is a precondition to give students the possibility to further develop their competencies.

Designing an appropriate learning environment should be based upon constructivist principles. According to constructivist didactics, teachers act as coaches and can only give students room for their individual learning experience. Learning in this theory depends on the individual world and on the things a person learned before. Understanding arises from the interaction between the learner and the environment [8]. [8] conclude that "cognitive conflict or puzzlement is the stimulus for learning and determines the organization and nature of what is learned. […] Knowledge evolves through social negotiation and through the evaluation of the viability of individual understandings." It is necessary that the learner ties up his already existing knowledge and expertise to further develop it in his own way. Therefore, each student learns individual things according to his previous understandings, skills, and knowledge even if they experience the same learning situation.

Successful learning happens in learning situations which are adapted to students' previous skills and knowledge. Therefore, one of the main challenges in constructivist didactics lies in recognizing students' prior knowledge, then create appropriate learning environments, and adapt them specifically to the prior knowledge of students. In our didactical approach we gradually build up students' competencies by leading them through consecutive exercises, and strengthen their analytical skills as well as their self-organization by activating self-reflection processes.

Learning takes place when students consider the topics as relevant for their purposes [2]. As a consequence, they are interested in the issues and motivation for learning arises. Instead of teaching solutions for problems which students cannot even imagine, we make them see and understand the problems right at the beginning. After recognizing the problem they learn possible solutions to solve it and apply their new knowledge (learning by doing). In educational psychology these principles are main factors for successful learning [2].

III. Evaluation And Lessons Learned

A. *Instructors' Perspectives on Lessons Learned*

As instructors, we were surprised by the lack of students' work techniques we observed. Initially, we assumed that students had already exercised basic work techniques or basic communication skills at school. Yet, apparently this was not the case: The first given task turned out to be too complicated. This became evident during group work. While it was no problem for students to assemble in a group, they seemed to have severe difficulties to organize themselves within the group. Instructors expected that there would be a team leader, one student who writes down the results, one who presents them, one student as time keeper, etc. But teams started to work without structuring themselves, let alone assign roles to individuals. Even though instructors, at least from their perspective, provided a precisely formulated work assignment, results were fairly unstructured. Students read the assignment once at the beginning of the lesson, and then started to work without having a second look on the assignment. As a result, the results did not accurately fit the assignment. Teams should write down their final results on a flipchart and present them in class. Although students were advised to better use two sheets of a flipchart, some of them used both sides of a single sheet.

A severe lack of work techniques became also visible in the requirements elicitation session: students neither succeeded in allocating roles and tasks within their team, nor did they agree on question strategies before the meeting even though they

were provided with some advice what they were supposed to do. The provided hints were already a reaction to the perceived shortcomings in the first group assignments. As a consequence, we made our second exercise more precise and tried to give students more advice on what they could do to master the challenge. However, it was not enough and obviously did not help students at all. They were not able to prepare themselves for the meeting with the customers as we expected. Even though nearly all students were interested in participating in a customer meeting, some of them appeared completely unprepared. They neither had thought about possible questions they could ask, nor had they decided about team roles etc. All in all, these and several other observations led us to the assumption of lacking working techniques.

As a consequence, we added a teaching goal during the term that students should improve personal working techniques such as time management, endurance, self-organization, and structured course of action. Apparently, instructors' original intention to concentrate on fostering context-sensitive non-technical competencies in requirements engineering was too ambitious since prerequisites were missing.

Obviously, providing theoretical information about writing on a flipchart or organizing a team has no effect on students' learning processes. Rather, they need to experience some situations by themselves before learning becomes possible, including the possibility to make mistakes and learn from them. Apparently, students must reverse a flipchart sheet during the presentation of their work to recognize room for improvement. There is no point in telling them solutions before they experience the problem.

As instructors, we draw the conclusion to supply even clearer task assignments with very precise descriptions of what to do in future courses. Exercises should even be fairly fine-grained including precisely formulated steps what to do next.

Therefore, according to constructivist didactics, the third task was not simply "Write a requirements document". Instead of giving students a complex problem in one big chunk, we took our students by the hand and guided them through the process. The large task "requirements document" was partitioned into several smaller exercises, such as "Develop a use-case diagram", "Specify use cases", or "Derive functional requirements". Each week students got a new small task.

Adapting microdidactical elements during the lesson is based on the didactical principle of participant orientation ("Teilnehmerorientierung") [9] which is perfectly in line with constructivist didactics. [10] describes it as "reading" and „flexing". Reading means attentively observing students, while flexing concerns reacting on recognized requirements and needs. This generates an iterative process of adapting teaching and learning.

In constructivist didactical theory, teachers act as coaches for students and foster technical skills in combination with non-technical skills. Therefore, in future courses problem statements must be considered in more depth. It is necessary to work them out in more detail, and the nature of tasks in assignments needs to be well thought-out. Due to the fact that university cannot change students' previous knowledge and skills they bring into their studies, university teachers have to change their view on students' competencies and their learning processes.

Moreover, students often do not have any idea which methods and tools may help to elicit requirements from stakeholders. They do not know basic techniques to conduct a conversation which gives them needed information about processes in companies and the resulting requirements. During this course, instructors recognized that even if students knew in principle how to cope with the tasks, they looked helpless on it and had no idea what to do. Therefore, in addition to specifying assignments, instructors added a lesson to follow up on the customer meetings. Topics of this lesson were - in addition to methodological aspects - communication skills, such as questioning, and self-organization, such as preparing a meeting. As described above, pair teaching was chosen as didactical approach. In this lesson, students should get more action knowledge how they could master the given challenges. Course evaluation shows that students found this follow-up helpful. Several students appreciated this particular lesson when they were asked for things they considered necessary and important in an evaluation.

B. *Student Evaluation*

An intermediary evaluation of the course was conducted using the Software Engineering Competence Assessment Tool (SECAT) which was developed to evaluate students' competencies from multiple perspectives such as teachers, lecturers, or other students [11]. In this case, we used a self-estimation of students' competencies. SECAT also allows focusing on the assessment of one or more of nine criteria which are allocated to three levels of competence (see fig.1).

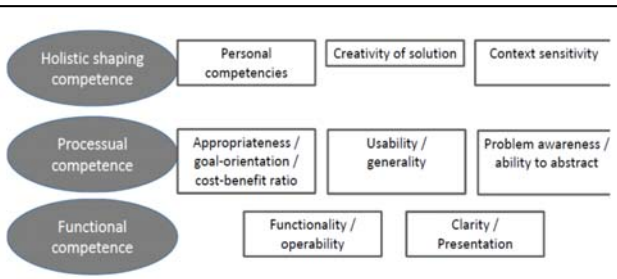

Fig. 1. Levels of competence in SECAT

Non-technical skills are high-level competencies in contrast to functional technical knowledge or the ability to present some content. In this case, we focus on problem awareness, context-sensitivity, and personal skills. Our teaching goals, namely improving problem awareness with respect to the importance of requirements engineering, fostering communication skills in context of customer meetings, and strengthening self-reflection as a basis of competence development are reflected in these three criteria. Each criterion was evaluated by means of 4 to 10

questions, according to the importance of the teaching goal (see tab. 1).

TABLE I. NUMBER OF SECAT QUESTIONS ACCORDING TO IMPORTANCE OF THE TEACHING GOAL

Criterion	Number of Questions
Problem awareness	4
Context sensitivity	10
Personal competencies	8
Creativity	2
Total	***24***

Each criterion is adapted to the specific situation and weighted by the number of questions per competence within the criterion. In this case, the main focus lies on context sensitivity which depicts in the competence "conducting a customer meeting". Table 2 shows the competencies which describe each criterion.

TABLE II. COMPETENCIES PER CRITERION

Criterion	Competencies
Problem awareness	Problem awareness / ability to abstract
Context sensitivity	• Moderation / Presentation • Conducting a customer meeting • Integrating in a team • Empathy • Endurance
Personal competencies	• Working techniques • Self-organization • Role allocation • Time management • Personal engagement • Goal orientation • Self-reflection
Creativity	• Creativity / Variety of methods

82 percent of a total of 20 students took part in the customer meeting, 88 percent modelled a business process on their own, and also 88 percent took part in the review process.

82 percent of our students find requirements engineering more interesting in comparison to the beginning of the term (see fig. 2). In the following figures, the left end of the scale means "Completely disagree", the right end means "Completely agree".

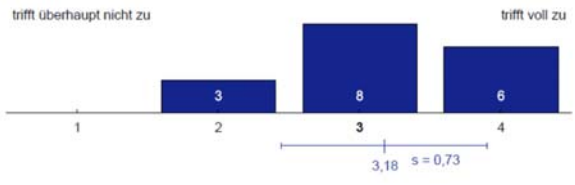

Fig. 2. Due to the course, I now view requirements elicitation more interesting than before the course

In last year's evaluation only 66 percent of our students agreed.

During the course, most of our students recognized the importance of requirements engineering for their future work (see fig. 3).

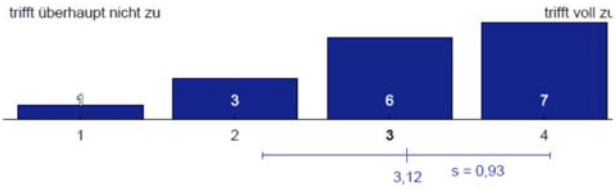

Fig. 3. Due to the course, I can now better appreciate the relevance of requirements for my work

As a result of the course nearly all students feel able to conduct a customer meeting for eliciting requirements (see fig. 4).

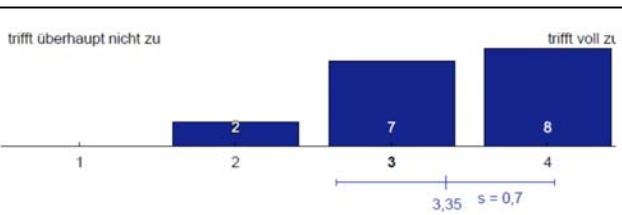

Fig. 4. Due to the course, I can now better conduct customer meetings.

Due to the course, students feel now able to reflect on situations and analyze them (see fig. 5).

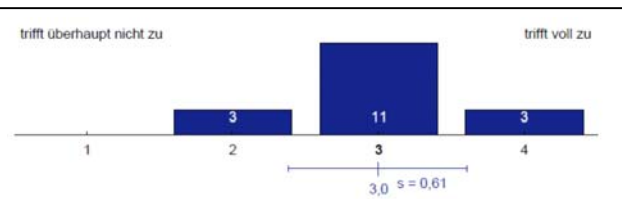

Fig. 5. Due to the course, I am now better equipped to analyze and understand specific conversations.

As a result of students' self-estimation with SECAT, competencies in the three main criteria, namely problem awareness with respect to the importance of requirements engineering, communication skills in customer meetings, and self-reflection, increased significantly (see fig. 6).

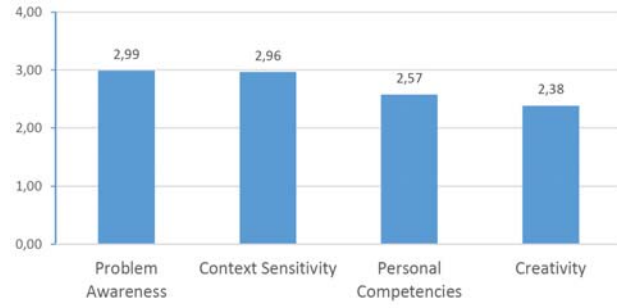

Fig. 6. Average increase of competence criteria

All in all, the evaluation showed a particular increase of competencies related to addressed intended learning outcomes (see fig.6 and fig. 7).

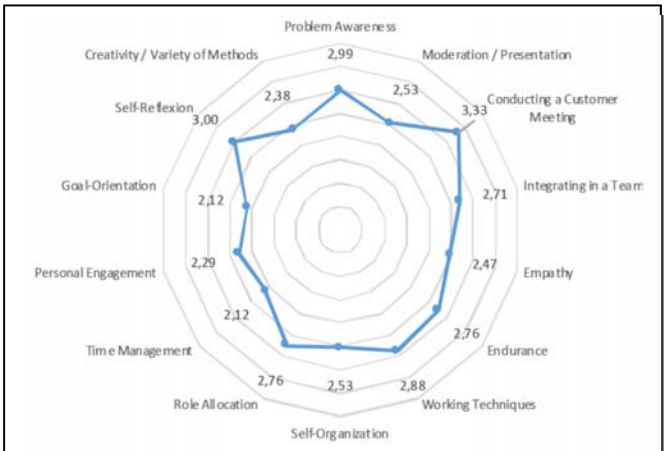

Fig. 7. Average increase of competencies over all students

Fig. 7 shows the largest increase of competence in "conducting a customer meeting", followed by "self-reflection" and "problem awareness". All values for these criteria are on a fairly high level of approximately 3 points.

Three out of 17 students (number 3, 5, and 10) did not take part in the customer meeting. Student nr. 10 with value 2.00 neither took part in the customer meeting, nor in the review of process models.

Evaluation results suggest that the chosen teaching approach allocated at constructivist didactics with consideration of psychological learning principles works well. Evaluation results indicate that the approach fosters students' competencies as explained in sec. II.C. Even intended learning outcomes which were added during the semester, such as working techniques, methodological skills, personal engagement, role allocation, or goal orientation, benefitted significantly. All students improved their competencies according to their self-estimation with values of at least 2 (see fig. 8).

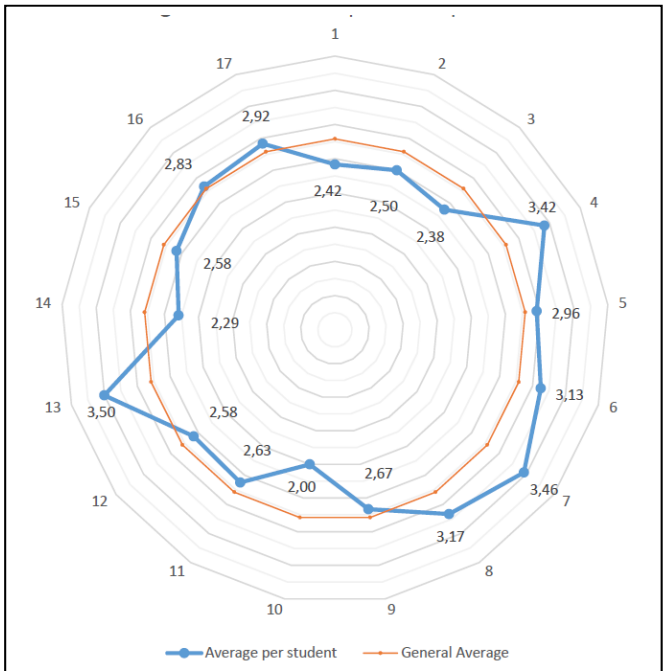

Fig. 8. Average increase of competencies per student

Also, even after being reluctant to be exposed to activating forms of learning, students seem to appreciate this format. In addition to statements in the evaluation which support this claim, this hypothesis is further substantiated by other indicators: 20 out of the 22 students who initially enrolled in the course actively participated in the course continuously, only 2 dropped out early. In addition, we had a regular physical attendance of 17 to 18 students in class throughout the complete semester, which is an unusually high rate. Since the course is an elective one without compulsory attendance, students would certainly have been scared away if they had not seen a real benefit in getting actively involved in the teaching and learning activities that we devised for the course.

IV. Summary and Outlook

We developed a didactical approach for requirements engineering education. Core ingredients of our approach are a realistic and integrated setting, which includes writing a requirements document for a complex application and, as of late, eliciting requirements from real customers. In our specific setting, customers play a double role: in addition to simply providing requirements, they also act as external experts for communication issues. Another main characteristic of our approach is the extensive active involvement of students in the learning process. In particular the latter aspect has a solid theoretical underpinning in constructivist didactics. An additional characteristic of our approach is a strong emphasis on non-technical skills which are particularly relevant for

requirements engineering, but also gain a very specific, context-sensitive shape in this particular domain.

Self-evaluations of participating students indicate significant increases in competencies that are relevant for requirements engineering and that we particularly targeted in the course. Currently, a final self-evaluation of students based at the end of the course is under way. In addition, we are just about to supplement and contrast the perspective of students with a SECAT-based evaluation from the instructor's perspective. Since the written examination associated with the course will be held shortly, we shall be in a position to correlate evaluation and examination results.

Although evaluation results so far indicate that the approach worked well, we still found potential for further enhancing our didactical approach.

It would be desirable to keep the meeting with a real customer on a regular basis for future courses. This seems to be the best way to make students understand the impact of requirements engineering. Unfortunately, organizational and financial difficulties have to be tackled before future students may be offered the opportunity for a real customer meeting. In a similar vein, it would be helpful if customers were not only available for an elicitation session, but also for, e.g., a review of business process models or requirements documents since this might uncover additional communication problems and expose potential for further competence development.

In future iterations of the course personal competencies such as working techniques and methodological skills should be taken into consideration right from the start. Instructors gained new insights into the level of basic skills of students. On this basis, they should adapt the didactical design to these additional intended learning outcomes, following the line of participant-orientation (see sec. IV). Our experiences indicate that university education must begin to foster basic skills at a much earlier point of time in bachelor programs.

Furthermore, it would be interesting to collect data from several cohorts of students. This would allow testing the hypothesis that this approach works well for similar groups of students.

ACKNOWLEDGMENT

We thank Ewa Sadowicz and Rainer Alt of EinfachStimmig, Nuremberg, for their active support.

The research project EVELIN is funded by the German Ministry of Education and Research (Bundesministerium für Bildung und Forschung) under grant no. 01PL12022A.

REFERENCES

[1] Y. Sedelmaier and D. Landes, "Using Business Process Models to Foster Competencies in Requirements Engineering," in Proc. 27th International Conference on Software Engineering Education and Training (CSEE&T), 2014, pp. 13–22.

[2] C. R. Rogers, *Freedom to learn:* A view of what education might become. Columbus, Ohio: Charles E. Merrill, 1969.

[3] A. Cockburn, *Writing effective use cases*. Boston: Addison-Wesley, 2001.

[4] J. W. Satzinger, R. B. Jackson, and S. D. Burd, *Introduction to systems analysis and design:* An agile, iterative approach, 6th ed. Mason, Ohio: Course Technology, 2012.

[5] U. Vigenschow, B. Schneider, and I. Meyrose, *Soft Skills für Softwareentwickler:* Fragetechniken, Konfliktmanagement, Kommunikationstypen und -modelle, 2nd ed. Heidelberg: dpunkt-Verlag, 2011.

[6] N. Green and K. Green, *Kooperatives Lernen im Klassenraum und im Kollegium:* Das Trainingsbuch, 3rd ed. Seelze-Velber: Kallmeyer, 2007.

[7] W. Niggemann, *Praxis der Erwachsenenbildung*. Freiburg: Herder, 1975.

[8] J. R. Savery and T. M. Duffy, "Problem Based Learning: An Instructional Model and Its Constructivist Framework," in *Constructivist learning environments: case studies in instructional design*, B. G. Wilson, Ed. 2nd ed, Englewood Cliffs N.J: Educational Technology Publications, 1998, pp. 135–148.

[9] U. Holm, *Teilnehmerorientierung als didaktisches Prinzip der Erwachsenenbildung - aktuelle Bedeutungsfacetten.* Available: http://www.die-bonn.de/doks/2012-teilnehmerorientierung-01.pdf (2014, May. 31).

[10] D. E. Hunt, "Lehreranpassung: 'Reading' und 'Flexing'," in Berichte, Materialien, Planungshilfen / Pädagogische Arbeitsstelle, Deutscher Volkshochschul-Verband, *Sensibilisierung für Lehrverhalten: Reaktionen auf D.E. Hunts „Teachers' adaption - 'reading' and 'flexing' to students"*, A. Claude, Ed, Frankfurt (Main): Pädag. Arbeitsstelle, Dt. Volkshochschul-Verb, 1986, pp. 9–18.

[11] Y. Sedelmaier and D. Landes, *A Multi-Perspective Framework for Evaluating Software Engineering Education by Assessing Students' Competencies*. In Proc. 44th Frontiers in Education Conference (FIE 2014), Madrid, Spain, to appear.