Teaching Goal Modeling to Engineering Professionals

An Experience Report

Marian Daun, Kevin Keller, and Jennifer Brings

paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen
Essen, Germany
{marian.daun, kevin.keller, jennifer.brings}@paluno.uni-due.de

Abstract. Model-based software engineering is a common means to cope with complexity, size, and safety-relevance of modern embedded systems. While conceptual modeling is often part of university curricula and the concepts of model-based engineering are also taught in industrial training, very few experience reports can be found on teaching model-based requirements engineering to industry professionals. In this paper, we report on our findings from teaching goal modeling with GRL as part of a model-based engineering course to industry professionals with no software engineering background. It showed that goal modeling is an appropriate means to cope with industrial problems, that teaching goal modeling can be used as gentle introduction into model-based software engineering, and that fundamental concepts of conceptual modeling are easy to understand for engineering professionals when taught in combination with goal modeling.

Keywords: model-based engineering, industrial training, goal modeling, GRL.

1 Introduction

Goal modeling approaches are of vital importance in model-based requirements engineering [1], [2]. Among others, goal modeling allows efficient and easy to understand structured documentation of high-level requirements [3]. Accordingly, goal modeling can be seen as starting point for continuous model-based engineering from requirements to code [4]. Furthermore, goal modeling approaches allow for a thorough analysis already in early phases [5], which is beneficial for efficient decision making [6] as well as safety analyses [7]. Although goal modeling is valued on a regular basis throughout literature (e.g., [8], [9]), it is rarely used in industrial practice for the development of embedded systems [10].

To improve this situation, it is often suggested to explicitly integrate goal modeling in universities' degree programs [11], which produces students capable to transfer goal modeling into industrial practice in the long run. Beyond that, there is, however, also a need to teach goal modeling in industrial training to achieve an immediate impact on the prevalence of goal modeling in industry. In this paper, we report on our experiences from teaching goal modeling with ITU Goal-oriented Requirements Language (GRL) [12] in an industrial setting.

This paper contributes an experience report discussing insights gained during teaching GRL as part of a model-based software engineering course to engineering professionals from a Germany-based internationally-operating large (i.e. >85,000 employees) company in the field of safety-critical embedded system development, who then applied there newly-learned skills to the development of one of their products. Participants were chosen by the company. They were engineering professionals, with no background in software engineering education and no prior knowledge of conceptual modeling. The use of GRL and the way of teaching goal modeling was seen as appropriate and was welcomed by engineering professionals. This paper focuses on the insights gained regarding the use of goal modeling in such a teaching setting, as these findings can lead to improving teaching model-based software engineering. In particular, we learned that teaching goal modeling should go beyond teaching how to create goal models themselves, but that furthermore, goal modeling can serve as perfect medium to (a) introduce model-based engineering and to (b) teach fundamental concepts of model-based engineering.

This paper is structured as follows: Section 2 provides some background information about the course and the role of goal modeling within the course. Section 3 provides details about the teaching approach to goal-modeling including its teaching goals, syllabus, and the teaching material used. In Section 4 we elaborate on the lessons learned from teaching goal modeling to engineering professionals. Section 5 relates our findings to related work and Section 6 concludes the paper.

2 Background

In this section, we will outline the overall course setup and the role of goal modeling within the educational setting. The course is meant to support a technology transfer project that aims at introducing continuous model-based engineering to the embedded systems industry. To this end, industry professionals learn how to create and read different kinds of models and how to ensure consistency across different models and abstraction layers. Overall the course comprises five modules aimed at different roles involved in the development ranging from requirements engineer to system architect, etc. Each module consists of about 25 blocks covering various requirements engineering, system design and quality assurance topics. While some blocks are part of multiple modules, each module focuses on the topics most relevant for that particular role. Goal modeling is of particular relevance for requirements engineers and system architects. The goal modeling blocks draw on our experience teaching i*-modeling to graduate students within an advanced requirements engineering course. To comply with industry's preference for using standardized languages we are using the GRL [12] instead. **Fig. 1** illustrates the topics covered in the requirements engineering module. This module consists of three parts: context analysis, goal- and scenario-based requirements engineering, and specification of requirements. While teaching how to read and create the respective models, the learners also receive instruction on theoretical foundations of context analysis, goal- and scenario-based requirements and specification of requirements.

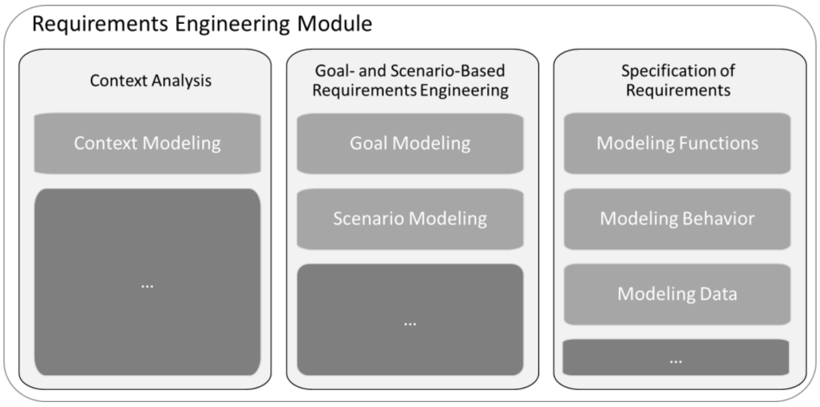

Fig. 1. Building blocks of the requirements engineering module

3 Teaching Goal Modeling

3.1 Teaching Goals

The goal modeling building blocks are designed to, first of all, teach how goal modeling is related to the overall approach (i.e. goal modeling as part of goal- and scenario-based requirements engineering). This is accompanied by the basics of teaching goal modeling, namely: how to read goal models and how to create goal models. The latter also emphasizes how to elicit goals, for instance, in cooperation with scenario-orientation, and how to identify and resolve conflicts within goal models. Based on the benefits of goal modeling (e.g., how to validate goal models, how to use goal models as reference for validating other artifacts), it is taught how different solution possibilities have different benefits and shortcomings depending on the models intended use and how readability of a goal model influences its perception.

3.2 Syllabus

The goal modeling blocks place emphasis on teaching the foundations of goal modeling with GRL as well as its implications for the use in model-based engineering of embedded system. First, the idea of intentional elements is taught. Participants learn to differentiate between goals, softgoals, tasks, resources, and beliefs. In particular, the use of beliefs to indicate why early design decisions have been made is taught (e.g., to indicate which laws result in which assumed measures). Hence, beliefs and resources are helpful when it comes to keeping track of consequences resulting from technological changes or changing laws and regulations.

Second, the use of decompositions is taught as well as the use of contributions. In doing so emphasis is placed on the different possibilities of structuring a goal model, e.g., in which situation a decomposition and when a contribution link seems more beneficial. Also special cases such as a subgoal which is a decomposed part of two supergoals is discussed, as this is not uncommon in the engineering of embedded systems, due to technological decisions taken by the original equipment manufacturer (OEM) in advance.

Third, the differentiation between different actors, their goals (or other intentional elements) and the dependencies between them, is part of the course. It is to mention, that in the engineering of embedded systems it has shown beneficial to not model stakeholders as actors [13], but the system under development and its neighboring systems (e.g., to document that the system under development relies on a context measurement, which is sensed by another system). However, the original concept of actor and dependency usage is also taught, so that participants can understand the underlying principle, and grasp why the modeling of systems as actors is beneficial in their case.

Last, rules for goal fulfilment and the propagation of fulfilments from subgoals to supergoals as defined by [12] are part of the course.

3.3 Teaching Material

The course mainly relies on online resources. The online materials comprise videos in classical lecture-style, textual learning materials as scripts, exercises and whiteboard-style videos discussing potential solutions and benefits of different solutions. Additionally, tool support and online tutorials for solving the exercises with the tool are given. Throughout the script and the lecture videos references to industrial practice are given. Exercises are specifically designed to give insight in realistic industrial problem situations. In addition, some FAQ videos also explicitly discuss typical industrial problem situations and different ways of handling conceptual modeling often found in industrial practice.

Concurrently to using the learning materials for self-study, the engineering professionals apply the newly-learned skills to the development to a real applications system. To support engineering professionals in applying the instructed material to the application system, bi-weekly conference calls and monthly full-day workshop meetings are conducted to discuss the engineering professionals’ questions about teaching material and their modeling of the application system.

4 Lessons Learned

From the introduction of the goal modeling teaching materials in the industrial setting, the feedback of participants, and, in particular, the discussions in the workshops on applying the technique to a concrete system, we gained several insights we will elaborate on in this section. While goal modeling in general was well received, questions, discussions and on-site teaching during the workshops showed that goal modeling can also be seen as advantageous teaching concept to introduce model-based software engineering (Subsection 4.1) and to teach fundamental concepts of model-based software engineering to engineering professionals (Subsection 4.2).

4.1 Goal modeling as introductory concept to model-based software engineering

Since goal modeling was easy to understand and perceived as useful by the engineering professionals, we assume that goal modeling itself can be a very useful introduction for teaching model-based software engineering in general. In this subsection, we summarize our major findings regarding the suitability as introductory concept.

Suitability of goal modeling as a starting point for teaching model-based engineering. We observed that participants were much quicker to grasp goal modeling concepts than other modeling languages taught in the course (context modeling, behavioral modeling, functional modeling, or architecture modeling). In particular, their initial draft of a goal model for their application system under development was of pretty good technical quality and discussions revealed that it adequately represented the intention of the system to be build. Discussions with the participants showed that the basic concepts of goal modeling with GRL - i.e. the use of intentional elements, decomposition, and contribution links – are seen as easy to understand and to apply to the application system. Hence, we consider teaching GRL and goal modeling as a good starting point for teaching model-based requirements engineering, In the original teaching material, however, we first introduced context modeling and then goal and scenario-based requirements engineering. As goal modeling has shown to be a good introduction to model-based engineering and to model-based requirements engineering in particular, the question arises from a pedagogical point, how to order the teaching blocks in a setup beginning with goal modeling.

Structuring of goal models. We gained the insight that teaching goal modeling to engineering professionals must emphasize the structuring of goals. As there is no hard and fast rule on how to structure goals and different goal models can perfectly describe the same system. Therefore, it is important to directly point out, what good structures might typical look like and what the different benefits associated with different structures are. E.g., the ability to abstract from particular subtrees to only discuss specific subtrees with stakeholders. Participants directly saw the benefit of these possibilities for their day-to-day work, when eliciting and negotiating requirements, but it must be stressed that this benefit of model-based engineering did not become apparent to the engineers until it was pointed out. Hence, teaching material should explicitly discuss advantages and disadvantages of different ways of structuring goal models.

It showed that the engineering professionals could easily transfer these benefits from other example systems to the application system. Hence, we assume that these things must not be taught explicitly using examples from the participants' day to day work experiences, as industry professionals are used to transfer concepts and have typically experience with different types of systems. However, benefits and even possible shortcomings modeling alternatives have for specific purposes must be taught as these are not directly seen. For instance, participating professionals started with a rather flat goal hierarchy. While they knew which goals belonged together and arranged them visually, they did not make use of subgoals to represent this structure.

4.2 Goal models as a point of reference for teaching more advanced software engineering concepts

From the discussions with the industry professionals we did not only learn that goal modeling is a good means to introduce model-based engineering to engineers, but furthermore to teach fundamental concepts in model-based software engineering. Hence, this subsection summarizes the insights we gained regarding the needs to teach further concepts using goal models as a point of reference:

Teaching how to cope with textual requirements. Even though industry professionals clearly saw the advantages of model-based development, they still have a need for documenting textual requirements (e.g., for negotiating legally-binding contracts) This creates the need to link the textual requirements to the model-based requirements. Instead of linking each model or parts of each model to textual requirements it was seen as beneficial to link textual requirements to goals and continue on completely model-based from there on. The use of goal models to link natural language requirements and model-based artifacts is exemplified by **Fig. 2**. The goals are used to transfer the textual requirements into the model-based world and subsequently keep them closely linked. Furthermore, the goal model then serves as main point of reference for traceability from natural language requirements to other requirements artifacts, such as scenarios, and to other engineering artifacts, such as the systems architecture.

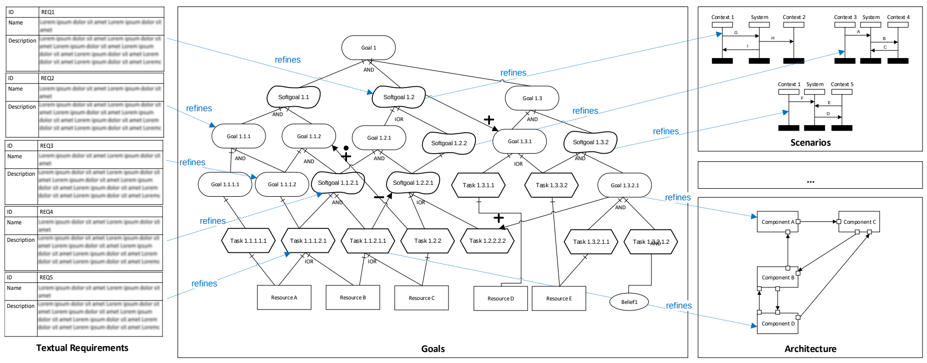

Fig. 2. Using goal models for establishing traceability

Teaching traceability using goal models. This approach inevitably leads to the necessity of teaching the use of traceability techniques, which is commonly seen as fundamental to model-based software engineering. Goal models are particularly suited for this because they already document several relationships and are easy to connect to scenarios which facilitates the illustration of traceability concepts. Furthermore, as goals enable easy assessment of what the system is capable of (i.e., which goals have been fulfilled) the benefits of using goal models as a kind of pivotal artifact.

Goal model as a pivotal artifact. As outlined, goal models were seen as the artifact to link with the textual requirements and to serve as point of reference for traceability concerns. Hence, leading to the need to keep the goal model constantly up to date. Therefore, there exists a strong likelihood for revisions and changes of the goal model. For example, the goal modeling needs to be restructured due to knowledge gains or design decisions made in later phases. This resulted initially in astonishment as typically textual requirements are not updated after the requirements phase and developed for every new project from scratch. However, participants easily gained an understanding for the benefits of keeping artifacts up to date and appreciated the iterative nature of model-based engineering. So also this concept should be more explicitly incorporated in future teaching materials. As another result of the necessary revisions, participants were stressing the need for adequate support for goal modeling in the development tool they are using.

Importance of unique identifiers and consistent identifier usage. To make use of fully automated methods but also to support communication with different stakeholders, identifiers should be kept consistent throughout the whole engineering but at least throughout the requirements engineering phase. This is another example, where participants totally agreed on the benefits and were easily convinced that this can significantly support their day-to-day work.

Communicating with stakeholders using goals. Participants were very intrigued to learn when to bring the stakeholders in, when doing model-based software engineering. Suggestions from an academic point of view as well as further discussions with the engineering professionals revealed that for this the use of goal models seems optimal, which is in accordance with common suggestions from literature [14]. However, it turned out that in industrial practice it seems advantageous to bring in the stakeholder at two points. First, to elicit and discuss initial requirements and to gain an understanding of the system, as desired by the OEM. At this point the use of stakeholders is typically minimized to avoid unnecessary costs and to not annoy the stakeholders with trivial questions. Second, stakeholders should be brought in for discussions and negotiations, when during development it becomes clear that a goal is not fulfillable. Participating industry professionals assumed that in many cases goals can then easily be changed so that they can be fulfilled. This results from the fact that requirements are formulated before it is totally clear what really is needed. For example, it might be defined that a certain response time is needed as this measurement is commonly used and seems to be a safe guess in the first place. When it becomes clear, that this is unfulfillable, discussions with the stakeholders, for example with the OEM, will take place to negotiate the real needed value what has not been done in the first place to accommodate stakeholders.

5 Related Work

There are several related experience reports about teaching goal modeling, which however, most report on experiences gained from teaching students not professionals. In this section, we will briefly introduce the main findings of these reports and compare them to our own findings, showing that there seems to be a significant difference between teaching goal modeling to software engineering students and to engineering professionals.

In [15] *Svee and Zdravkovic* report their experience from six years of teaching i* modeling to undergraduate and graduate students. The authors conclude that i* modeling is unpopular, especially with students with little experience. In contrast *Babar et al.* [16] state they experienced students to be appreciative of the i* framework.

Several papers report on various difficulties students have with certain model elements, e.g., dependencies [16], [17], SR internal relationships [18] and actors [16],[18]. Suggestions for helping students learn goal modeling include, the use of model skeletons and tables as intermediate artefact [18]. There is also a discussion on whether to start with SR or SD diagrams. While *Barbar et al.* [16] report that students found SD diagrams easier than SR diagrams, [19], [18] recommend an interactive approach. *Paja et al* [11] experienced that teaching, not only goal modeling but also goal-oriented analysis techniques, increases students understanding of goal models.

Bennaceur et al. [19] examined the correct application of a set of guidelines to help students with creating goal models and found that several guidelines were only rarely applied correctly by students. In [20] *Amyot* reports on the students' desires for smaller exercises including solutions to help them better asses their progress. In [16] the authors point out the need for realistic not too small examples to increase students' understanding. In [21] and [22] Nakamura et al. propose a role-play approach for teaching goal modeling (KAOS models) in university education.

In summary, recent studies came to sometimes quite contrasting results. For example; learning goal modeling was sometimes seen as easy, sometimes as hard; it was sometimes determined that simplified easy to understand examples contribute to teaching goal modeling sometimes the need for larger and realistic examples was reported. Our own findings from teaching goal modeling to engineering professionals showed that the basic concept was easy to grasp and there seems to be no reason for individualized examples in the teaching materials, which come from the participants' domain. As we outlined in Section 4, engineering professionals quite easily grasped the idea of goal models and were able to apply goal modeling to the application system. Therefore, there was no need for often suggested model skeletons or other guidelines in model creation. Hence, future work could cope with differences between students and professionals w.r.t. their needs in learning goal modeling and other model-based concepts, as has also been identified by other researchers for other areas of software engineering education (e.g. [23]).

6 Conclusion

In this paper, we reported on our experiences from teaching goal modeling with GRL to engineering professionals in the embedded industry. We identified that teaching goal modeling can be seen as gentle introduction to teaching conceptual modeling to engineering professionals without software engineering knowledge. In particular, goal modeling has shown to be suitable to teach fundamental concepts of model-based engineering such as traceability and relating natural language requirements to model-based artifacts. However, as we report from one example of teaching goal modeling with GRL to industry professionals, future work will have to investigate, how these findings can be incorporated into a sound teaching framework for model-based engineering of embedded systems. In addition, as comparison with related work has shown, it might be valuable to closer analyze different needs in learning goal modeling between student participants and industry professionals.

Acknowledgment. This research has partly been funded by the German federal ministry for education and research under grant no. 01IS15058C.

References

1. Lamsweerde, A. van, Letier, E.: From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering. In: Radical Innovations of Software and Systems Engineering in the Future. pp. 325–340. Springer, Berlin, Heidelberg (2004).
2. Mylopoulos, J.: Goal-oriented requirements engineering. In: 12th Asia-Pacific Software Engineering Conference (APSEC'05). p. Paper 1 (2005).
3. Kavakli, E., Loucopoulos, P.: Goal modeling in requirements engineering: Analysis and critique. Inf. Model. Methods Methodol. 102, (2005).
4. Daun, M., Tenbergen, B., Weyer, T.: Requirements Viewpoint. In: Pohl, K., Hönninger, H., Achatz, R., and Broy, M. (eds.) Model-Based Engineering of Embedded Systems, The SPES 2020 Methodology. pp. 51–68. Springer (2012).
5. Ponsard, C., Massonet, P., Molderez, J.F., Rifaut, A., Lamsweerde, A. van, Van, H.T.: Early verification and validation of mission critical systems. Form. Methods Syst. Des. 30, 233 (2007).
6. Boehm, B.W.: Software Engineering Economics. Prentice-Hall Advances in Comp, Englewood Cliffs, N.J (1981).
7. Cleland-Huang, J., Settimi, R., BenKhadra, O., Berezhanskaya, E., Christina, S.: Goal-centric Traceability for Managing Non-functional Requirements. In: Proceedings of the 27th International Conference on Software Engineering. pp. 362–371. ACM, New York, NY, USA (2005).
8. Lamsweerde, A. van: Goal-oriented requirements engineering: a guided tour. In: Proceedings Fifth IEEE International Symposium on Requirements Engineering. pp. 249–262 (2001).
9. Maiden, N.: What has requirements research ever done for us? (goal-modeling techniques). IEEE Softw. 22, 104–105 (2005).
10. Sikora, E., Tenbergen, B., Pohl, K.: Industry needs and research directions in requirements engineering for embedded systems. Requir. Eng. 17, 57–78 (2011).
11. Paja, E., Horkoff, J., Mylopoulos, J.: The Importance of Teaching Goal-Oriented Analysis Techniques: an Experience Report. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 37–42. CEUR-WS.org (2015).
12. International Telecommunication Union: User Requirements Notation (URN). (2012).

13. Pohl, K., Sikora, E.: COSMOD-RE: Supporting the Co-Design of Requirements and Architectural Artifacts. In: 15th IEEE International Requirements Engineering Conference (RE 2007). pp. 258–261 (2007).
14. Yu, E.S.-K.: Modelling Strategic Relationships for Process Reengineering, (1996).
15. Svee, E.-O., Zdravkovic, J.: iStar Instruction in Mixed Student Cohort Environments. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 19–24. CEUR-WS.org (2015).
16. Babar, Z., Nalchigar, S., Lessard, L., Horkoff, J., Yu, E.S.K.: Instructional Experiences with Modeling and Analysis using the i* Framework. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 31–36. CEUR-WS.org (2015).
17. Dalpiaz, F.: Teaching Goal Modeling in Undergraduate Education. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 1–6. CEUR-WS.org (2015).
18. Horkoff, J.: Observational Studies of new i* Users: Challenges and Recommendations. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 13–18. CEUR-WS.org (2015).
19. Bennaceur, A., Lockerbie, J., Horkoff, J.: On the Learnability of i*: Experiences from a New Teacher. In: Horkoff, J., Lockerbie, J., Franch, X., Yu, E.S.K., and Mylopoulos, J. (eds.) Proceedings of the 1st International iStar Teaching Workshop co-located with the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015), Stockholm, Sweden, June 9, 2015. pp. 43–48. CEUR-WS.org (2015).
20. Amyot, D.: Goal Modeling Education with GRL: Experience Report. In: Castro, J., Filho, G.A.C., and Liaskos, S. (eds.) Proceedings of the Eighth International i*Workshop, iStar 2015, in conjunction with the 23rd International Requirements Engineering Conference (RE 2015), Ottawa, Canada, August 24-25, 2015. pp. 1–6. CEUR-WS.org (2015).
21. Nakamura, T., Kai, U., Tachikawa, Y.: Requirements engineering education using expert system and role-play training. In: 2014 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE). pp. 375–382 (2014).
22. Nkamaura, T., Tachikawa, Y.: Requirements engineering education using role-play training. In: 2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE). pp. 231–238 (2016).
23. Wendt, K.D., Reily, K., Heimdahl, M.P.E.: First Steps towards Exporting Education: Software Engineering Education Delivered Online to Professionals. In: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET). pp. 241–245 (2016).