ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

Research and Practice on the Reform of Experiment Teaching in Software Engineering Courses Empowered by AI Agents

Linlin Li, liangxu Sun

College of Computer and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, Liaoning, China

Abstract

With the rapid development of artificial intelligence technology, AI agents, as intelligent systems capable of autonomous perception, decision-making, and task execution, are changing teaching models. In response to the issues in traditional software engineering course laboratory teaching, such as insufficient practical skills development, low student participation, and lack of personalized guidance, propose an AI agent-based experimental teaching reform scheme. It constructs a teacher-student-AI agent collaborative teaching model, deeply integrating AI agents into the entire experimental teaching process, including core aspects such as intelligent code generation, real-time error diagnosis, automated code review, and personalized learning support. By comparing the effects of two rounds of experimental teaching, it was found that students' project completion, code quality, team collaboration skills, and satisfaction with the teaching model were all significantly improved. This study provides a reference teaching reform path and practical scheme for software engineering course experimental teaching in the AI era.

Keywords

AI agents; software engineering; experimental teaching; teaching reform.

1. Introduction

Software engineering is an engineering discipline with high practical requirements. The experimental teaching component of its professional courses plays an important role in cultivating students' engineering practice abilities, teamwork skills, and innovative thinking[1-2]. Traditional experimental teaching models in software engineering courses generally have some problems[3-4]. For example, the experiment content is disconnected from actual enterprise needs; students have difficulty accessing real software development scenarios; teacher guidance resources are limited, making it hard to meet students' personalized learning needs; the experimental evaluation system is single, making it difficult to comprehensively assess students' overall abilities; students lack sustained motivation to learn, and participation and completion rates in experiments are low[5-7].

In recent years, artificial intelligence technologies represented by large language models have made rapid progress. AI agents can autonomously perceive the environment, make decisions, and perform actions. In the field of education, AI agents have already shown enormous application potential[8-10]. For example, the Khanmigo intelligent assistant developed by Khan Academy based on GPT-4 is already used by students worldwide; Tsinghua University has developed a full-featured intelligent agent platform covering multi-level teaching needs; Xi'an Jiaotong University has developed the JiaoxiaoZhi educational intelligent agent platform, creating teacher-student-specific educational intelligent agent applications.

In the experimental teaching of software engineering professional courses, the application prospects of AI agents are particularly broad[11-13]. Intelligent programming assistants such

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

as GitHub Copilot have been widely applied in real development scenarios. Students using AI-assisted tools perform better in programming tasks and have stronger learning motivation. However, there is still relatively little research and practice on systematically integrating AI agents into the experimental teaching of software engineering courses, and a complete theoretical framework and practical implementation plan are lacking.

This paper studies the path of empowering software engineering professional course experimental teaching with AI agents, constructing a teaching model of AI agent + project-driven + blended learning, and systematically addressing the problems existing in traditional experimental teaching. Through statistical analysis and effectiveness assessment of the experimental teaching process over several semesters, the effectiveness of the experimental teaching reform program is verified, providing reference and guidance for experimental teaching in computer-related professional courses.

2. AI Agents and Their Application Value in the Field of Education

AI agents refer to intelligent entities that can perceive the environment, make decisions independently, and perform corresponding actions to achieve specific goals. Unlike traditional AI tools, AI agents possess characteristics such as autonomy, reactivity, proactiveness, and sociality, enabling them to continuously learn and optimize their behavior strategies in complex environments. The core technical architecture of AI agents includes perception modules, cognitive modules, decision-making modules, and execution modules.

In the field of education, AI agents have significant application value. First, they can provide personalized learning support by analyzing students' learning behaviors and knowledge mastery, customizing learning paths, and recommending learning resources. Second, they can serve as intelligent teaching assistants, handling repetitive tasks such as homework grading, question answering, and learning evaluation, thereby freeing up teachers' time. Third, they can create immersive learning environments through multi-modal interactions and situational simulations, enhancing the enjoyment and engagement of learning. Fourth, they can scientifically analyze learning outcomes, tracking and evaluating students' learning processes comprehensively based on big data technologies.

3. Analysis of Problems in Experimental Teaching of Software Engineering Courses

This section analyzes the problems in traditional experimental teaching of software engineering courses, mainly manifested as follows:

(1) Single experimental teaching mode

Traditional experimental teaching for software engineering courses mainly uses verification experiments and course design formats, where students complete predetermined tasks following the steps in the experiment manuals, lacking space for independent exploration and innovative practice. Students often know the procedures but not the underlying principles, making it difficult to apply the knowledge flexibly to solve practical problems. Most students feel that the experiments are monotonous, lack challenge, and fail to stimulate learning interest.

(2) Insufficient personalized guidance resources

Software engineering course experiments are usually conducted in small group collaborations, with one teacher needing to guide multiple groups simultaneously, making it difficult to respond promptly to each student’s individual needs. When students encounter problems during experiments, they often have to wait a long time for teacher guidance, affecting learning efficiency and experience. Since students have significant differences in foundational

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

knowledge and learning progress, a uniform guidance approach cannot meet personalized learning needs.

(3) Experimental content disconnected from real enterprise practice

The content of traditional software engineering course experiments often comes from textbooks, creating a significant gap with actual enterprise development scenarios. During school, students have little exposure to real software project requirements, development processes, and quality standards, resulting in a long adaptation period after graduation. There is a noticeable gap between students' engineering practice and teamwork abilities and job requirements.

(4) Incomplete evaluation system

Traditional experimental evaluation for software engineering courses is mainly based on experiment reports and project presentations, which cannot comprehensively reflect students' performance and abilities during the experimental process. The evaluation criteria are highly subjective, lacking an objective and quantified indicator system. The feedback is not timely, making it difficult for students to adjust learning strategies based on responses.

Currently, research on experimental teaching in software engineering courses mainly focuses on several areas. First is teaching mode reform, such as project-driven teaching, case-based teaching, and flipped classrooms; second is practical ability cultivation, emphasizing engineering activities centered on delivering operational software; third is industry-education integration, introducing real software projects and enterprise resources through school-enterprise cooperation; fourth is evaluation system optimization, constructing a diversified evaluation mechanism combining process-based and result-based assessment.

In recent years, the application of AI technology in software engineering education has gradually gained attention. By applying AI programming assistants like GitHub Copilot in programming teaching, it has been found that students using AI tools perform better in programming tasks and show stronger learning motivation. Although AI agent technology has achieved preliminary results in the educational field, systematic research on experimental teaching of software engineering courses remains insufficient. First, there is a lack of a theoretical framework for deeply integrating AI agents with software engineering course experiments; second, there is a lack of large-scale, long-term teaching practice verification; third, there is a lack of detailed data support and effect evaluation system.

4. Objectives and Framework of AI Agent-Empowered Experiment Teaching for Software Engineering Courses

The AI agent-empowered experiment teaching of software engineering courses centers on students, is competency-oriented, and leverages AI as an enabler. It constructs a new experiment teaching model of teacher-student-AI agent collaboration to enhance students' software engineering practice abilities, enabling them to independently complete the development of complex software systems; cultivate students' AI tool application skills and critical thinking; strengthen students' teamwork and communication abilities; and stimulate students' learning interest and innovative awareness.

The overall framework of the AI agent-empowered experiment teaching reform for software engineering courses is shown in Figure 1, comprising four layers: input layer, AI agent core engine layer, functional module layer, and supporting layer.

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

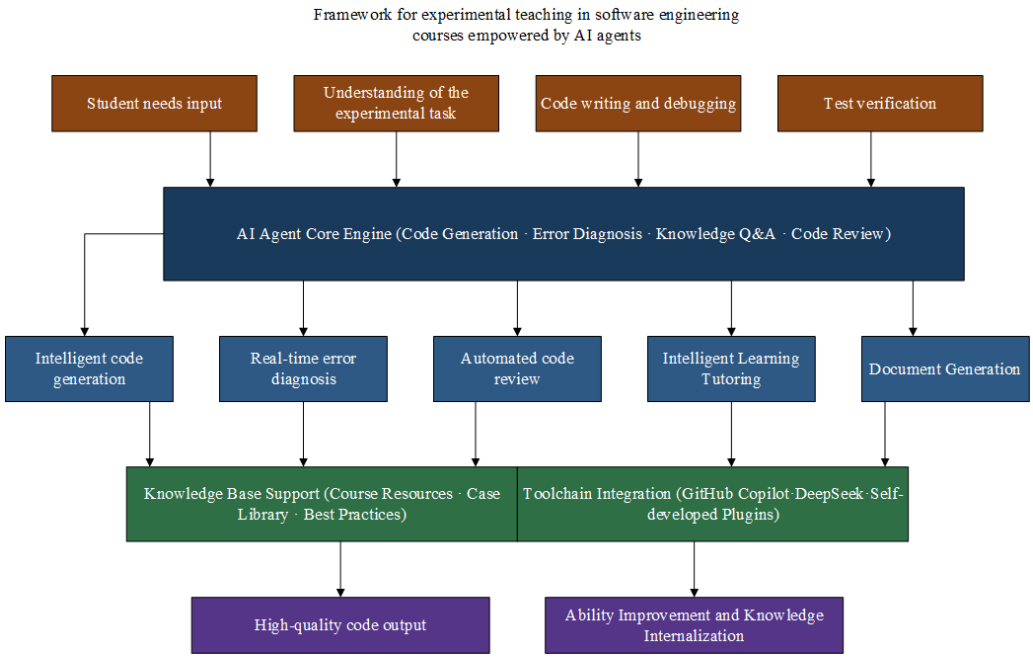

Figure 1. Framework for experimental teaching in software engineering courses empowered by AI agents

During the experiment process, the input layer provides students with system input interfaces such as student requirements, task comprehension, coding and debugging, and test verification. The AI agent core engine layer is responsible for understanding student inputs, calling corresponding functional modules, and generating output results. The functional module layer includes intelligent code generation, real-time error diagnosis, automated code review, intelligent learning guidance, and document generation. The supporting layer provides knowledge base support and toolchain integration to ensure the effective operation of the AI agent.

Within the framework, the AI agent plays multiple roles. First, as an intelligent teaching assistant, it handles repetitive question answering and basic guidance tasks, freeing up teacher resources; second, as a learning partner, it provides students with 24-hour online learning support; third, as a code reviewer, it automates the review and quality assessment of students' code; fourth, as a knowledge engine, it integrates a vast amount of software engineering knowledge, supporting intelligent retrieval and recommendation.

The framework implementation adopts a privatized large model deployment solution, based on open-source large models such as DeepSeek and ChatGLM, to build an on-campus AI agent service platform. The platform integrates mainstream AI programming assistants like GitHub Copilot and Tongyi Lingma and develops or introduces plugins and tools targeting the experiment teaching of software engineering courses. A knowledge base covering course textbooks, experiment guides, enterprise cases, code repositories, and project documentation is constructed, and retrieval-augmented generation technology is used to enhance the professionalism and accuracy of the AI agent.

5. Design and Implementation of Experimental Schemes for AI Agent-enabled Software Engineering Courses

A core comprehensive practical course in software engineering was selected as the target of reform. The course adopts a project-driven teaching mode, and students complete a complete software project in small groups, covering the whole life cycle of the project such as

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

requirements analysis, system design, coding implementation, testing and debugging, and document writing. The content of the project comes from real topics of school-enterprise cooperation, scientific research projects optimized by teachers, and creative projects of students' choice. The types of projects cover various software forms such as web applications, mobile applications, and desktop tools, and each student can find their own professional direction.

Teaching implementation is divided into three stages: pre-class preparation, in-class implementation and after-class improvement. In the pre-class preparation stage, teachers publish preview tasks and reference materials through online platforms, and students learn relevant technical knowledge independently. AI agents provide full-time intelligent Q&A services to answer questions encountered by students during the preview process. At the same time, the AI agent generates personalized preview suggestions based on students' learning conditions. In the implementation stage of the class, classroom teaching focuses on project discussion, difficult problems, design review and code verification. The teacher guides students to conduct in-depth discussions on specific issues in the process of project promotion, and the AI agent assists in recording the discussion content and generating minutes. Use AI agents to assist in code generation and error diagnosis in student coding. In the after-class improvement stage, students complete experimental tasks and project development with the assistance of AI agents. AI agents conduct automated reviews of students' code, pointing out potential issues and suggestions for improvement. Students optimize and iterate based on feedback to continuously improve code quality.

In the whole process of experimental teaching of software engineering courses, a number of AI agent application scenarios are designed, mainly including:

(1) Intelligent code generation

When students write code, they can describe their requirements through natural language, and the AI agent automatically generates a code framework that meets the specifications. Students make modifications and improvements on this basis, greatly improving coding efficiency.

(2) Real-time error diagnosis

When students encounter errors while debugging code, the AI agent automatically analyzes the error message, locates the root cause of the problem, and provides suggestions for fixes. AI agents can also provide preventive guidance for common mistakes.

(3) Automated code review

After students submit code, the AI agent automatically conducts code review, checks for code specifications, potential defects, security vulnerabilities, and other issues, and generates a detailed review report.

(4) Intelligent learning and tutoring

Students can ask questions to AI agents when they encounter conceptual understanding problems during the learning process. AI agents guide students to think deeply through interactive Q&A and do not give answers directly.

(5) Automatic document generation

AI agents automatically generate technical documentation based on project code and comments, including API documents, README files, etc., to reduce the burden of writing documents for students.

(6) Test case generation.

AI agents automatically generate unit test cases based on code logic to help students improve testing efficiency and meet test requirements.

Select multiple semester teaching rounds to compare the teaching practice effect. Among them, senior students used the traditional teaching mode as a control group. The lower grade students

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

used AI agent-assisted teaching mode as the experimental group. The trend of key indicators in the teaching process is shown in Figure 2, and the project completion, student participation, and AI agent usage rate show a steady upward trend.

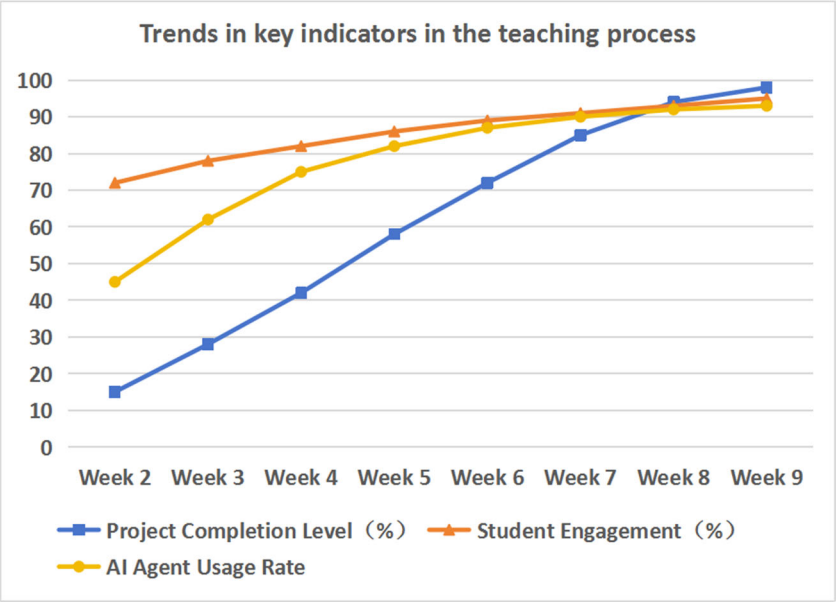

Figure 2. Trends in key indicators in the teaching process

The comprehensive comparative analysis results of the teaching effects of the experimental group and the control group are shown in Figure 3. The experimental group is significantly superior to the control group in all five dimensions: project completion, code quality scores, documentation standardization, team collaboration, and innovation ability.

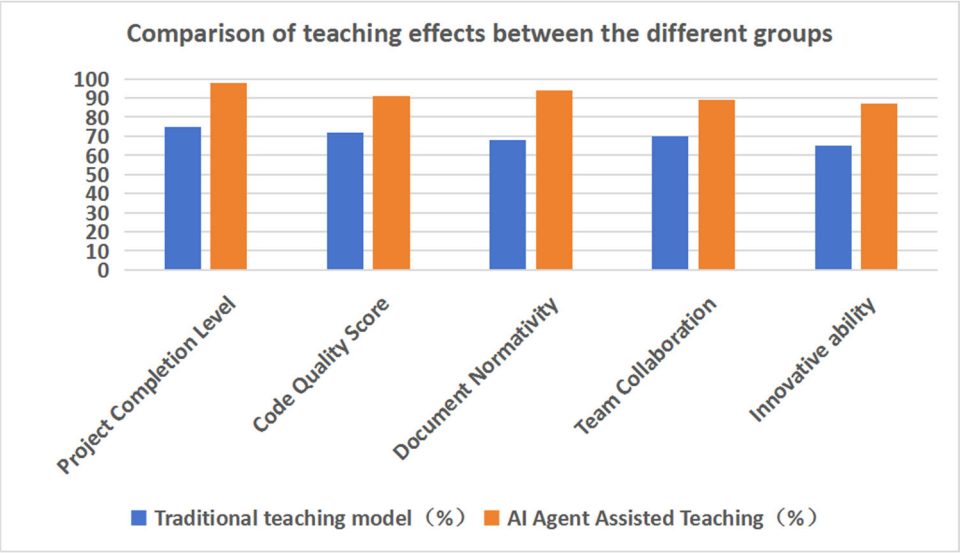

Figure 3. Comparison of teaching effects between the different groups

In order to scientifically and comprehensively assess students' abilities, a multi-dimensional ability evaluation system was designed, including programming ability, system design, teamwork, problem analysis, AI tool application, and engineering practice. By evaluating the abilities of students in the experimental group, it was found that there were significant improvements in multiple dimensions, as shown in Figure 4.

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

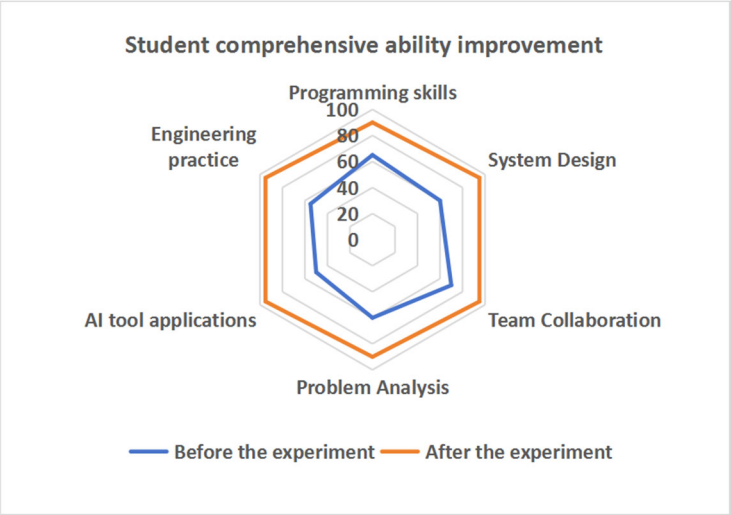

Figure 4. Student comprehensive ability improvement

The quality of students' code was quantitatively assessed using static code analysis tools. The main metrics include code standard compliance rate, unit test coverage, code reuse rate, completeness of comments, and defect density. The code quality metrics of the experimental group showed a continuous improvement trend, as shown in Figure 5.

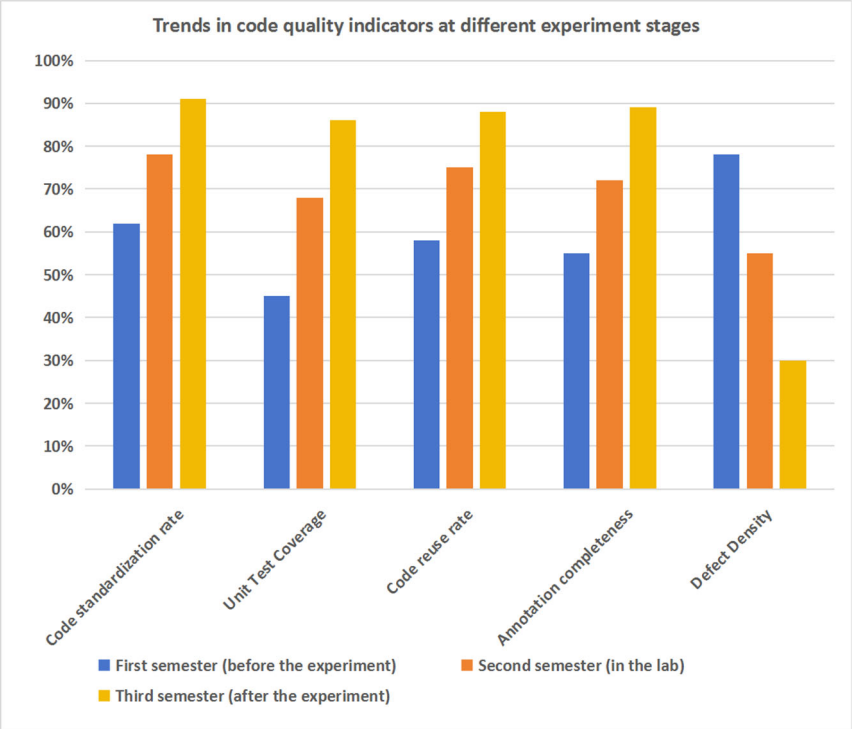

Figure 5. Trends in code quality indicators at different experimental stages

Students were divided into three levels based on their academic ranking: students with weak foundational skills, students with intermediate skills, and outstanding students. The impact of AI agent-assisted teaching on students of different ability levels was analyzed. All three levels of students showed significant improvement after the experiment, with students with weak foundational skills showing the greatest improvement, as shown in Figure 6. AI agent-assisted teaching is particularly beneficial for students with weak foundational skills, helping to narrow the gap in abilities among students and achieve more equitable education.

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

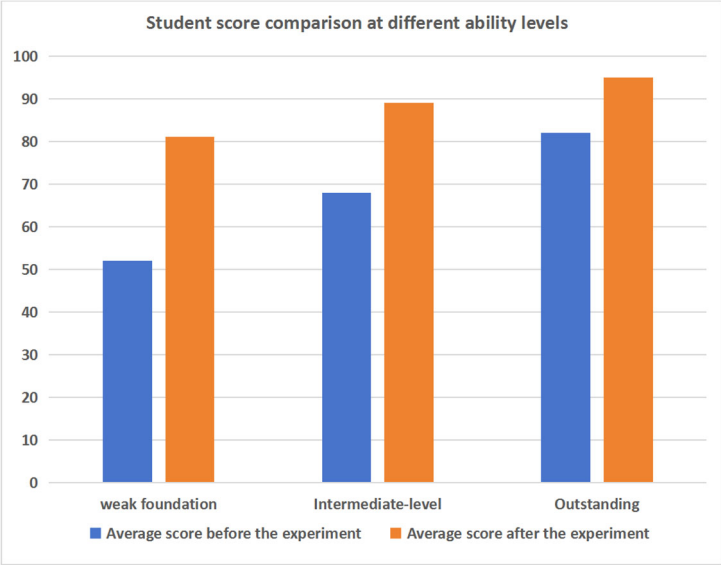

Figure 6. Student score comparison at different ability levels before and after the experiment

The usage statistics of AI agents for code generation, error diagnosis, student Q&A, code review, documentation generation, and test generation are shown in Figure 7, and the survey on satisfaction with AI agent-assisted teaching is shown in Figure 8. Students in the experimental group reported that with the assistance of AI agents, they could quickly obtain effective help and high-quality code suggestions, significantly improving learning efficiency and reducing repetitive work.

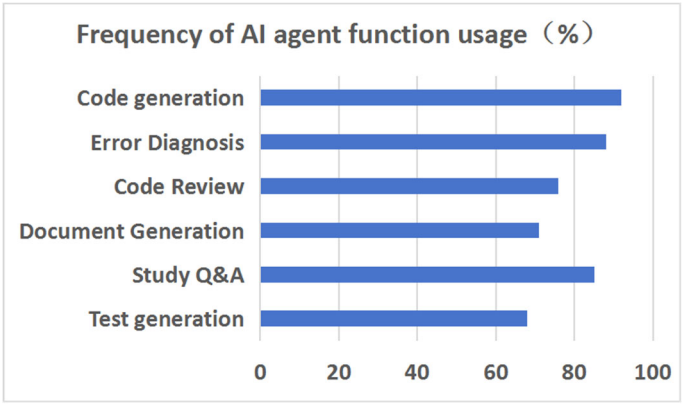

Figure 7. Frequency of AI agent Function usage

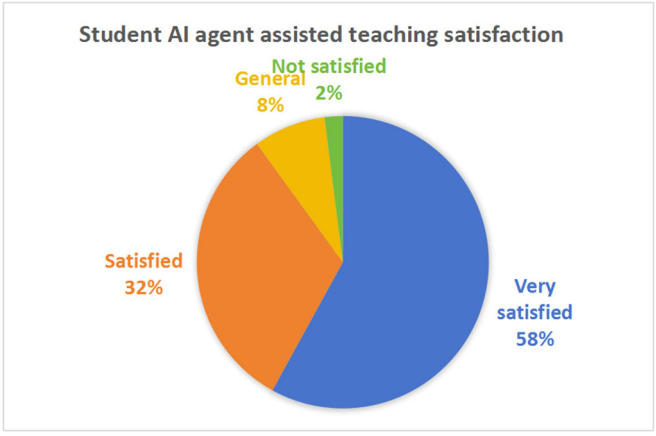

Figure 8. Student AI agent assisted teaching satisfaction

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

The average workload of the experimental group and the control group at each experimental stage is compared in hours. AI-enabled teaching assistance significantly reduced student workload at multiple experimental stages, as shown in Figure 9.

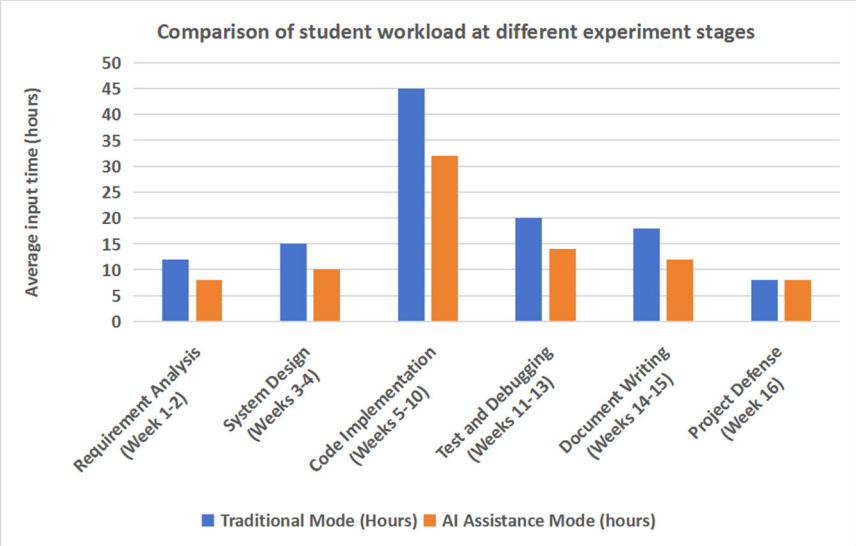

Figure 9. Comparison of student workload at different experiment stages

In summary, the achievements of the above experimental teaching reform are mainly attributed to the following key factors:

(1) Scientific framework design

The teacher-student-AI agent collaborative teaching model fully leverages the respective advantages of teachers, students, and AI agents, forming an effective collaboration mechanism. Teachers focus on instructional design and personalized guidance, students become the main body of learning, and AI agents handle repetitive tasks and basic support work.

(2) Comprehensive technical support

The private deployment of large model platforms and the construction of rich knowledge bases ensure the professionalism and accuracy of AI agents. Integration with mainstream tools such as GitHub Copilot provides students with a convenient user experience.

(3) Reasonable scenario design.

Multiple AI agent application scenarios cover the entire process of experimental teaching in software engineering courses, meeting the diverse needs of students at different stages. Scenario design follows the principle of augmentation rather than replacement and emphasizes cultivating students' critical thinking and problem-solving abilities.

6. Conclusion

Through practical teaching practice, the effectiveness of AI agents empowering the reform of experimental teaching in software engineering courses has been verified. Students' engineering practice abilities have significantly improved, project completion and code quality have greatly enhanced; students' learning enthusiasm and participation have notably increased, and satisfaction with the teaching model has reached a very high level; students' ability to apply AI tools has been effectively cultivated, enabling them to skillfully use AI agents to assist in software development; teachers' teaching efficiency has significantly increased, and students' workload has been greatly reduced.

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

However, there are still some issues that require further research in the future, such as the quality of AI agent responses, students' over-reliance on AI agents, the cost of using AI agents, and the balance between AI assistance and academic integrity.

Teaching must actively embrace change and pursue innovation. By deeply integrating AI agents into experimental teaching, it is possible to cultivate highly qualified software engineering professionals who are better adapted to future demands and make greater contributions to the development of the software industry and the construction of the digital economy.

References

[1] Deng Juan, Peng Rong, Yu Li, et al. Curriculum construction based on school-enterprise collaborative education under the background of engineering certification: A case study of software engineering "knowledge engineering" course [J]. Research in Higher Engineering Education, 2023, (02): 75-79.

[2] Zhou Yanping, Liu Quan, Du Junwei, et al. Innovation and Exploration of Practical Teaching Mode of Software Engineering Based on the Integration of Industry and Education [J]. Software Review, 2025, 24 (04): 191-199.

[3] Liao Yong, Zhou Shijie, Tang Yu, et al. Construction of core curriculum system for software engineering for new engineering [J]. Research of Higher Engineering Education, 2022, (04): 10-18.

[4] Chen Zhigang, Shi Jinjing, Kui Xiaoyan. Discussion on the construction of national first-class undergraduate majors in software engineering under the background of "double first-class" construction [J]. Chinese Journal of University Teaching, 2022, (06): 27-33 40.

[5] Shen Liwei, Peng Xin. Exploration of Software Engineering Teaching Methods Centered on Practical Ability [J]. Software Review, 2023, 22 (12): 1-6.

[6] Zhao Shasha, Tang Xianwu, Zhang Dengyin. Curriculum Reform of Software Engineering Theory and Practice Based on GRADE-CLAP Model: The Three-dimensional Path of "Breaking the Boundary-Reconstructing-Empowering" and Implementing Engineering Certification [J]. University Education, 2025, (23): 31-35.

[7] Hong Mei, Yan Binyu, Yu Jing. Curriculum Teaching Design for Students' Ability Cultivation: A Case Study of Software Engineering [J]. Chinese Journal of University Teaching, 2022, (07): 39-44.

[8] Chen Zhigang, Sun Yiming. New Exploration on Software Engineering Education and Teaching Reform under the Background of Artificial Intelligence [J]. Computer Education, 2026, (03): 1-2.

[9] Wu Fati, Gao Shurui. Paradigm Shift of Classroom Teaching in the Intelligent Era: The Structure and Pattern of Human-Intellectual Integration [J]. Journal of Electronic Education, 2026, 47 (01): 75-83.

[10] Ma Gang, Gulinaz. Research on AIGC-empowered teaching mode of software engineering practice course based on AI-TPCL framework [J]. Computer Knowledge and Technology, 2026, 22 (03): 146-149 160.

[11] Mao Xinjun, Dong Wei, Yin Liangze, et al. Exploration of Large Language Model Assisted Software Development Practice [J]. Research of Higher Engineering Education, 2025, (06): 33-38.

[12] Xu Dan, Shi Jinlong, Qian Qiang, et al. Practical Exploration of Teaching Conversation Tutoring System Empowered by Large Model and Agent [J]. Laboratory Research and Exploration, 2026, 45 (01): 161-167.

[13] Huaxia, Xue Rui, Lei George, et al. Construction and Practice of Hybrid Future Learning Center in Colleges and Universities Based on AI Large Model [J]. Laboratory Research and Exploration, 2026, 45 (03): 222-227.

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202605_9(5).0009

[14] Yang Shanshan, Lin Tao, Ma Xinjuan. Teaching Reform and Practice of Case-Based Intensive Learning Curriculum [J]. Laboratory Research and Exploration, 2025, 44 (05): 169-173.

[15] Yang Jian, Qiang Yan. Software Engineering Teaching Reform with AI-Empowered OBE Concept [J]. Computer Education, 2025, (08): 46-51.