Innovative Research of Project Case in Software Engineering Teaching Mode

Wenping Chen, Jie Huang, and Fanqing Meng(✉)

School of Information Science and Engineering, Zaozhuang University, Zaozhuang 277160, Shandong, China
mengfanqing@uzz.edu.cn

Abstract. Aiming at the problems existing in the teaching process of Software Engineering, as well as the training orientation of application-oriented undergraduate talents and the needs of enterprise talents, this paper adopts the deep integration of modern information technology, practical projects and classroom teaching to promote the software engineering curriculum reform, and carries out diversified teaching innovation from teaching models, teaching projects, teaching evaluation and other aspects.

Keywords: Software Engineering · enterprise demand · curriculum reform

1 Introduction

In the context of new engineering, how to effectively cultivate innovative software engineering talents is one of the key issues that local application-oriented universities need to address. Software Engineering, as a core professional course for computer majors, is an important way to cultivate students' comprehensive ability in software engineering [1]. Software Engineering is a required course for students to participate in discipline competitions and optimize their work, it is also a regular subject for graduate entrance exams or written recruitment exams for major enterprises. This course plays a connecting role in cultivating students' knowledge, abilities, and comprehensive qualities [2–4].

1.1 Analysis of Learning Situation

The variance of distribution of students' professional foundation in this course is uneven. Some students have systematically developed complete software and won certain competition awards, while others understand relevant concepts of software development, but they are unable to systematically connect the knowledge points involved in software development process, and more importantly, their professional foundation is weak. Moreover, some students have weak professional foundation.

There is a certain gap between current teaching content and mode of software engineering courses and the actual needs of enterprises, making it difficult to adapt to the needs of talent development and cultivation [5, 6].

D. Kumar et al. (Eds.): IEIT 2023, AHSSEH 10, pp. 758–764, 2023.
https://doi.org/10.2991/978-94-6463-230-9_90

The traditional grading model for courses is relatively simple, emphasizing results over processes, and evaluating students' mastery of software engineering courses by assigning different weights to assessment content [7, 8]. It mainly assesses students' understanding of the basic knowledge, but this method is not sufficient to improve students' hands-on ability, and is not conducive to stimulating students' interest in actual software design and development.

Therefore, we must change our educational concepts and explore new teaching models for Software Engineering course.

2 Innovate Teaching Methods, Optimize Teaching Design, and Enrich Teaching Resources

In order to address new challenges faced by software engineering talent cultivation described above, we have conducted a series of exploratory practices in the teaching of Software Engineering courses. Course group adopts diversified teaching methods based on students' basic learning situation, with the goal of overcoming student differences and improving students' interest. And through teaching technology and means, it can promote the teaching quality of software engineering courses from various aspects.

2.1 Optimize Teaching Design with Projects and Cases as Task Driven

Combine project teaching with case teaching. Software Engineering teaching model based on project teams and cases includes two aspects: group project practice throughout the entire semester in practical courses, and "case analysis centered" teaching in theory course. Driven by real enterprise tasks, case practice, and case solving, enhance theoretical knowledge understanding, and cultivate students' necessary abilities for professional positions such as project analysis, design, code testing, and teamwork.

Teaching content runs through projects and focuses on the entire process of typical project development. Connect isolated knowledge points from books through projects to gradually demonstrate the complete process of project cases. According to teaching progress, course group will demonstrate key points and precautions to be paid attention to in the classroom during project initiation, analysis, design, development, and testing of the expansion project. At the same time, students completed group training projects in parallel. To promote students' ability to develop engineering skills in project management and project implementation while improving their technical skills.

The selection of cases in instructional design is divided into three categories: basic cases, which help students understand basic ideas and structures of knowledge points. Synchronize training cases, which consolidate knowledge through specific applications through analysis, design, testing, and other operations on small projects. Expanding project, through the demonstration of enterprise training projects, students can use knowledge they have learned to analyze which part of the system uses this knowledge, and experience the importance of specific projects in software engineering. The teaching process design is shown in Fig. 1.

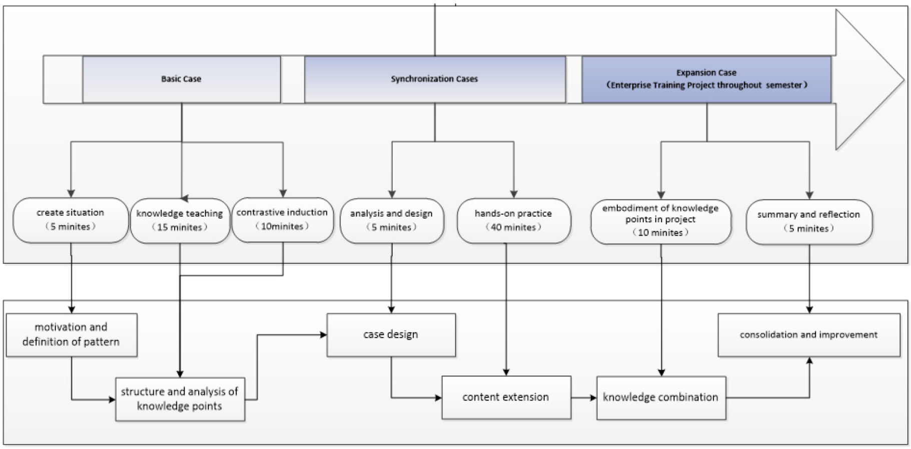


Fig. 1. Teaching process design.

2.2 Selection of Projects and Cases

Course group assigns students projects and cases with appropriate difficulty levels through research on teaching content combined with an understanding of students' learning level. Students are the main body of learning. Through full communication between teachers and students around projects and cases, lead students to improve their ability to identify, analyze, and solve problems during development process.

By collecting interesting and real cases from various stages of software engineering, as shown in Fig. 2, cases come from textbooks, enterprise experiences, forums, classic anecdotes etc., so that students can truly feel the charm and breadth of software engineering. Adopt a software engineering teaching reform strategy oriented to enterprise needs to stimulate students' interest in learning. Through the content derived from software engineering, the market share of positions, salary rankings, growth space, and other aspects, supplemented by detailed data and vivid cases, students can realize the role of learning software engineering well for personal growth and development.

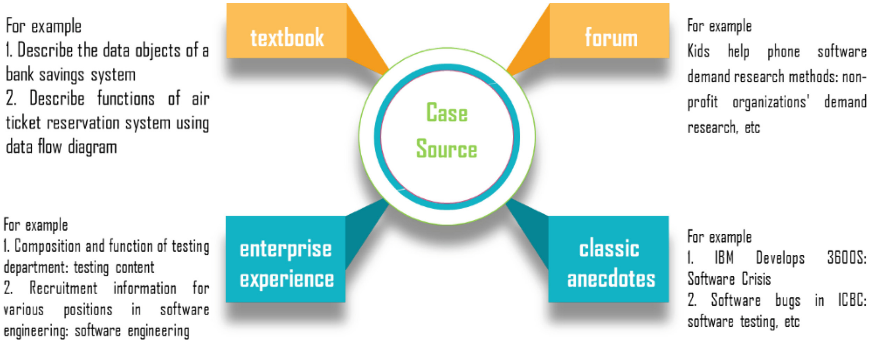

For example
1. Describe the data objects of a bank savings system
2. Describe functions of air ticket reservation system using data flow diagram
textbook
forum
For example
Kids help phone software demand research methods: non-profit organizations' demand research, etc
Case Source
For example
1. Composition and function of testing department: testing content
2. Recruitment information for various positions in software engineering: software engineering
enterprise experience
classic anecdotes
For example
1. IBM Develops 360OS: Software Crisis
2. Software bugs in ICBC: software testing, etc

Fig. 2. Case source.

2.3 Group Cooperation Around the Entire Process of Project Development

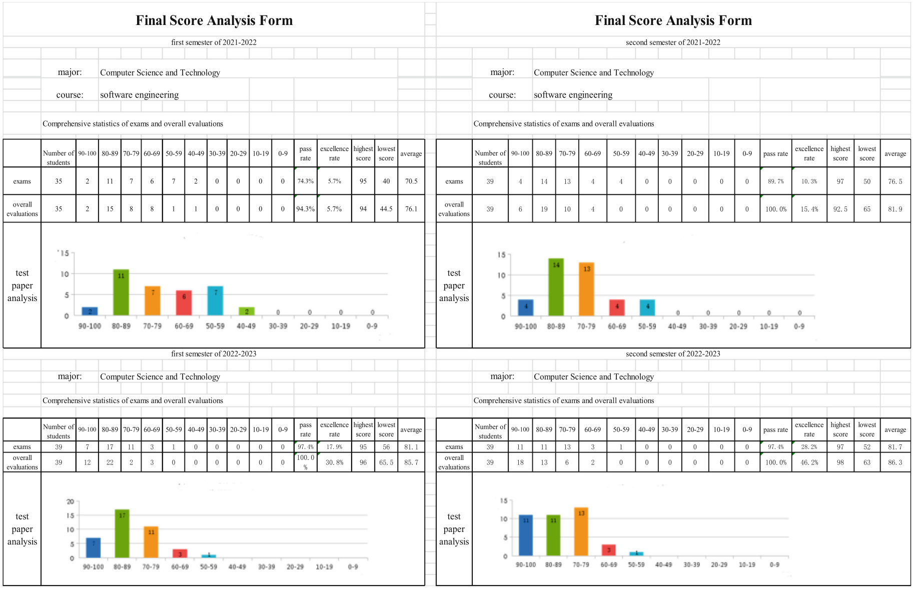
For the entire process of software development training and relatively complex teaching content, group discussion and cooperation are adopted. At the beginning of the semester, the teacher provides some topics (including project descriptions, open source project introductions, and previous outstanding works) for students to choose. Students can freely form a project group based on their personal interests, past competitive works, or other optional topics. Each group has 4–6 students sharing a topic.

According to teaching progress of software engineering and characteristics of project development stages, teachers will regularly check the progress of each team project, and provide targeted feedback through group reports combined with the teaching content. Periodic project reports submitted by each group are scored and evaluated through combination of group mutual evaluation and teacher inspection, and the quality of project acceptance is used as the group score.

2.4 Optimize Assessment Methods

Adopt multiple evaluation methods in assessment to form a personalized evaluation with broad content and strong objectivity. Process evaluation and summative evaluation are equally emphasized, with final score accounting for 50% and usual score accounting for 50%. The evaluation of the learning process will be more objective.

Combination of general evaluation and personalized evaluation. For each student, general evaluation is composed of group practice scores and final scores. Final exam is organized flexibly and has a variety of examination contents. Using multiple forms



Final Score Analysis Form

first semester of 2021-2022

major: Computer Science and Technology

course: software engineering

Comprehensive statistics of exams and overall evaluations

	Number of students	90-100	80-89	70-79	60-69	50-59	40-49	30-39	20-29	10-19	0-9	pass rate	excellence rate	highest score	lowest score	average
exams	35	2	11	7	6	7	2	0	0	0	0	74.3%	5.7%	95	40	70.5
overall evaluations	35	2	15	8	8	1	1	0	0	0	0	94.3%	5.7%	94	44.5	76.1

test paper analysis

Final Score Analysis Form

second semester of 2021-2022

major: Computer Science and Technology

course: software engineering

Comprehensive statistics of exams and overall evaluations

	Number of students	90-100	80-89	70-79	60-69	50-59	40-49	30-39	20-29	10-19	0-9	pass rate	excellence rate	highest score	lowest score	average
exams	39	4	14	13	4	4	0	0	0	0	0	89. 7%	10. 3%	97	50	76. 5
overall evaluations	39	6	19	10	4	0	0	0	0	0	0	100. 0%	15. 4%	92. 5	65	81. 9

test paper analysis

first semester of 2022-2023

major: Computer Science and Technology

Comprehensive statistics of exams and overall evaluations

	Number of students	90-100	80-89	70-79	60-69	50-59	40-49	30-39	20-29	10-19	0-9	pass rate	excellence rate	highest score	lowest score	average
exams	39	7	17	11	3	1	0	0	0	0	0	97. 4%	17. 9%	95	56	81. 1
overall evaluations	39	12	22	2	3	0	0	0	0	0	0	100. 0 %	30. 8%	96	65. 5	85. 7

test paper analysis

second semester of 2022-2023

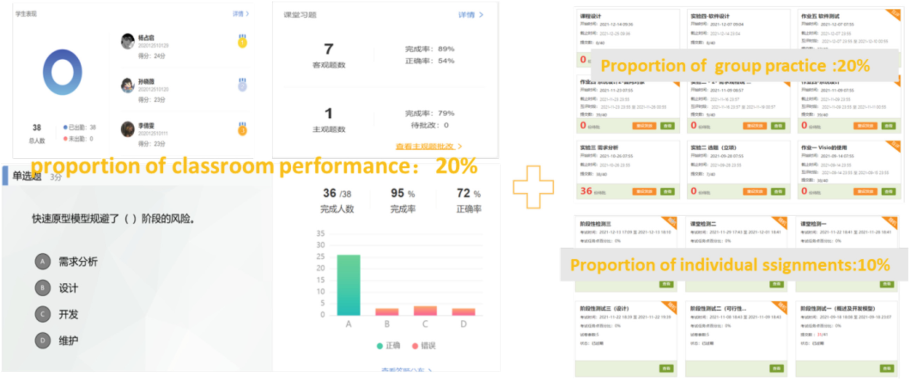
major: Computer Science and Technology

Comprehensive statistics of exams and overall evaluations

	Number of students	90-100	80-89	70-79	60-69	50-59	40-49	30-39	20-29	10-19	0-9	pass rate	excellence rate	highest score	lowest score	average
exams	39	11	11	13	3	1	0	0	0	0	0	97. 4%	28. 2%	97	52	81. 7
overall evaluations	39	18	13	6	2	0	0	0	0	0	0	100. 0%	46. 2%	98	63	86. 3

test paper analysis

Fig. 3. Final score analysis form



快速原型模型规避了（）阶段的风险。

A 需求分析
B 设计
C 开发
D 维护

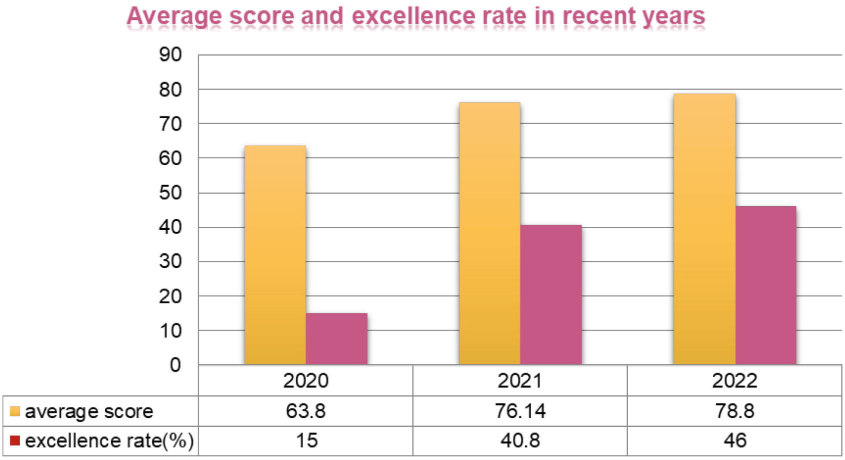
Fig. 4. Process evaluation

such as written tests, oral defense, and operations, combined with data from Superstar and Rain classroom platforms, conduct result assessment on comprehensive application ability of students to analyze and solve problems. Final score analysis form is shown in Fig. 3, Process evaluation is shown in Fig. 4.

3 Teaching Implementation Effect

This course has achieved certain results in teaching reform and innovation. In terms of teaching effectiveness and feedback from teachers and students, students' learning interest and participation have significantly increased, and teacher-student interaction has become more active. Teaching evaluation results show that students are highly satisfied with their mastery of course. After comparing final scores of the two semesters, average score and excellent rate of students have improved, and failure rate has decreased, as shown in Fig. 3.

In innovation competitions, students' learning enthusiasm and initiative have greatly increased, and team spirit, innovation ability, and communication and expression skills have significantly improved. Teacher team organizes students to participate in various software competitions at provincial level and even nationwide, and the participation rate and award rate of various projects have improved.



	2020	2021	2022
■ average score	63.8	76.14	78.8
■ excellence rate(%)	15	40.8	46

Fig. 5. Comparison of average score and excellence rate in recent three years.

References

1. Jiang Y. Development and Design of Online Examination System for Teaching Reform of Marketing Course Based on UML[C]// International Conference on Smart Grid and Electri-cal Automation. IEEE, 2021.
2. Fan, G., et al. "Teaching Reform Conception of Software Engineering Core Courses for Cloud Application." Computer Education 12(2022):7.
3. Zhoulin W U, Wang W, Lili J I, et al. Teaching Reform and Practice of Animal Products Processing under the Background of Intelligent Agriculture[J]. Asian Agricultural Research, 2022, 14(11):3
4. Yao L, Zhang Q, Chi Q, et al. Teaching Reform of Java Web Programming Course Based on the Concept of OBE[C]// 2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT). 2020.
5. Gren L. A Flipped Classroom Approach to Teaching Empirical Software Engineering[J]. IEEE Transactions on Education, 2020, PP(99):1–9.
6. Cico O, Jaccheri L, Nguyen A, et al. Exploring the intersection between software industry and Software Engineering education -A systematic mapping of Software Engineering Trends. 2020.
7. FJ García-Pealvo, A García-Holgado, A Vázquez-Ingelmo, et al. Planning, Communication and Active Methodologies: Online Assessment of the Software Engineering Subject during the COVID-19 Crisis[J]. RIED Revista Iberoamericana de Educación a Distancia, 2021, 24(2).
8. He Y, Zhang Y, Liu L, et al. Exploration on Blended Teaching Mode of "Introduction to Software Engineering" Based on SPOC[J]. Computer Education, 2022(12):5.