Research on the Instructional Design for Software Testing under the Background of Engineering Education Accreditation

Yan Yang*, **Rong Li**
Computer School, Central China Normal University, Wuhan, China

Abstract: Based on analyzing the characteristics of engineering education accreditation, this paper studies the *Software Testing Technology* course under the background of engineering education accreditation from the aspects of course objectives, theoretical content, practical content, assessment and evaluation. Some instructional design ideas and methods are proposed, which implement the concept of student-centered, results oriented, and continuous improvement.

Keywords: Engineering Education Accreditation; Software Testing; Instructional Design

1. Introduction

Professional accreditation in engineering education represents a globally recognized system for ensuring the quality of engineering education. This also serves as a crucial groundwork for the global acknowledgment of engineering education and the credentials of engineers [1]. At the heart of professional accreditation in engineering education lies the verification of engineering graduates adhering to industry-acknowledged quality benchmarks, essentially a certified assessment tailored to the training goals and graduation export prerequisites [2]. Accrediting engineering education professionally necessitates creating a specialized curriculum, assigning teaching personnel, and setting up educational environments, all centered on the fundamental goal of attaining students' graduation skills. The focus is on creating an ongoing enhancement process and a culture within the field to guarantee the excellence and dynamism of professional training [3].

Engineering education professional accreditation encompasses three key principles: a philosophy focused on students, an educational approach centered on results, and a culture of ongoing quality enhancement [4]. Known as student-centered education, it necessitates an educational objective centered on student development, a teaching structure dedicated to enhancing student capabilities, faculty and educational resources addressing student learning outcomes, and an assessment centered on gauging student effects. The outcome-focused educational approach fully elucidates the interplay between school orientation, training goals, graduation prerequisites, curriculum structure, instructional activities, educators, and educational materials. The orientation of schools sets the goals for professional development; these goals dictate the graduation prerequisites for students; these prerequisites shape the educational syllabus; and the educational system itself dictates the structuring of instructional activities, educators, and the distribution of educational materials. Cultivating a culture of ongoing enhancement involves creating a permanent assessment system and persistent advancement. The goals of cultural development, criteria for graduation, and the educational methodology are assessed; every educator bears the duty for ongoing enhancement; and the impact of this ongoing progress is evident in the students' academic achievements [5].

Software testing course is the core course of software engineering, and the teaching goal of the course is to cultivate software testers, and both theoretical teaching and practical teaching should be carried out around this goal [6]. Lately, as China's informationization construction and software sector swiftly evolves, the significance of software testers in software creation has escalated, coinciding with a surge in the local testing industry. The primary focus of this course's instruction is on nurturing marketable skills. Consequently, it's crucial to apply the three key principles of engineering education's professional accreditation in setting goals, instructing in content and methodologies, conducting evaluations, and other facets of the software testing technology course. This approach aims to enhance students' understanding of the software's quality in this discipline, to grasp the fundamental principles of software testing and its associated technologies, and to develop students' proficiency in analyzing and assessing intricate engineering challenges within the software domain. Training students in the analytical and evaluative skills for intricate engineering challenges in the software sector is crucial.

2. Design of Course Objectives

Software Testing Technology course is a professional core course for software engineering majors, which focuses on the teaching of principles, methods and related technologies in software testing, including black-box testing, white-box testing, unit testing, integration testing, system testing, acceptance testing, object-oriented software testing, the use of software testing tools and other aspects. Combined with the course's support for professional graduation requirements and its index points, the specific objectives of the course are designed as follows:

1) To be able to systematically master the basic concepts and theories of software testing, to learn the methods of software testing, the use of software testing tools, the management of the software testing process, and to utilize professional knowledge to evaluate the testing solutions of software applications;

2) Be able to apply theories related to software testing, analyze literature to find multiple solutions for testing software applications, and correctly describe the solutions used;

3) Ability to develop test scripting tools with the help of automated testing techniques to evaluate engineering projects to be tested according to the needs of actual software testing projects.

3. Instructional Design of Software Testing Course Based on Engineering Education Accreditation

3.1 Design of Theoretical Teaching

The reform of software testing technology course for engineering education accreditation requires a change in thinking, updating the teaching concept, further revising and improving the course syllabus, and adjusting the course teaching mode and content. Our university has integrated and optimized the teaching content of software testing course, taking into account the requirements for engineering education accreditation and talent cultivation in the field of software engineering. The flipped classroom teaching model has been introduced. Basic theoretical knowledge is taught through software testing engineering project cases. The smart teaching platform "Xiaoya" is used to publish course resources before class. Topic discussions, classroom quizzes, etc. are conducted during class. Online interactive Q&A are conducted after class. In this way, a combination of online and offline course teaching is achieved.

The theoretical teaching content and requirements are as follows:

1) Overview of software testing: understand the work of a software test engineer; master the software testing process; master the basic ideas of software testing; understand the methods of constructing test cases.

2) Boundary value testing: master the boundary value analysis method in software testing; master the connotation of the graph model for software testing; understand the boundary value analysis, boundary value test testing, robustness boundary value testing and other related methods of selection and significance; master the basic principles of boundary value testing.

3) Equivalence class testing: master the division of equivalence classes; master the traditional equivalence class testing methods; master the improved equivalence class testing methods; skillfully use the methods of equivalence class testing to realize the testing of actual programs; understand the edge testing; master the principles and considerations of equivalence class testing.

4) Decision table based testing: master the definition of decision table; master the process and strategy of decision table testing; master the relationship between the causal diagram method and the input domain division constraints; master the decision table testing techniques; master the principles and precautions of decision table testing.

5) Path testing: understand program diagrams;

master the division of DD paths; master the basic steps of base path testing; master the basic principles and considerations of path testing.

6) Data Flow Testing: knowledge of the definition and use of data flow testing; knowledge of program slicing based testing; knowledge of some program slicing tools; knowledge of algorithms for test case selection based on program slicing.

7) Integration testing: grasp the significance of software testing adequacy metrics; familiarize yourself with the significance of stakes and drivers in testing; grasp the testing adequacy of different coverages; understand the differences and connections between different coverage standards.

8) System Testing: grasp the significance of clues inside system testing; grasp model-based clues; coverage metrics for system testing; understand the difference between long use case and short use case testing; understand the methods associated with system testing.

9) Object-Oriented Testing: understand the concepts and basic methods about object-oriented testing; understand the differences between object-oriented testing and traditional testing, and familiarize with the different levels of testing methods in object-oriented testing.

3.2 Task-Driven Practical Teaching

The main features of task-driven teaching method are task-oriented, teacher-guided and student-themed. This teaching mode is fully consistent with the student-centered engineering education training model. Software testing course is a highly practical course, students need to master specific testing methods and the use of testing tools. Software testing itself is exploratory in nature, which requires students to take the initiative to apply the theories they have learned to comprehensively design test cases and carry out testing activities. Taking into account the graduation requirements and indicators of the course, the practical content of the course can be divided into two parts: basic practice and extended practice, as shown in Table 1.

Table 1. Practical Teaching Content of Software Testing Course

Teaching Practice	Experimental Content	Experimental Purpose
Basic Practice	Black-box testing: boundary value method, equivalence class method, decision table method, error speculation method, etc.	Master basic testing skills and test case design methods, and be able to carry out testing activities for simple problems. Master the basic professional knowledge in the field of software engineering required to solve complex software engineering projects.
	White-box testing: logical coverage testing, independent path testing, etc.	
	Usage of testing tools: JUnit unit testing tool, RFT functional testing tool, RPT performance testing tool, RTM test management tool, etc.	Understand and master the use of current mainstream testing tools.
Extended Practice	ATM functional testing, library management system testing, university teaching management system development and testing, etc.	Develop the ability to analyze and design complex engineering problems.

In the above two types of practical teaching, basic practice refers to the course supporting experiments, which aim to require students to master basic testing skills and test case design methods, and be able to carry out testing activities for simple problems. In addition, considering the practical characteristics of software testing, students need to understand and master the use of current mainstream testing tools.

In the extended practice, students need to take the initiative to complete the tasks under the guidance of the teacher, with the aim of cultivating their ability to analyze and design complex engineering problems. Different from the basic practice, this part of practical teaching requires students to form project teams, with different students playing different roles in the project (e.g., system architects, system developers, testers, project managers, etc.). Since testing activities run through the whole life cycle of software development, this practice requires team members to work together to complete the whole project process, including requirements analysis, outline design, system implementation, system testing, deployment and acceptance.

3.3 Multiple Assessment Methods

In order to reflect the comprehensive effect of the course and judge the learning ability of students, a comprehensive, objective and diversified assessment method has been reformed, which emphasizes not only the assessment results but also the learning process. Starting from the concept of "student ability output", the overall evaluation score after the reform consists of two parts: phased assessment in class and final assessment after practical experience. And these two parts account for 60% and 40% of the scores, respectively. The phased assessment mainly includes basic assessment and extracurricular assessment. Among them, the basic assessment mainly includes attendance and daily task completion, while the extracurricular assessment focuses on assessing students' participation in various competitions and other aspects. The final assessment mainly includes the individual's course design report and the team's on-site defense.

4. Conclusions

On the basis of analyzing the characteristics of engineering education accreditation, this paper implements the three major concepts of engineering education professional accreditation, and conducts research and design from the aspects of software testing course goal design, theoretical teaching content, practical teaching content, and teaching evaluation. The aim is to improve the software quality awareness of students in this major, help them master the basic theory and related technologies of software testing, cultivate their analytical and evaluation abilities to effectively solve complex engineering problems in the software field.

In the teaching of this course, we have referred to the standards of engineering accreditation and designed the teaching process, emphasizing teaching output. However, we still need to continue to use engineering education accreditation as the standard, constantly try new methods in teaching, think about new teaching ideas, increase the ability of continuous curriculum reform, and strive to cultivate excellent software testing talents.

References

[1] Chen J., Chen X., Zeng Z., You D. Exploring the Teaching Reform of Centralized Practical Courses under the Background of Engineering Certification and "Three Creations"--Taking Software Quality Assurance and Testing Course Design as an Example[J]. Software Journal, 2023, 22(10): 221-224.

[2] Ding Zhiguo, Wu Jianbin. Research on Teaching Reform of Software Quality Assurance and Testing Course in the Context of Engineering Certification[J]. Computer Education, 2018(5): 33-39.

[3] Zheng H, Li Z, Liu JF. Reform and Practice of Software Testing Technology Course in the Context of Professional Certification of Engineering Education[J]. Computer Knowledge and Technology, 2022, 18(22): 174-177.

[4] Liu Jin, Cheng Yanqin, Wang Yong, Liu Tao, Zhang Xinyang. Teaching Research on Software Testing Technology under the Perspective of Engineering Education Accreditation[J]. Computer Knowledge and Technology, 2020, 16(11): 47-48.

[5] Wang, Antelope Yi, Tian, Linlin, Han, Zhimin, and Li, Ying. Assessment Method of Software Testing Course Based on Engineering Certification[J]. Heilongjiang Science, 2022, 13(11): 27-29.

[6] Liu Long, Shen Hua, Han Xue, Ju Ernan, Guo Shujie. Research on the Evaluation Method of Graduation Requirements and Course Objectives Achievement Based on Professional Accreditation of Engineering Education[J]. Computer Education,2021(8):175- 180.