# Teaching Model-Based Systems Engineering for Industry 4.0: Student Challenges and Expectations

Arvid Butting, Sinem Konar, Bernhard Rumpe, Andreas Wortmann
Software Engineering, RWTH Aachen
Aachen, Germany
www.se-rwth.de

## ABSTRACT

Industry 4.0 and its international siblings envision revolutionizing manufacturing through integrating software-intensive systems from business plans to manufacturing systems to products and across the complete value-added chain. We observe a trend towards model-based systems engineering in the context of Industry 4.0, which requires finding a balance between modeling challenges and manufacturing challenges in educating software engineering students to become a vital part of this revolution. We conducted a project class on model-based systems engineering for Industry 4.0 to uncover the challenges in preparing the participating computer science students for playing a role contributing to the vision of Industry 4.0. To this end, we instrumented the class with a questionnaire and semi-structured interviews to understand the challenges and expectations of students on this topic. We report the results of both and lessons learned for future project classes on model-based systems engineering for Industry 4.0.

## CCS CONCEPTS

• **Applied computing** → **Education**; • **Software and its engineering** → *Architecture description languages*;

## KEYWORDS

Industry 4.0, MBSE, Project Class, Teaching Challenges

**ACM Reference Format:**
Arvid Butting, Sinem Konar, Bernhard Rumpe, Andreas Wortmann. 2018. Teaching Model-Based Systems Engineering for Industry 4.0: Student Challenges and Expectations. In *ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MODELS '18 Companion), October 14–19, 2018, Copenhagen, Denmark.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3270112.3270122

## 1 INTRODUCTION

Industry 4.0 envisions integrating automation systems with processes and stakeholders of the complete value-added chain to achieve highly customized mass-production [5]. A prime objective of Industry 4.0 research is the smart factory [8] in which interoperability, information transparency, and decentralized decision making aim to reduce unscheduled downtime to improve resource efficiency [6]. Model-based systems engineering (MBSE) is a key enabler for efficiently engineering the complex automation systems of systems of Industry 4.0 [22]. As such, applying MBSE to integrated automation systems is one of the challenges today's students will face as tomorrow's Industry 4.0 software engineers. Teaching MBSE for Industry 4.0 requires finding a balance between modeling, implementing, and systems engineering while facing educational challenges. Related challenges, such as teaching (1) agile methods [12, 19], (2) development for cyber-physical systems [14, 16], (3) model-based software engineering [9, 10, 15], (4) software architectures [7], or (5) through real-world projects [11, 20], are subject to individual investigation. The challenges of expectations from the perspective of students learning model-based techniques for Industry 4.0 – *i.e.*, smart manufacturing with software-intensive cyber-physical production systems (CPPS) – however, demand for an investigation to ensure teaching future's Industry 4.0 software experts properly.

To this effect, we conducted a project class on MBSE for Industry 4.0 investigating these challenges. In this 15-week class, students conceive, model, implement, and evaluate the software and (parts of) the hardware of a smart factory. For an exploratory investigation, we instrumented the class with a questionnaire and semi-structured interviews to understand the students' challenges and expectations on the topic. In detail, we report on investigating three main research questions:

- **RQ1** What are the technical and non-technical challenges faced by students applying agile MBSE to a smart factory in a project class?
- **RQ2** Which technical and non-technical competencies are expected to be learned or improved by students with reference to their future professional careers?
- **RQ3** Which teaching methods and supplementary materials are suitable for mastering technical and non-technical competencies in the context of an Industry 4.0 MBSE project class?

To investigate the latter, we employed interactive quizzes, design thinking, and problem-based learning.

In the following, Section 2 describes the project class and Section 3 details our study conducted with this class. Section 4 presents observations and Section 5 discusses its validity. Section 6 summarizes the lessons learned.

## 2 COURSE DESCRIPTION

We conducted our investigation within the project class "MBSE for Industry 4.0" at RWTH Aachen University. In this class, the students modeled, implemented, deployed, and evaluated the structure and behavior of a smart factory demonstrator consisting of various CPPS and mobile robots. Project classes are centrally organized


*MODELS '18 Companion, October 14–19, 2018, Copenhagen, Denmark*

ACM ISBN 978-1-4503-5965-8/18/10…$15.00
https://doi.org/10.1145/3270112.3270122

for students enrolled in various degree programs. Each computer science student must choose at least one of the offered project classes. The organizers advertise their classes and students apply for these. Consequently, only students interested in the topics enroll in the classes and our study design is influenced by this central organization: (1) the class has to respect university-wide teaching goals; (2) its duration is limited to 15 weeks; and (3) selection of participants was regulated. Moreover, project class and study are subject to teaching goals of our department and the supervisors. These include deploying specific modeling techniques and CPPS, including computer science bachelor (B.Sc.) and master (M.Sc.) students in the same class, and employing agile methods.

Based on their applications, we selected 20 students, 10 from each program. All students had a basic understanding of MBSE and Java from the mandatory B.Sc. software engineering course and other basic courses. CPPS are not subject of computer science curriculum and, consequently, the students had no experience with such systems. To mitigate this, we used production systems from the Fischertechnik[1] "Robotics in Industry" line and LEGO NXT robots. Both systems are suitable for education due to their robustness and ease of use. The course was organized in 15 weekly, mandatory project-wide meetings of two hours each. Additionally, students could access the prepared smart factory demonstrator on a daily basis. The class was aligned in three stages: (1) Preparation stage: Two weeks before the first meeting, we assigned a topic for an initial presentation to each student based on a vote where students could indicate two favored topics. In the first two meetings, each student gave an initial presentation about a technological topic relevant to the project class. The topics included developing software for Industry 4.0, multi-agent systems, Scrum, MBSE, Fischertechnik robots, the LeJOS NXT[2] middleware operating the LEGO NXT robots, and the MontiArc [3, 17] architecture description language. The LEGO NXT and MontiArc were known by some of the M.Sc. students. (2) In the subsequent MBSE introduction stage of two weeks, students were taught MBSE with MontiArc including its code generation for LEGO NXT robots. After successfully demonstrating their software architectures deployed to and running on the LEGO robots, we moved to the Industry 4.0 implementation stage. (3) In the remaining time, the students were split into teams of 5 and implemented the factory demonstrator. To this end, each team comprised B.Sc. and M.Sc. students and was led by a M.Sc. Scrum master. The four teams "product variability", "product finalization", "automated logistics", and "interaction" were responsible for developing parts of the overall demonstrator as depicted in Figure 1. To address cross-cutting concerns, *e.g.*, a joint communication protocol, responsible persons of the teams met regularly and delegated their decisions to the teams. In the final project class meeting, the students prepared a live demonstration of the developed smart factory demonstrator in action. The goal of the project class was to realize a smart factory demonstrator producing mass-customized yogurts, inspired by the MyJoghurt[3] demonstrator and based on 23 user stories of four different roles[4]. We only prescribed modeling

[1]Fischertechnik Robotics in Industry: www.fischertechnik.de/en/service/elearning/teaching/robotics-in-industry
[2]LeJOS Website: www.lejos.org
[3]The MyJoghurt Demonstrator: http://i40d.ais.mw.tum.de/index/industrie
[4]Available on the Companion Website: www.se-rwth.de/materials/mbse4ind

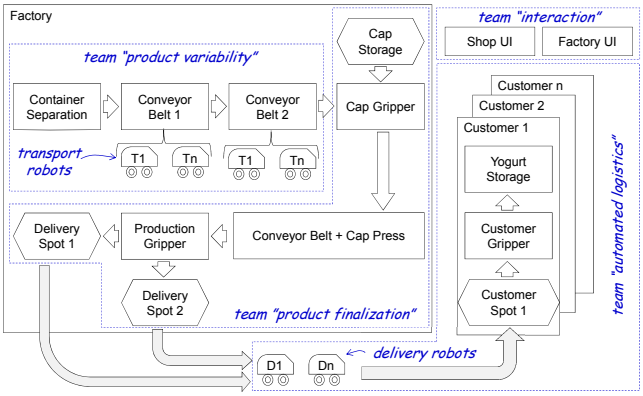


**Figure 1: Arrangement of the physical part of the smart factory demonstrator and separation into teams.**

the CPPS' architectures with MontiArc and implementing their behavior through embedded automata or in Java. The components of the yogurt factory's software architecture and the delivery systems are visualized in Figure 1. Physical locations are visualized as hexagons and LEGO robots are depicted as schematic vehicles. Production starts with an empty yogurt cup delivered by the container separation that is transported by a conveyor belt. A mobile filling robot can fill the cup with a yogurt flavor that matches the ordered product. Optionally, a second filling robot can deliver a second flavor. Afterward, a cap is put on top of a cup and pressed to seal it. Then, the production is finished and delivery robots bring the yogurts to the customers. Grippers at each customer's location pick up the yogurt and store it.

The course was supervised by one post-doctoral researcher, two Ph.D. students, one education assistant with a background in computer science education, and two teaching assistants supporting mitigation of technical challenges. The post-doc supervised the overall implementation of the project class. The Ph.D. students organized meetings and supported students. The education assistant gave educational guidance, deployed teaching techniques, conducted the questionnaires and the interviews.

## 3 STUDY DESIGN

To investigate the challenges and expectations of this project class, we collected data based on multi-method triangulation [13] through semi-structured interviews and a questionnaire, both consisting of open and closed questions. Further, observations from the weekly Scrum meetings, discussions, and presentations of deliverables were taken into account. Thus, we could collect data on the working and learning processes of individual participants and on cooperation between or within groups. Open questions support exploring the reason for a student answer in great detail, whereas closed questions enable better comparison between the results. Several verbal scales were deployed and transformed to numbers. For instance, a verbal scale for the evaluation of the perceived importance of certain competencies was provided to students as follows: (0) not specified, (1) not important at all, (2) less important, (3) rather important, (4) important, or (5) very important. Furthermore, we employed five-level Likert scales.

*Semi-structured Interviews.* After 10 weeks, when all teaching materials were handed out and elaborated by students, we conducted semi-structured interviews with all 20 students. These took 30 - 45 minutes, consisted of 18 questions and were carried out individually between each student and the education assistant. At the beginning of each interview, students were informed about the study's purposes, their opportunity to contribute to improvements in teaching, and the meaning of the employed scales. Moreover, all teaching materials and a list of competencies were provided to students. Ambiguities regarding the interview framework or the interview questions could be eliminated interactively. The interview questions and the questionnaire also are available on the companion website[5]. In the first part of the interview (i1-i6) students' characteristics (*e.g.,* age, semester, study program) were ascertained. Furthermore, they were asked about their working experience, participation in other project classes and prior knowledge regarding contents and courses, which were relevant for the project class and software development.

In the second part of the interview (i7-i11) the main focus was on getting insight into the expectations and perceived challenges students were faced with and reasons for these (**R1** and **R2**). Thus, students were inquired for their expectations (1) regarding technical, non-technical and methodological aspects of the project class in general (i7) and (2) regarding challenges or even anxieties deploying agile and model-driven software development (i8). Subsequently (i9), they were asked to compare their expectations with the actual situation, *i.e.,* to what extent (positively or negatively) it differs from their initial expectations. Since our project class dealt with Industry 4.0, question i10 (*cf.* Table 1) focused on the subjective importance of realistic and application-oriented examples for students (**R2**). Furthermore, a list of technical and non-technical competencies, built with the help of curricula and job descriptions, was provided to each student. Students were asked to assess their perceived importance of learning and improving each competence in the project class with reference to their future professional career (i11). To facilitate comparisons among students (**R2**), they were supposed to employ the verbal scale for rating importance as mentioned above. However, they were welcome to add further competencies to the list.

In the third part (*cf.* Table 1), we focused on evaluating methods and supplementary materials that were used in the project class (i12-i16) (**R3**). Factors impacting and supporting the learning and working process of students should be uncovered. Further, it should be revealed what students learned and which competencies they developed in the project class. Finally, students were inquired after their suggestions for enhancement (i17-i18), such that methodology and materials can be adapted in a manner that they promote learning and working processes of students in future courses (**R3**).

*Questionnaire.* To track development and challenges of diverse project working phases, we conducted a questionnaire at the end of the project class (after 15 weeks) with a subsequent discussion in class. The questionnaire (see Table 2) was handed out in class, took 20 minutes, was conducted anonymously, and used the same verbal scales with the same increments as the interviews. The questionnaire consists of 16 open and closed questions, where some

[5]Companion Website: www.se-rwth.de/materials/mbse4ind

| | |
|---|---|
| i10 | Do you think it is important to get an insight into real-world software engineering challenges in the project class "MBSE for Industry 4.0"? Classify your opinion using this scale and give a short reason: (0) not specified, (1) not important, (2) less important, (3) rather important, (4) important, (5) very important. |
| i12 | How important are transparent goals for you? Please, use the scale (see i10) of importance for your assessment. |
| i13 | How useful was it to mention the goals on the worksheets? (0) not specified, (1) not useful, (2) less useful, (3) rather useful, (4) useful, (5) very useful. |
| i14 | How did you like the teaching materials? How useful was each work sheet? Please, rate each worksheet according to the scale for rating usefulness (see i13) and give reasons for your judgment. |
| i15 | How did you like the interactive quizzes? |
| i16 | Which technical and non-technical competencies could you develop in the project class? Please, underline each competence of the list (*cf.* q3.1) according to the following rating system: (two lines) Yes, learned a lot. (one line) Yes, learned mediocre. (no line) Not learned at all. |
| i17 | How could your technical and non-technical competencies be promoted beyond the methodology, materials, and tools deployed? |
| i18 | To what extent do you feel prepared for your professional career through the project class? Which suggestions do you have for a better preparation? |

**Table 1: Excerpt of the interview questions on teaching methodology and materials. The full questionnaire is available from the companion website.**

questions from the interview were repeated. It investigates the students' interests (q1-q7, q15) and factors generally impacting their working and learning processes (**R1**-**R2**). Particularly, we inquired what the students learned (q3.2, q7, q11) (**R3**), which challenges (q10) (**R1**) they encountered and which suggestions for the improvement of the project class and methodology for learning, teaching, and organization (q8-q9, q12-q14) they have.

To investigate the overall results, we transcribed the interviews manually and devised categories summarizing the different topics. We applied the same categories to the answers of open questions from the questionnaire. To examine differences between B.Sc. and M.Sc. students, we split the data accordingly. For quantifiable data from closed questions with ordinal scales, we determined the median, which we will present in the results. After analyzing data, we partially related these and explain observations of our study in the following section.

## 4 OBSERVATIONS

This chapter presents the results and observations of the interviews, the questionnaire, and the notes from weekly meetings. It summarizes the technical and non-technical challenges faced by students applying agile MBSE to a smart factory in a project class, competencies they expect to learn within the project class, and teaching methods that we identified as suitable.

### 4.1 Challenges Faced by Students

After inspecting the transcripts of the interviews, we clustered the challenges addressed by students into three categories: (1) organization and task, (2) subject contents, and (3) working processes of the group. While challenges regarding (2) and (3) are explicitly mentioned in the questionnaire, challenges regarding (1) were implicitly addressed by students when stating requirements for their study process (q12-q13).

Challenges in the first category *organization and task* regard the understanding of the project class' statement of task and organizational issues. Some students perceived the task description, the objectives of the course, and challenges for software in Industry 4.0 as insufficiently and ambiguously communicated. They also described instructions as inconsistent, since the responsiveness of individual supervisors was not clear. To this effect, different supervisors

| | |
|---|---|
| q1 | Which professional occupation or function do you aim at or can you imagine to carry out later in your job? Name one (or more) job descriptions or occupations. |
| q2 | Which topics and contexts are of particular interest to you with regard to your professional career? |
| q3.1 | How important is it to you to learn or extend the following competencies during your studies as preparation for your professional life? Write a number in the column next to each competence: (0) not specified, (1) not important, (2) less important, (3) rather important, (4) important, (5) very important.<br>Technical competencies: Model-Driven Software Development, Agile Software Development (*e.g.*, Scrum, XP), Classical Software Development (*e.g.*, V-model, waterfall model), Software Architecture Design, Requirements Engineering, Object-Oriented Programming, Development of Cyber-Physical and Embedded Systems, Testing Software, Designing (UI).<br>Non-Technical competencies: Planning Ability, Organizational Ability, Structured Working, Goal-oriented Working, Analytical Ability, Problem-solving Ability, Abstraction Ability, Communication Ability, Team Ability, Presentation, Conflict Management, Time Management, Creativity, Innovative Ability, Self-Reliance, Flexibility, Reflectiveness. |
| q3.2 | Which competencies could you learn or extend in the project class? Write next to each competence (see q3.1): (2) Yes, learned a lot. (1) Yes, learned mediocre. (0) Nothing learned at all. |
| q4 | Why are you frustrated or disinterested in some courses? And how do you explain that some courses at university fail? |
| q5 | Which factors are conducive to your learning process and working behavior? Write a number next to each factor: (0) not specified, (1) not conducive, (2) less conducive, (3) rather conducive, (4) conducive, (5) very conducive.<br>Motivation of the supervisors, teaching competence of the supervisors, supervision, transparent learning objectives, methods used, topicality of the topic, relevance to reality and application, additional working material, use of different media, provision of different media, use of modern tools, group size of the course, space, fun. |
| q6 | Do you feel prepared for your professional life by your computer science studies? From a (a) technical and (b) non-technical point of view?<br>(a) Not at all ○ ○ ○ ○ ○ Very good, because...<br>(b) Not at all ○ ○ ○ ○ ○ Very good, because... |
| q7 | To what extent do you feel prepared for your professional life after the project class? From a (a) technical and (b) non-technical point of view?<br>(a) Not at all ○ ○ ○ ○ ○ Very good, because...<br>(b) Not at all ○ ○ ○ ○ ○ Very good, because... |
| q8 | How useful were the following tools or methods for your working process or study progress? Write a number next to each aspect: (0) not specified, (1) not useful, (2) less useful, (3) rather useful, (4) useful, (5) very useful: Slack, Problem-Oriented Learning, Design Thinking, Visualization Techniques, Individual Discussions as a Group with the Potential Client, Group-internal Exchange during sessions, Use of Scrum, Interactive Quizzes, Video presentations. |
| q9 | What was particularly interesting and helpful from the "motivation and excursion phase"? What was less useful? |
| q10 | Which technical and non-technical challenges and problems did you have with agile and model-driven software development? |
| q11 | What did you learn technically and non-technically in the project class? What do you take away for your professional life? |
| q12 | What have you missed and would have been beneficial for your working process and your learning? |
| q13 | How could your strengths and weaknesses regarding your technical and non-technical competencies be promoted with regard to your professional life in general and with regard to agile or model-driven software development? |
| q14 | Would you recommend the project class to your fellow students? What was special? Can you reason in two to three sentences, why your fellow student should choose it or why not? |
| q15 | Technical and non-technical competencies should be equally considered and promoted at the university as preparation for professional life. To what extent do you agree with this statement?<br>I do not agree at all ○ ○ ○ ○ ○ I agree completely |

**Table 2: Questions on students' experience and suggestions.**

responded to questions differently, sometimes even contradictory. As a result, this limited the students' progress. Likewise, students lacked (controlling and monitoring) support by supervisors and a structured process, *e.g.,* they missed rules, milestones and strict deadlines for project deliverables. Some students perceived the workload as too much and emphasized that the familiarization with different topics, *e.g.,* Industry 4.0, software development for embedded hardware, and patterns for decentralized decision making, was time-consuming.

The second category, *subject contents*, comprises challenges that arose, *e.g.,* from missing prior knowledge or from applying new content. (New) contents that students found challenging were particularly: software testing, agent communication and implementation of the communication protocol, the design of architectures using MontiArc, Scrum, Python, digital twins, and AutomationML [4]. Some students highlighted in that context missing materials as, *e.g.,* further literature or software documentation. Another problem students frequently commented on is the difficulty in dealing with MontiArc, in which they encountered several issues. The number of M.Sc. students perceiving challenges with MontiArc has increased at the end of the project class. In the interviews and questionnaire, B.Sc. students stated challenges in using software development tools such as Eclipse and Git and expressed their need for an introduction into these. Challenges in applying new content also include the integration of Scrum into the working process. Some students had difficulties in understanding the Scrum roles or in applying Scrum principles: "*In the project class, we could not apply 'real' Scrum. As we do everything within the entire group, there is no strict distribution of tasks*". Some students also reasoned difficulties with a lack of prior knowledge and the need for a Scrum workshop, while others justified neglecting Scrum due to lack of time. Instead, they preferred focusing on the realization of software components. Especially towards the end of the project, students faced challenges in integrating all software components. The implementation of the communication protocol and delays in previous implementation tasks challenged students in integration. Likewise, late testing of the software and difficulties in the communication and cooperation of the groups impacted the process. Besides this, there were difficulties in distributing the work among members of the groups. This lead to the situation that some students implemented essential components in individual work, which particularly challenged students with limited prior knowledge. This reduced students' communication, the overview of group progress, and the project progress in general. While the heterogeneous prior knowledge within groups was a perceived challenge by B.Sc. and M.Sc. students in the interviews, this aspect was not mentioned in the questionnaire. B.Sc. students emphasized that they benefited from the mixed groups.

Although technical and non-technical challenges were inquired equally in an open way, students most frequently addressed issues related to non-technical challenges in the interview. At the end of the project class, the perception of technical challenges increased, but their frequency is balanced with non-technical challenges. The latter were particularly concerned with the *working processes of the group*. Overall, only two students explicitly stated their satisfaction with the group dynamics outside the weekly project class meetings. In both, questionnaire and interview, students most commonly commented on challenges in communication and conflict management, which concerns particularly the collaboration within a group, but also the collaboration among different groups. In the interview, the second most mentioned issue were ambiguities in work and role distribution. This includes both, the view of the Scrum master who distributed the tasks and coordinated the group, and the view of group members who received the work instructions: "*The Scrum master in our group did not assign much to the individual person, which was problematic. So, sometimes I didn't know exactly what to do*". In addition, students had problems working in a structured and goal-oriented manner. Other challenges students mentioned at the end of the project class are the coordination of their group's time management and the ability to work in a team. The latter includes group dynamics and the organization of group meetings. Students specified group members' working attitude, individual progress, motivation problems, and lack of independence as hindering factors of the group's and overall project's progress.

## 4.2 Competencies Students Expected to Learn

One objective of our study is to identify the competencies students expect to learn within the project class (i7-i9) and our observations on the fulfillment of the expectations. For these, the interviews revealed three categories: (1) subject contents, (2) working processes in a group, and (3) competence development. In general, the students wanted to extend their knowledge of software development processes and gain practical experience through application and implementation of principles that they learned in theory. For instance, students emphasized that they were interested in applying principles of model-based software engineering and generative software development. Additionally, they were interested in Industry 4.0, as it is a current topic expected to have a large influence on software engineering challenges in the future. Moreover, the students were excited about robotics and the Fischertechnik factory plant demonstrator. Several students later described their experience as tangible and visible computer science: "*It is nice to see something moving. That you see what you have created*". Most students expected to gain practice and experience in developing software within a team to prepare for their job (i10). This also reflects in almost half of all participants desiring tasks and work processes (*e.g.,* Scrum) to be realized as in actual companies.

In general, 42% (8 of 19, one invalid return sheet) of the students felt well prepared for technical challenges in their job through their study program, 5.26% felt very well prepared. Through the project class, 37% felt well prepared for their job, but none of the students felt very well prepared. Being confronted with complex tasks and complex software in the project class, we reason that students noticed that they are not as well prepared as they expected. Furthermore, students criticized that the employed research software (MontiArc) did not have the expected stability. Students felt better prepared for non-technical challenges in their job through the project class (11% well prepared and 68% very well prepared) than through their study program in general (16% well prepared and 47% very well prepared). In both, interview and questionnaire, students criticized a general neglect of non-technical competencies (*e.g.,* working in groups and self-organization) in university courses and 80% (q15) demand their fostering.

In the interview (i9), students explicitly demanded the promotion of competencies. Almost half of all participants (40% B.Sc., 50% M.Sc. students) justified the relevance by means of the qualification for the job and the opportunity to practice in class. Some students even suggested that supervisors should differentiate between participants' work roles within a project (*e.g.,* Scrum master) to foster certain competencies. Table 3 presents results of the questionnaire and interview, where $Q_B$ is the questionnaire median of B.Sc. students and $I_B$ their interview median. $Q_M$ and $I_M$ are the respective medians of M.Sc. students.

**Technical Competencies** Classical software development appears to be unimportant for students. Particularly M.Sc. students want to expand competencies in agile software development and MBSE, whereby the perceived importance for students decreased in the questionnaire. In contrast, cyber-physical systems are more important to B.Sc. students. While learning object-oriented programming was very important for B.Sc. students in the interview, the perceived importance decreased in the questionnaire. For M.Sc. students, it remained equal. In contrast, the perceived importance of expanding competencies in software testing increased for M.Sc. students.

| | | Question | $Q_B$ | $I_B$ | $Q_M$ | $I_M$ |
|---|---|---|---|---|---|---|
| Expected To Learn | Technical | Classical Software Development | 3 | 2 | 3.5 | 2.5 |
| | | Agile Software Development | 3.5 | 4 | 4.5 | 5 |
| | | MBSE | 4 | 3 | 4.5 | 4 |
| | | Cyber-Physical Systems | 3.5 | 4 | 3 | 3 |
| | | Object-Oriented Programming | 5 | 4 | 4 | 4 |
| | | Software Testing | 4.5 | 4.5 | 4 | 5 |
| | Non-Technical | Time Management | 4 | 4 | 4.5 | 4.5 |
| | | Goal-Oriented Working | 4 | 5 | 4 | 4 |
| | | Working Independently | 4 | 4 | 4.5 | 5 |
| | | Teamwork | 4 | 4 | 5 | 5 |
| | | Planning and Organizing | 4 | 4 | 5 | 4 |
| | | Problem Solving | 4 | 4 | 5 | 4.5 |
| | | Innovativeness | 4 | 3 | 5 | 4 |
| | | Communication | 5 | 4 | 5 | 4.5 |
| | | Presenting | 4 | 3 | 5 | 4 |
| | | Creativity | 3 | 4 | 3 | 3.5 |
| Learned | Technical | MBSE | 1 | 2 | 1 | 1 |
| | | Agile Software Development | 1 | 2 | 1 | 1 |
| | | Requirements Engineering | 1 | 1 | 2 | 1 |
| | | Software Architecture Design | 0 | 2 | 1 | 1 |
| | Non-Technical | Planning | 1 | 1.5 | 1 | 1 |
| | | Teamwork | 1 | 2 | 2 | 1 |
| | | Presenting | 1 | 1.5 | 1 | 1 |
| | | Classical Software Development | 0 | 0 | 0 | 0 |
| | | User Interface Design | 0.5 | 1 | 0 | 0 |
| | | Software Testing | 0.5 | 1 | 0 | 1 |
| | | Abstraction Ability | 1 | 1 | 0.5 | 0 |
| | | Goal-Oriented Working | 1 | 1 | 0 | 1 |
| | | Reflectivity | 1 | 1.5 | 0 | 0 |

**Table 3: Excerpt of the observations of interview and survey.**

**Non-Technical Competencies** In the open interview questions, M.Sc. students most frequently commented on time management skills. Results of the questionnaires (i11, q3.1) confirm this. B.Sc. students primarily want to learn goal-oriented working. For M.Sc. students, working independently and improving teamwork skills is important. In addition, in the interview, they perceived it as more important to develop the ability to plan and organize than B.Sc. students. Furthermore, it is more important for M.Sc. students to improve competencies in problem-solving. Likewise, in the interview, the innovativeness is more important to them than to B.Sc. students, but their perceived importance decreased in the questionnaire. The same applies to the communication ability. Finally, learning or extending competencies in presenting is more important for M.Sc. students, but decreased for both study groups. Among the discussed competencies, creativity appears to be least important for students, but the perceived importance increased in the questionnaire. Other non-technical competencies (*e.g.,* flexibility, conflict management, abstraction) that are not pointed out above resulted in similar values between 3.5 and 4.5.

## 4.3 Perception of Teaching Materials

We deployed and investigated the interactive quizzes, design thinking, and problem-based learning, which were received differently by B.Sc. and M.Sc. students.

**Kahoot!** We employed Kahoot![6] interactive quizzes as part in every second weekly meeting to foster interaction and to consolidate knowledge. With Kahoot! students can answer multiple choice questions presented on a projector through their smartphones and the results are displayed immediately, giving direct feedback to the students. We created 86 questions from which the quizzes were generated. Out of these, 18 questions were repeated at the end of

[6]Kahoot! Website: www.kahoot.com

each Kahoot! for learning control. The questions were based on Bloom's revised taxonomy [2] on learning objectives: 64 questions require remembering, 21 understanding, and 1 applying knowledge. The questions focused on the initial presentations of the students and particularly on Scrum, MBSE, and Industry 4.0. The purpose of using Kahoot! was to loosen up the learning atmosphere as well as to check and secure the students' knowledge.

Based on the interview results (i15), Kahoot! quizzes are a welcome change during classes, which almost all participants mentioned. Since feedback about the correctness of answers follows immediately in the game, there is the potential to learn contents through the quiz [1]. About half of all participants (especially B.Sc. students) stated that they had built up or deepened their knowledge. However, students also mentioned obstructive or challenging factors: *e.g.,* time pressure due to the quick succession of questions and awarding bonus points for quick answers. Students also struggled with recalling their knowledge, since many questions were related to the initial presentations. While most students (65%) assessed the quiz positively and useful for their learning, the other students considered it merely knowledge test and noted that explanations after the quizzes were missing, which hindered learning.

**Design Thinking** Design thinking (DT) [18] is a staged method for the creative solution of various challenges comprising the stages of problem definition, research, idea formation, prototyping, and testing. We employed DT as it is solution-oriented and focuses on the iterative development of prototypes, which relates to agile software development. To investigate design thinking, we created three related worksheets, which were deployed in the weeks 5 to 7. (1) Apply DT to the initial user stories and conceive a first prototype of the overall system. (2) Consolidate understanding of DT and iterate on the first prototype. (3) Apply DT to conceive a method of efficiently elicitating user feedback based on your updated prototype. The aim of deploying these DT worksheets was to promote the students' skills regarding teamwork, communication, and reflectivity on work processes. For this purpose, each worksheet contained some guiding questions on the procedure and contents to be reflected.

Many students (90% B.Sc., 60% M.Sc.) found the reflection in their groups helpful and mentioned that they drew conclusions for their further collaboration. They emphasized that they had not previously been aware of the overall processes and work steps still to be completed since groups have been split into sub-groups. As a result, according to students, DT enabled a better understanding of requirements and an overview as well as promoted the ability to work in a team and solve problems. In the interviews, B.Sc. students rated DT as rather useful (3 of 5), while M.Sc. students found it generally useful (4). In the questionnaire, the benefit of DT was rated 3 by the B.Sc. students, and 3.5 by the M.Sc. students. This could be an effect of deploying DT early in the project class only as well as of changing focus from applying methods to deploying something executable towards the end. Moreover, the students of the UI team favored design thinking over the other teams. This might be due to the UI challenges being closer related to designing whereas the other teams' challenges were closely related to programming, CPPS, or algorithms. Future studies of DT will show its applicability to these fields.

**Problem-Based Learning** We also applied problem-based learning (PBL) [21], which assumes that learning requires actively dealing with a challenge, understanding it, and independently solving it. Hence, supervisors support students only when explicitly required. Similarly to DT, we provided introductory materials to the students, who then were encouraged to actively reflect on their group's challenges and to choose a technical or non-technical problem regarding their development process on their own. Through this, students should understand their challenges, devise a solution or strategy for solving it (ability to solve problems). To achieve this, we assigned the task of applying PBL to a self-selected technical or non-technical problem to each team. To this end, each student of the team prepared for a joint PBL session. Selected problems included the communication protocol between the different teams' CPPS, the internal collaboration within one of the groups, and the coordination between the groups.

Overall, the students (50% B.Sc., 60% M.Sc.) found PBL helpful for revealing, approaching and solving problems. Moreover, they stated that PBL fosters the ability to work in a team. In the interview, the benefit of PBL was rated useful (4) by the B.Sc. and M.Sc. students, while the questionnaire revealed (3) for both. According to five B.Sc. students (from two different teams out of four), their teams faced challenges in deploying the method. Either they could not find a common solution due to organizational reasons within their group (sub-groups) or they did not continue due to limited time. This might explain the different rating results.

## 4.4 Transferring Teaching Methods

Seven of ten B.Sc. students and two of ten M.Sc. students stated that they could use the teaching aids as a guide and believe they could reproduce these in future projects. Some students (40% M.Sc.) looked at methods through the lens of a teacher and, based on this, rated methods' benefits for others higher. For instance, some M.Sc. students considered DT as being helpful for B.Sc. students, to support guidance in new projects. Also, there were students (10% B.Sc., 50% M.Sc.) who were familiar with the methods or who employed their own strategies (such as consultation of colleagues or specific websites). Overall, four students considered the methods generally helpful but assigned lower ratings due to contextual conditions (*i.e.,* limited time for DT and PBL) or educational conditions (*e.g.,* different software engineering expertise between B.Sc. and M.Sc. students).

Based on their own perception (i16, q3.2), the B.Sc. students learned or expanded competencies in MBSE and agile software development, whereas M.Sc. students learned requirements engineering. As B.Sc. and M.Sc. students were similarly distributed among the different teams, we cannot explain this perception. In contrast, the perceived development of B.Sc. student competencies in software architecture design was reported as expected: they reported no competencies in the interview but claimed to have learned a lot in the questionnaire, while for M.Sc. students, this was more uniform.

Regarding non-technical competencies, students have learned or improved, for example, their abilities in planning, teamwork and presentation skills. Competencies, whose median has resulted in 0 for at least one student group are: classical software development, design of user interfaces, testing software, abstraction ability,

goal-oriented working, and reflectivity. The results for the abilities of innovation and creativity yielded 0 (not learned at all) for the median of both students groups in the interview and questionnaire. Students' demands or ambiguities regarding the terminology might impact their rating as well: Some students were wondering in the interview if it is possible at all to promote and learn these competencies. Besides the competencies presented, students rated the development of the other competencies from the list (q3.1) usually with (1) (yes, learned mediocre). Students' ratings might have changed (*cf.* Table 3) due to context conditions (*e.g.,* certain requirements and challenges of different project phases) or their awareness of personal learning or expanding needs.

## 5 DISCUSSION

Our exploratory study on the challenges and expectations when learning and teaching MBSE for Industry 4.0 is subject to various threats that prevent unconditional generalization. For instance, our project class employed specific modeling techniques, a tailored variant of Scrum, and specific kinds of CPPS. However, this study is the first on the topic including the challenges and expectations found by students and future replication will benefit from discussing its threats.

When teaching sufficiently complex topics, a large number of teaching materials may be suitable for achieving desired learning results. To investigate the effect of these materials, multiple observations – ideally in different contexts – should be made. However, due to the limited time-frame of 15 weeks, we opted for deploying different teaching materials on a weekly basis instead of handing out the same few teaching materials multiple times. This enables exploring a large variety of different techniques instead of focusing on a few. Our observations may guide selection and deployment of teaching in future studies. For our class, we selected B.Sc. and M.Sc. students. This is due to the teaching goal of exposing B.Sc. students to complex system engineering challenges as early as possible. The results on perceived challenges and the students' expectations, therefore, differ and focusing on students from the same level could produce more distinct results.

The interviewer was known to the students as a teaching assistant and, as such, may have been perceived as part of the teaching staff despite clarifying roles in the first sessions. However, this might have affected the interview results. Whether future studies should aim for a completely neutral interviewer depends on the complexity of the project class: a teaching assistant interviewer is familiar with the context, part of the challenges, and, hence, can better clarify the interview questions and answers than an external reviewer. Aside from these threats specific to our study, there are threats to causality (the internal validity) and generalizability (external validity) of our findings. Considering internal validity, our study lacks comparable results (*e.g.,* by running the project class multiple times to find recurring challenges and expectations) and, due to RWTH Aachen University's computer science curriculum, the students were unfamiliar with CPPS. External validity is threatened as (1) we inquired qualitative results instead of quantitative results; (2) the limited number of participants gives weight to personal opinions over 'objective' observations; and (3) the students have similar educational backgrounds.

## 6 LESSONS LEARNED

From our observations, we learned various lessons for future project classes on MBSE for Industry 4.0. In general, students were pleased with applying MBSE to Industry 4.0, because it is application-oriented and experienceable. We suggest competence-oriented teaching and learning methods and fostering non-technical competencies besides technical ones. For instance, students want to learn to work in teams and in a goal-oriented and structured manner. Especially B.Sc. students emphasized guiding questions and working steps within methods being beneficial for better grasping the requirements and for supporting initial implementation approaches. To foster this, we propose to (1) Provide students with detailed requirements and objectives of the project task, fix milestones, the form of deliverables, and organizational rules at the beginning. (2) Perform regular checks throughout the semester to monitor the progress and check if deadlines are met. (3) Integrate testing the software as a regular deliverable. (4) Discuss impediments in small teams to better support individual students. (5) Split supervisors' responsibilities, such that students have one contact person for each kind of question. Furthermore, students faced challenges in the collaboration within and between groups regarding, *e.g.,* communication, group dynamics, and conflict management. To improve on this, we suggest integrating face-to-face meetings with individual groups into the sessions. This supports: teamwork processes, solving current issues and technical challenges, and gaining an overview of group progresses. Moreover, short, weekly presentation by students helped to understand and adjust the development progress. Apart from meetings, providing groups with communication means (we employed Slack[7]) has proven helpful.

While the student presentations at the beginning of the project class were perceived as unnecessary and too time-consuming, they helped building a common body of knowledge students benefited from. To save time, we plan to reduce initial group presentations to crucial topics for the project class only. While reducing the broader view on the theme of the project class, the potentially added depth of redundant initial presentations on the same topic might foster their understanding. The mixed groups of B.Sc. and M.Sc. students helped (particularly B.Sc. students) to spread heterogeneous knowledge. In the future, we plan to introduce cross-cutting teams for, *e.g.,* realizing the communication protocol, to better spread knowledge and foster collaboration. The interactive quizzes were perceived as a good means to break the meetings' monotony. Further, students complained about strict time limits, which they found stressful.

From the implementation of the questionnaire, we learned that questions have to be precisely focused. With too broad (particularly open) questions opinions of students become very similar. Students wanted to be strictly guided and liberated from as many design decisions as possible, which we found surprising. Moreover, we intend to perform more questionnaires in a single project class to observe trends and to perform the same questionnaire in different classes, similar to former project classes on MBSE for robotics [16].

## REFERENCES

[1] H. Z. Abidin and F. H. K. Zaman. 2017. Students' perceptions on game-based classroom response system in a computer programming course. In *2017 IEEE 9th International Conference on Engineering Education.* 254–259.

[7] Slack Website: www.slack.com

[2] L. W. Anderson, D. R. Krathwohl, P. W. Airasian, K. A. Cruikshank, R. E. Mayer, P. R. Pintrich, J. Raths, and M. C Wittrock. 2001. A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Abridged Edition. *New York: Longman* (2001).

[3] A. Butting, A. Haber, L. Hermerschmidt, O. Kautz, B. Rumpe, and A. Wortmann. 2017. Systematic Language Extension Mechanisms for the MontiArc Architecture Description Language. In *Modelling Foundations and Applications (ECMFA'17), Held as Part of STAF 2017*. 53–70.

[4] R. Drath, A. Lüder, J. Peschke, and L. Hundt. 2008. AutomationML-the glue for seamless automation engineering. In *Proceedings of 13th IEEE International Conference on Emerging Technologies and Factory Automation*. 616–623.

[5] J. Dregger, J. Niehaus, P. Ittermann, H. Hirsch-Kreinsen, and M. ten Hompel. 2016. The digitization of manufacturing and its societal challenges: a framework for the future of industrial labor. In *2016 IEEE International Symposium on Ethics in Engineering, Science and Technology (ETHICS)*. 1–3.

[6] CRO Forum. 2015. The Smart Factory – Risk Management Perspectives. (2015). http://www.thecroforum.org/wp-content/uploads/2015/12/CROF-ERI-2015-The-Smart-Factory1.pdf Accessed: 2018-08-30.

[7] M. Galster and S. Angelov. 2016. What Makes Teaching Software Architecture Difficult?. In *Proceedings of the 38th International Conference on Software Engineering Companion (ICSE '16)*. ACM, 356–359.

[8] M. Hermann, T. Pentek, and B. Otto. 2016. Design principles for Industrie 4.0 Scenarios. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE, 3928–3937.

[9] D. S. Kolovos and J. Cabot. 2016. Towards a Corpus of Use-Cases for Model-Driven Engineering Courses. In *Joint Proceedings of the 12th Educators Symposium (EduSymp 2016) and 3rd International Workshop on Open Source Software for Model Driven Engineering (OSS4MDE 2016) co-located with the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016)*. 14–18.

[10] K. Lano, S. Y. Tehrani, and H. Alfraihi. 2015. Experiences of Teaching Model-Based Development. In *Proceedings of the MODELS Educators Symposium 2015 co-located with the ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems*. 43–54.

[11] J. Ludewig and I. Bogicevic. 2012. Teaching software engineering with projects. In *2012 First International Workshop on Software Engineering Education Based on Real-World Experiences (EduRex)*. 25–28.

[12] C. Matthies, T. Kowark, K. Richly, M. Uflacker, and H. Plattner. 2016. How Surveys, Tutors, and Software Help to Assess Scrum Adoption in a Classroom Software Engineering Project. In *Proceedings of the 38th International Conference on Software Engineering Companion (ICSE '16)*. ACM, 313–322.

[13] P. C. Meijer, N. Verloop, and D. Beijaard. 2002. Multi-Method Triangulation in a Qualitative Study on Teachers' Practical Knowledge: An Attempt to Increase Internal Validity. *Quality and quantity* 36, 2 (2002), 145–167.

[14] S. Mosser, P. Collet, and M. Blay-Fornarino. 2014. Exploiting the Internet of Things to Teach Domain-Specific Languages and Modeling: The ArduinoML project. In *Proceedings of the MODELS Educators Symposium co-located with the ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems (MODELS 2014)*. 45–54.

[15] G. Mussbacher, D. Amyot, R. Breu, J.-M. Bruel, B. H. C. Cheng, P. Collet, B. Combemale, R. B. France, R. Heldal, J. Hill, J. Kienzle, M. Schöttle, F. Steimann, D. Stikkolorum, and J. Whittle. 2014. *The Relevance of Model-Driven Engineering Thirty Years from Now*. Springer, 183–200.

[16] J. O. Ringert, B. Rumpe, C. Schulze, and A. Wortmann. 2017. Teaching Agile Model-Driven Engineering for Cyber-Physical Systems. In *International Conference on Software Engineering: Software Engineering and Education Track (ICSE'17)*. IEEE, 127–136.

[17] J. O. Ringert, B. Rumpe, and A. Wortmann. 2013. A Case Study on Model-Based Development of Robotic Systems using MontiArc with Embedded Automata. In *Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme*. 30–43.

[18] P. G. Rowe. 1991. *Design Thinking*. MIT press.

[19] J.-P. Steghöfer, E. Knauss, E. Alégroth, I. Hammouda, H. Burden, and M. Ericsson. 2016. Teaching Agile: Addressing the Conflict Between Project Delivery and Application of Agile Methods. In *Proceedings of the 38th International Conference on Software Engineering Companion*. ACM, 303–312.

[20] J. Vanhanen, T. O. A. Lehtinen, and C. Lassenius. 2012. Teaching real-world software engineering through a capstone project course with industrial customers. In *2012 First International Workshop on Software Engineering Education Based on Real-World Experiences (EduRex)*. 29–32.

[21] D. F. Wood. 2003. Problem based learning. *BMJ* 326, 7384 (2003), 328–330.

[22] A. Wortmann, B. Combemale, and O. Barais. 2017. A Systematic Mapping Study on Modeling for Industry 4.0. In *Conference on Model Driven Engineering Languages and Systems (MODELS'17)*. IEEE, 281–291.