Teaching Reform of Operating Systems Course for Cultivating Innovative Talents

Dong Wang* and Chenpu Li

school of cyberspace, Hangzhou dianzi university, Hangzhou, Zhejiang, 310018, China

*wangdong@hdu.edu.cn

Abstract. Amid the current wave of digitalization and intelligent development, the Operating Systems course, as a core subject in the Computer Science and Technology curriculum, plays a crucial role in fostering innovative technical talents. However, traditional teaching methods often fall short of meeting the demands for developing students' innovative and practical abilities. This paper analyzes the existing issues in conventional teaching approaches for the Operating Systems course and proposes a series of reform strategies aligned with the goals of cultivating innovative talents. The proposed solutions offer practical pathways for advancing the teaching of Operating Systems.

Keywords: Teaching program, Teaching content, Construction of curriculum teaching system, Training of innovative talents.

1 Introduction

The Operating Systems course is one of the most essential core courses in the Computer Science and Technology curriculum, often referred to as the "soul" of computer technology. It plays a pivotal role in bridging foundational knowledge and advanced applications within computer systems. Operating systems serve as the critical bridge between hardware and software, providing resource management, task scheduling, and essential system services.

More importantly, the Operating Systems course is indispensable for cultivating innovative talents. It develops students' systematic thinking abilities, enabling them to understand the design and operation of computer systems comprehensively. By guiding students to explore system optimization strategies, the course stimulates their innovative thinking and problem-solving skills.

Thus, the Operating Systems course is not only a cornerstone for solidifying theoretical knowledge but also a vital platform for enhancing practical abilities, fostering a spirit of innovation, and shaping professional competencies. Its irreplaceable significance in modern higher education underscores its role as a foundation for cultivating future-ready talents in the field of computing [1].

R. Koul et al. (eds.), *Proceedings of the 2nd International Conference on Educational Development and Social Sciences (EDSS 2025)*, Advances in Social Science, Education and Humanities Research 924,
https://doi.org/10.2991/978-2-38476-400-6_59

2 Current Issues in Operating Systems Course Teaching

With the rapid advancement of digital transformation and emerging technologies, fields such as artificial intelligence, big data, cloud computing, and the Internet of Things (IoT) have raised higher expectations for computer science professionals. Modern enterprises demand graduates who not only possess a solid theoretical foundation but also have the ability to design efficient system solutions from an innovative perspective. However, current Operating Systems course teaching in universities falls short of meeting these expectations and fails to fully achieve the goal of cultivating innovative technical talents [2-3].

2.1 Disconnect Between Theory and Practice

Traditional teaching approaches primarily emphasize theoretical instruction, focusing on the systematic and comprehensive delivery of knowledge while neglecting its integration with practical application. Students often passively absorb information in class and struggle to apply their knowledge to real-world problems. Although practical experiments are included in the curriculum, they are usually limited to simple code implementations or functionality verifications, lacking the complexity and diversity found in real-world operating system development. This deprives students of opportunities for in-depth exploration and innovation.

2.2 Monotonous Teaching Content and Methods

Current syllabi for Operating Systems courses feature relatively rigid and uninspiring teaching content and methods. Instructors typically adhere to fixed textbook material, making it difficult for courses to align with rapidly evolving cutting-edge technologies and industry needs. As a result, students often lose interest and engagement in the subject. Furthermore, the lack of heuristic and inquiry-based teaching methods diminishes students' curiosity and stifles their innovative thinking, reducing the learning process to rote memorization of concepts and knowledge points.

2.3 Limited Assessment Methods

End-of-term assessments are a critical component of the teaching process, as they measure the effectiveness of instruction. However, existing evaluation systems place excessive emphasis on final exam results, relying on a single-dimensional standard to assess students. This approach often overlooks students' performance in experimental design, project development, and teamwork. Such summative assessment methods fail to reflect students' comprehensive learning process and skill levels and lack mechanisms to encourage innovation during the course.

2.4 Insufficient Practical Resources and Support

Operating Systems courses often face limitations in practical resources and teaching support. These courses require substantial software and hardware resources, such as virtual machines and embedded development environments. However, many universities lack the necessary resources to provide sufficient hands-on opportunities for students. Moreover, due to gaps in training and practical experience, teaching staff may find it challenging to incorporate the latest technological developments and industry case studies into their lessons, further exacerbating the disconnect between course content and industry demands.

In light of these challenges, this paper proposes a comprehensive teaching reform plan based on course objectives, contemporary educational technologies, and industry requirements. By optimizing teaching models, enriching practical components, and improving evaluation mechanisms, the proposed reforms aim to enhance students' innovation capabilities and practical proficiency.

3 Teaching Reform Strategies for Cultivating Innovative Talents

The measures for scientifically reforming the Operating Systems course to cultivate innovative talents include the following aspects:

3.1 Implementing a Project-Driven Teaching Model

Project-driven teaching is a learning approach centered around real-world projects, effectively enhancing students' hands-on and innovative abilities [4]. In the Operating Systems course, practice-oriented projects linked to real-world scenarios can be introduced. Examples include designing and implementing a simplified process scheduling system, developing a file management system with customized functionalities, or improving certain functional modules of existing open-source operating systems.

In course design, instructors should clearly define project objectives and provide initial guidance, enabling students to independently complete the entire process of requirements analysis, solution design, coding, and functional testing. Through this project-driven approach, students not only master core operating system concepts but also cultivate teamwork skills and innovation awareness while solving problems.

3.2 Enriching Diverse Practical Components

To address the issue of monotonous traditional experiments, teaching reform should increase the diversity and progression of practical activities.

- **•Three-Level Practice Structure:** Establish a framework consisting of foundational experiments, advanced experiments, and comprehensive projects, allowing students to consolidate their knowledge through a step-by-step process.

- •**Interdisciplinary Joint Projects:** Introduce cross-disciplinary projects that integrate emerging fields such as artificial intelligence or the Internet of Things (IoT), enabling students to design operating system functionalities relevant to these areas.
- •**Participation in Competitions and Open-Source Projects:** Encourage students to engage in programming competitions or contribute to open-source projects. For instance, they could participate in memory management optimization contests or collaboratively develop open-source tools, thereby enhancing their comprehensive skills and innovative capabilities [5].

3.3 Improving a Multi-Dimensional Evaluation Mechanism

To comprehensively assess students' learning processes and abilities, the traditional reliance on final exams as the sole evaluation method should be replaced by a multi-dimensional system.

- •**Process Evaluation:** Focus on students' performance in classroom participation, experiment completion, contributions to project development, and innovative highlights, providing dynamic, continuous assessments.
- •**Outcome Evaluation:** Beyond assessing experiment reports and code quality, include result presentations, peer reviews, and instructor evaluations to comprehensively evaluate the overall effectiveness of projects.

This multi-dimensional evaluation mechanism can better reflect students' abilities and potential while motivating their enthusiasm and innovation throughout the learning process.

3.4 Strengthening Practical Resources and Faculty Development

To support the implementation of innovation-driven practical teaching, universities must invest in adequate resources and faculty development:

- •**Enhanced Practical Resources:** Provide sufficient software and hardware resources, such as high-performance computing equipment, virtual experiment platforms, and embedded development tools, to support complex practical tasks.
- •**Industry Collaboration:** Establish mechanisms for collaboration with enterprises, incorporating real-world demands and case studies into course content to ensure alignment with industry developments.
- •**Faculty Training:** Regularly involve instructors in technical training and practical projects to keep them updated with the latest technological advancements. For instance, faculty members could participate in open-source project development or invite industry experts to deliver lectures and provide guidance, helping students understand the current challenges and innovation trends in operating system development.

Through the above teaching reform measures, the Operating Systems course will transcend traditional theoretical instruction to become an essential platform for

inspiring students' innovative thinking, enhancing practical skills, and fostering comprehensive abilities. These reforms will lay a solid foundation for their future development in the technology sector.

4 Practical Case Study of Course Optimization

4.1 Project Background and Objectives

The core objective of the practical project is to equip students with essential operating system technologies and enhance their ability to solve real-world problems. The project is themed "Designing and Optimizing a Simplified Process Scheduling System" and requires students to integrate theoretical knowledge to design a system module capable of simulating task scheduling in operating systems. Additionally, students must propose optimization improvements based on existing scheduling algorithms. Through this project, students will complete the entire process from requirements analysis to functional implementation and from theoretical validation to optimization, aiming to cultivate their system design skills, innovation awareness, and teamwork spirit.

4.2 Project Implementation Process

(1) Project Design and Grouping

At the beginning of the course, the instructor designs detailed project tasks based on real-world requirements and divides the class into groups of 3-5 students each, with clearly defined roles. The project tasks are structured into three phases:

- •**Basic Tasks:** Implementing fundamental scheduling algorithms, such as First-Come-First-Served (FCFS) and Shortest Job Next (SJN).
- •**Advanced Tasks:** Simulating complex scheduling scenarios.
- •**Innovative Tasks:** Designing and implementing an improved scheduling algorithm.

Students select their roles and responsibilities within the group based on their interests and abilities, fostering teamwork and collaborative skills.

(2) Combined Teaching and Guidance

Throughout the project, theoretical instruction is integrated with hands-on guidance. While delivering lectures, instructors conduct specialized discussions to explain the core principles of scheduling algorithms and introduce case studies, such as modern task scheduling strategies for multi-core processors. Students are encouraged to seek help from instructors or teaching assistants when facing technical challenges, and regular group discussions are held to provide feedback on project progress.

(3) Phase Deliverables and Review

At the end of each project phase, students submit a deliverable report and present their group's progress. During presentations, students explain their algorithm design, experimental results, and improvement proposals. Both the instructor and peers

participate in discussions and evaluations. These iterative reviews help students refine their project designs while enhancing their presentation skills and critical thinking.

(4) Final Deliverables and Competitions

At the end of the course, each group submits its final deliverables, including code implementations, experimental reports, and optimization analyses. Exceptional projects are recommended for participation in university- or provincial-level innovation competitions. Through these competitions, students experience the process of translating practical results into applications while improving their competitiveness and confidence.

4.3 Expected Project Outcomes

(1) Significant Improvement in Theoretical Understanding and Practical Skills

By adopting a project-driven teaching approach, students gain practical mastery of fundamental theories and key technologies in process scheduling. Compared to traditional theoretical learning, this method allows students to comprehend the core functionalities and implementation principles of operating systems in realistic scenarios.

(2) Substantial Enhancement of Innovation Capability

During the innovation phase, students are encouraged to propose various scheduling optimization algorithms based on priorities. Experimental results demonstrate that these optimized approaches can effectively improve system efficiency, fostering their ability to identify problems, propose improvements, and validate solutions.

(3) Strengthened Teamwork and Comprehensive Skills

The group-based approach promotes teamwork and collaboration among students. Additionally, through phased presentations and competition showcases, students improve their communication, logical reasoning, and problem-solving skills holistically.

This practical case study demonstrates that a project-driven teaching model effectively enhances students' theoretical comprehension, practical capabilities, and innovation awareness. Future teaching efforts will focus on diversifying project types, optimizing practical resources, and exploring innovative tasks linked to emerging technologies. Furthermore, deeper collaborations with industry will be pursued to align the curriculum with real-world demands, contributing to the cultivation of high-quality innovative computer science talents.

5 Conclusion

The teaching reform of the Operating Systems course, centered on project-driven learning, diverse practical activities, and an improved evaluation mechanism, provides a solid foundation for enhancing students' participation, practical skills, and innovation capabilities. Future research will aim to further optimize the reform framework and explore its application in other computer science courses.

References

1. Luo Yu, Wen Yanjun. Construction of Operating System Textbooks to Strengthen System Capability Training [J]. Computer Education, 2024, (11): 18-21.
2. Gong Nana, Cao Yun. Exploration of an Employment-Oriented Curriculum Group for Embedded Systems [J]. Computer and Information Technology, 2024, 32(05): 130-133.
3. Wang Jun. Teaching Reform of the Network Operating System Course Based on a Project-Driven Model [J]. Information and Computers (Theoretical Edition), 2024, 36(18): 42-44.
4. Zhu Lin, Lang Qianwen. Analysis of Issues in the Project-Driven Teaching Model in Computer Teaching Reform at Universities [J]. Computer Products and Distribution, 2019, (06): 185.
5. Liu Jing, Tie Jun, Du Xiaokun. Curriculum Teaching Reform Exploration of "Horizontal Integration, Vertical Extension": A Case Study of Core Computer Science Courses [J]. Higher Education Journal, 2024, 10(34): 139-142.