A Hybrid Teaching Model for the Software Requirement Analysis and Modeling Course

Haibo Li(✉) and Miao Chen

College of Computer Science and Technology, Huaqiao University, Xiamen, China
lihaibo@hqu.edu.cn

Abstract. The Software Requirement Analysis and Modeling course, as a core course of software engineering, is extremely difficult to provide knowledge to students because of its ambiguity, uncertainty, variability and subjectivity. To solve the problem, a hybrid teaching model is proposed in which flipped classrooms, case-centered teaching, and scenario-based teaching are integrated used. In addition, designing software prototype as a part of requirements verification is added into course. By comparing to traditional prototyping tools, the preliminary software requirement and modeling can be validated better. Through practice, the hybrid teaching model can further improve students' learning and has guiding significance for the reform of the courses in software engineering.

Keywords: software engineering · software requirements analysis · software modeling · software prototype

1 Introduction

At present, all colleges and universities in China are promoting the construction of "first-class undergraduate majors" with engineering education professional certification. Under the OBE teaching model, the selection of teaching contents, the allocation of teaching resources and the organization of teaching steps are all proceeding revolving around the teaching goals achievement [1]. The Software Requirements Analysis and Modeling course, as an important core course of software engineering, has the characteristics of boring basic theory and strong practicality. Therefore, the traditional teaching method based on lectures already has many shortcomings [2]. Simultaneously, as software requirement is vague, uncertain, variable and subjective [3], there is no mature method to follow to capture user requirement. For students without software design experience and social experience, it is even more difficult to learn such knowledge [4]. Therefore, Software Requirements Analysis and Modeling course reconstruction according to course objectives, content and student characteristics has become the top priority of course reform.

Based on the 15-year teaching history of this course, the problems existing in teaching procedure is analyzed in this paper. The course reconstruction and teaching mode design around the teaching goals are proposed, which include specifically how to carry out case-centered teaching method and scenario-based teaching method, and how to organize

V. Balakrishnan et al. (Eds.): ICMEIM 2022, AHSSEH 6, pp. 548–552, 2022.
https://doi.org/10.2991/978-94-6463-044-2_69

a flipped classroom. The valuable methods are also provided for the construction of software engineering courses.

2 Analysis of Traditional Teaching Model

User requirements are the source of software development. As many requirements are difficult to describe clearly and objectively, the main means of requirement acquisition is the combination of methods such as interviews, rapid prototyping, and requirements tracking matrices. This makes the teaching process of the software requirement analysis and modeling courses generally suffer from boring knowledge, uncertain theoretical approaches, and low student interest. Four particularly important aspects were addressed.

1. It is difficult to arrange course practice for students to experience the real software requirements analysis process due to the lack or absence of course practice.
2. If teachers lack software project development experience, they seldom use real-life examples to illustrate the requirements analysis process in the teaching process.
3. Requirements verification is very weak. The current method mainly uses the prototype method to verify the requirements. However, the prototype tool is good at showing the human-computer interaction, not the verification of the software model.
4. The role of human factors in needs analysis is often disregard. Students do not realize the role of interpersonal communication in the requirements acquisition process.

3 Design of Teaching Model

3.1 Teaching Model

The training goals of the Software Requirements Analysis and Modeling course include software modeling for complex software engineering problems, defining and formulating problems to be solved in complex software engineering projects, etc. [5]. To achieve these goals, a hybrid teaching model is proposed to solve the problems in the traditional teaching model, as shown in Fig. 1.

The software requirements analysis process consists of four phases: requirements acquisition, requirements analysis, requirements specification, and requirements verification [6]. Among them, the sources of requirements acquisition include users, hardcopy data (forms or report), relevant documents and domain experts. Interviews are the main way to obtain requirements from people and are full of uncertainty and subjectivity. Therefore, it is ideal for students to experience this process by designing specific scenarios and role-playing, to learn to flexibly grasp the different ways of obtaining information from different sources.

We select typical categories of projects, such as inventory management, procurement management, intelligent transportation, smart medical, etc. We prepare some relevant reports, rules and regulations and other documents in advance to performance and interpret requirements acquisition process. Finally, although software requirements has no permanent form, and its acquisition has no permanent approach, interpreting the process according to the theory allows students to fully understand the ways and means of acquiring software requirements.

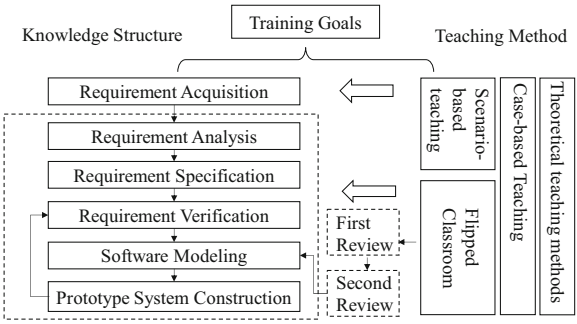

Fig. 1. Hybrid teaching model

Students are divided into teams, each of which is assigned a project to learn requirements analysis, requirements specification, requirements verification, software modeling, and building a prototype system. The course of studies covers the theory first, and practice which depends on self-study through document review, MOOC, etc.

Building a prototype system complements requirements verification and is a closed-loop design. Currently, most of the requirement verification use prototyping tools, such as Axure RP, to facilitate requirement confirmation with users through rapid design of human-computer interaction systems. We believe that there are shortcomings in this verification method. Most of these tools specialize in human-computer interaction design. However, more and more software projects now show data-intensive and algorithm-intensive characteristics. The core parts in these projects lack effective means of verification.

Therefore, in view of the shortcomings of the existing prototype tools, we extend the software requirements analysis and modeling process to the software development stage. It is a helpful complement to the traditional requirements verification by designing our own preliminary software prototypes.

3.2 Course Reconstruction

According to the model proposed in the previous section, the teaching content should be reconstructed. A seminar-style lecture mode is created with the help of software project cases and a problem-oriented approach. The OBE-based teaching goals drive the course content into two categories: requirements modeling and software modeling. A case is a task for a team, for example, an inventory management system. In case-centered mode, the team starts from the data flow diagram of requirements analysis to software modeling, and finally the design of a prototype system to verify the requirements and modify them, forming a complete closed loop. If there is insufficient course time, the teaching of requirement acquisition could be limited to a typical case. In the scenarios, the teacher team acts as the software project party A as far as possible, and sets up "ideal" and "undesirable" practices to cooperate with party B, which is acted by students. Specifically, these practices can be the completeness of the forms provided, the attitude of the staff, the clarity of the description of the requirements, and a variety of other

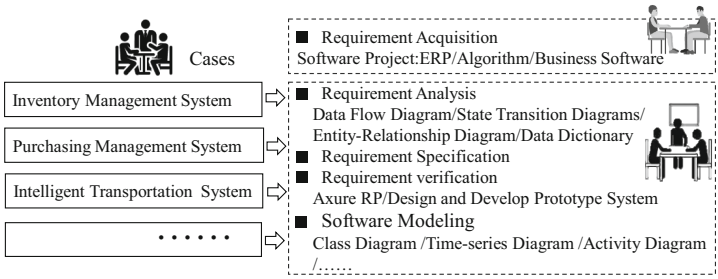

Fig. 2. Software Requirement Analysis and Modeling course reconfiguration

ways to examine the student's ability to grasp the software requirements. The course reconstruction is shown in Fig. 2.

4 Effect Evaluation

The software engineering of Huaqiao University has been approved as a construction of national first-class undergraduate majors. Using the teaching model proposed in this paper, the teaching team of Software Requirements Analysis and Modeling course has completed 5 semesters of teaching practice at both undergraduate and master's levels. Specifically, the effectiveness is obvious very much, and includes the following aspects.

1. The students gained a deeper understanding of the requirements acquisition process for complex software projects. Simultaneously, the students learned about responsibilities of software requirement engineers and how to deal with the interpersonal communication.
2. The students have a clearer understanding of the purpose and role of requirements modeling and software modeling. The students could identify and determine the core problems or bottlenecks, and the scope of requirements in complex software engineering projects. They were able to improve the process through self-study, verification and re-improvement.
3. The development of software prototype systems is added to the process of requirements verification. The students skilled in development tools were able to develop a software prototype system using SSM framework or Python to verify the requirements.

However, there are still some problems in the teaching practice process. For example, both undergraduate and master students have deficiencies in using software development tools. Only half of them could master the tools through self-learning, which has a greater impact on the requirement verification of algorithm-intensive software project cases.

5 Conclusion

Based on 15 years of experience in teaching Software Requirement Analysis and Modeling courses, under the background of promoting the construction of "first-class undergraduate majors" with engineering education professional certification, an OBE-based hybrid teaching model is proposed in this paper. In this model, flipped classrooms, case-centered teaching, and scenario-based teaching are integrated used. Especially in software requirements verification process, students appropriately develop software prototypes to validate software requirements according to the new characteristics of the current development of the software industry. A complete cognitive and logical closed-loop of the software requirements analysis process can form.

References

1. Yin, Zhiqiang, Liu, Zenghui, Zhang, Zhixiong, Chang, Jucai, and Hu, Xuelong. Experiences from a new project-driven and outcome-based educational concept in a blasting engineering study program [J]. International Journal of Emerging Technologies in Learning, 2021, 16(8):145-161.
2. Li, Xiaoyan; Bi, Bing; Xu, Zhiru; Wang, and Jingying. Construction of multi-dimensional teaching reform system of Biochemistry based on outcome-based education [J]. Chinese Journal of Biotechnology, 2020, 36(10):2226–2233.
3. Gren, Lucas. A Flipped Classroom Approach to Teaching Empirical Software Engineering [J]. IEEE Transactions on Education, 2020, 63(3):155-163.
4. Zheng, ShanHong, Zhao, Hui, Peng, Xinyi, Wang, Guochun, and Dong, Yaze. Exploration and practice of four-in-one teaching model for the software engineering course [J]. Software Engineer, 2019,22(9):41–43,40.
5. L.X. Xu and H.Y. Wu, "Collective intelligence based software engineering [J]. Computer Research and Development 2020,57(3):487-512.
6. K.E. Wiegers, and J. Beatty, Software Requirements [M]. 3rd ed., Microsoft Press, 2016.