Empowering with Design Thinking: Teaching Reform and Content System Reconstruction of Programming Courses in an Intelligent Technology Environment

Yandan Xue
School of Media Technology, Communication University of China Nanjing, Nanjing, China

Abstract:Based on the theoretical coupling between design thinking and intelligent technology, this study systematically carries out curriculum teaching reform and content system reconstruction to address the core pain points of traditional programming courses in the intelligent technology environment, such as "emphasizing knowledge over ability" and "disconnection between theory and practice". By using literature research, case analysis, and empirical research methods, the "thinking technology ability" trinity teaching objectives are established, a design thinking oriented project-based teaching model is constructed, a hierarchical modular curriculum content system is reconstructed, and a multi-dimensional integrated full process evaluation system is established. Using the C language programming course as a practical case, 86 new students majoring in computer science were selected to conduct teaching experiments in groups. The results showed that the experimental group had significantly better participation rates in innovation competitions, project quality, and core competency scores than the control group. The course pass rate increased from 70% to 85%, and student satisfaction reached 90%. Research has shown that the deep integration of design thinking and intelligent technology can effectively improve teaching quality and students' comprehensive literacy, provide feasible paths for the cultivation of intelligent technology talents, and have important reference value for similar course reforms.

Keyword: Design Thinking; Intelligent Technology Environment; Programming Courses; Reform In Education; Content System Reconstruction; Multiple Evaluation

1. Introduction

In the current era of comprehensive penetration of intelligent technologies such as artificial intelligence and big data, the demand for compound talents possessing programming skills, innovative thinking, and problem-solving abilities has exploded. Traditional programming courses generally suffer from the drawbacks of "emphasizing grammatical knowledge and neglecting practical application" and "emphasizing one-way indoctrination and neglecting innovation cultivation", resulting in students who "can program but cannot solve practical problems" and are difficult to adapt to the industry's requirements for innovative talents. Design thinking, as a people-oriented innovative methodology, with "demand orientation - interdisciplinary collaboration - iterative optimization" as its core logic, provides a new path for curriculum reform. Deeply integrating it with intelligent technologies can not only address the pain points of the disconnection between theory and practice and the lack of innovation ability cultivation in traditional teaching, but also help students build dual competencies of "technical application + thinking innovation". It is of great value to improving the intelligent technology talent cultivation system and supporting the high-quality development of the digital industry.

Relevant research abroad started earlier. Stanford University integrated design thinking into engineering courses, while the Massachusetts Institute of Technology developed an intelligent programming platform to support personalized learning. However, both lack a systematic integration model of "design thinking + intelligent technology". Domestic research has gradually gained momentum in recent years, with some universities attempting to introduce design thinking or intelligent technology tools. However, there is a problem of superficial application, and a mature system of deep integration of "thinking - technology - content" has not been formed. A systematic reform plan is urgently needed.

This study employs literature research, case analysis, and empirical research methods, with three core innovations: First, it constructs a dual-driven teaching model of "design thinking + intelligent technology" to break through the limitations of a single reform dimension; second, it establishes a hierarchical and modularized course content system to achieve the coordinated cultivation of knowledge, skills, and thinking; third, it designs a diversely integrated whole-process evaluation system that takes into account both knowledge mastery and literacy improvement.

2. Analysis of Existing Problems in Programming Courses under Intelligent Technology Environment

2.1 The Disconnect between Course Content and Intelligent Technology

The course content updates are lagging behind, still focusing on traditional programming language syntax and basic algorithms, with insufficient integration of cutting-edge technologies such as artificial intelligence, big data processing, and IoT programming. Taking Python courses as an example, most textbooks still focus on the basics of syntax, with little involvement in topics closely related to intelligent technology such as TensorFlow framework applications and data analysis visualization, resulting in a disconnect between students' knowledge and the actual needs of the industry.

At the same time, the content system lacks coherence, and each knowledge point is explained in isolation. The practical activities are mostly verification experiments (such as "writing a calculation function"), lacking comprehensive training for real projects. Students find it difficult to form a complete ability chain of "requirement analysis scheme design code implementation testing optimization", and their hands-on ability and engineering literacy are weak.

2.2 One Sidedness of Teaching Evaluation And Feedback

The current teaching evaluation still focuses on "summative exams" as the core, with exam content emphasizing grammar memory and code correctness, accounting for over 70%, neglecting the evaluation of students' comprehensive qualities such as design thinking, innovation ability, and teamwork. For example, even if students can proficiently write sorting algorithm code, they cannot design a reasonable technical solution for the "intelligent attendance system" and can still achieve high scores.

The evaluation subject is single, with only teachers leading the evaluation, lacking student self-evaluation and peer evaluation processes; The feedback mechanism is not sound, and homework grading only gives "correct/incorrect" or simple scores without in-depth analysis of the problems. Students cannot clarify the direction of improvement, and teaching evaluation is difficult to play the role of "promoting learning through evaluation".

3. Teaching Reform Strategies for Intelligent Technology Environment Programming Course Driven by Design Thinking

3.1 Conceptual Reshaping: Establishing the Three in One Teaching Goal of "Thinking Technology Ability"

Breaking through the traditional "knowledge imparting" orientation and building a diverse and collaborative teaching objective system:

At the thinking level: cultivate students' innovative thinking, critical thinking, and systematic thinking, and teach them to use design thinking to analyze and solve complex problems;

Technical level: Master core programming language skills and intelligent technology application abilities, including cutting-edge technologies such as artificial intelligence basics, big data processing, and IoT programming;

Ability level: Enhance practical operation ability, teamwork ability, and self-learning ability, and achieve the training goal of "being able to program, innovate, and solve problems well".

3.2 Mode Innovation: Building a Design Thinking Oriented Project based Teaching Mode

Using real projects in the field of intelligent technology as carriers, fully integrate the five stages of design thinking into the entire teaching process:

1. Project selection: Adhering to industry needs, select practical and challenging project themes such as smart security, smart home, smart campus, etc., such as "Smart Dormitory Electricity Safety Monitoring System";
2. Implementation process: Students work in

groups of 4-6 people, first conducting surveys and interviews to understand user needs (empathy), and then extracting core questions (such as "how to monitor overloaded electricity in real time and issue warnings"); Then generate technical solutions (ideas) through brainstorming, such as "using sensors to collect data and C language programming to implement warning logic"; Subsequently, develop a system prototype (prototype production) and complete code writing and debugging through an intelligent programming platform; Finally, invite users to test and collect feedback for iterative optimization (testing);

3. Teaching support: With the help of online collaboration platforms (GitHub, Tencent Docs) to support cross temporal group collaboration, project progress is tracked through intelligent teaching platforms, and teachers provide targeted guidance throughout the process as "guides".

3.3 Evaluation Reform: Establish a Multi-Dimensional and Integrated Whole Process Evaluation System

Constructing an evaluation system with diverse subjects, comprehensive content, and diverse methods:

Evaluation subject: covering teacher evaluation (60%), student self-evaluation (20%), and group peer evaluation (20%), taking into account professional judgment and peer feedback;

Evaluation content: It includes not only knowledge mastery (code correctness, technical application proficiency), but also design thinking ability (accuracy of requirement analysis, innovation of solutions), practical ability (project completion quality), and collaboration literacy (team contribution);

Evaluation method: Combining process evaluation (accounting for 60%, including classroom performance, project progress reports, and group discussion participation) with summative evaluation (accounting for 40%, including project achievement display and comprehensive skill testing) to comprehensively reflect students' learning outcomes.

4. Refactoring the Content System of Programming Courses in the Intelligent Technology Environment

4.1 Architecture Design of Modular Content System

Build a four layer modular content system, with each module progressively and organically connected:

1. Basic programming modules (accounting for 30%): covering programming language syntax (such as Python/C language), data structures, and algorithm foundations, laying the foundation for programming ability and emphasizing the application of syntax in practical scenarios;

2. Intelligent technology application module (accounting for 40%): Core modules, including artificial intelligence basics (machine learning algorithm introduction, TensorFlow framework application), big data processing (data acquisition, visualization programming), IoT development (sensor data processing), and explaining technical principles and practical methods through case studies;

3. Project practice module (accounting for 20%): Set up three levels of projects: basic (such as "intelligent temperature monitoring program"), advanced (such as "intelligent attendance system"), and innovative (such as "AI based image recognition tool"). Students choose according to their abilities to achieve comprehensive application of knowledge;

4. Expand elective modules (accounting for 10%): covering cutting-edge topics such as quantum computing programming and blockchain applications, as well as case studies in industries such as smart healthcare and smart finance, to broaden students' horizons.

4.2 Support and Guarantee Construction of Content System

1. Textbook construction: Develop characteristic textbooks that integrate design thinking and intelligent technology, highlight the "case+project" orientation, and provide supporting digital resource packages;

2. Practice platform: Build an integrated practice platform of "online programming environment+virtual laboratory+project management tool", supporting students to carry out project development anytime and anywhere;

3. Teacher development: Enhance teachers' intelligent technology application ability and design thinking teaching level through specialized training and enterprise training, and introduce enterprise engineers to teach part-time;

4. Resource library: Integrate high-quality resources such as teaching courseware, micro lesson videos, project cases, online test questions, etc., to form a dynamically updated course

resource library.

5. Practical Cases and Effectiveness Analysis: Taking C Language Programming Course as an Example

5.1 Case Background and Implementation Process

Selecting freshmen majoring in computer science from a certain university (2 classes, a total of 86 students) as the subjects, the experimental group (43 students) adopted the reform plan of this study, while the control group (43 students) continued to use the traditional teaching mode.

The experimental group focuses on the "Intelligent Dormitory Management System" as the core project and implements teaching according to the five stages of design thinking: clarifying students' needs for dormitory management (such as personnel attendance, electrical safety, and repair services) through questionnaire surveys; Defining "developing a low-cost, easy-to-use intelligent management program" as the core issue; Design a technical solution for "data collection logic processing terminal display"; Using C language combined with sensor modules to create prototypes; Invite dormitory administrators and students to test and iterate for 3 rounds of optimization. During the teaching process, personalized tutoring is provided by tracking student code submissions, group discussions, and other data through an intelligent teaching platform.

5.2 Multi dimensional evaluation of reform effectiveness

Student ability improvement: The participation rate of the experimental group in the innovation competition reached 60% (compared to only 30% in the control group), and they won 3 awards in the school's intelligent technology innovation competition (compared to 1 award in the control group); Student self-evaluation shows that the scores for needs analysis, innovative design, and team collaboration ability have increased by an average of 22% compared to the control group. The project results are significantly better than the control group in terms of functional integrity and innovation;

Teaching quality optimization: The pass rate of the experimental group courses has increased from 70% before the reform to 85%, and the excellent rate (above 80 points) has increased from 15% to 33%; According to a questionnaire survey, the satisfaction rate of the experimental group students with the course content and teaching methods reached 90%, an increase of 25 percentage points compared to the control group. Most students reported that they could feel the practical value of knowledge and had stronger learning motivation.

5.3 Case Reflection and Improvement Direction

In practice, two shortcomings were found: firstly, the difficulty gradient setting of the project was not precise enough, and some students with weak foundations lagged behind in the prototype production stage; Secondly, there is a lack of interdisciplinary collaboration resources and professional guidance when it comes to sensor hardware debugging.

Improvement direction: Firstly, optimize the hierarchical design of projects, and add "introductory adaptation tasks" in addition to basic, advanced, and innovative projects, providing step-by-step guidance for students with weak foundations; The second is to strengthen school enterprise cooperation, jointly build interdisciplinary practice platforms with intelligent device enterprises, and introduce enterprise technical experts to conduct special lectures and practical guidance.

6. Conclusion

The "Design Thinking+Intelligent Technology" dual drive teaching reform plan and modular content system constructed in this study effectively solve the core pain points of outdated concepts, disconnected content, and one-sided evaluation in traditional programming courses. Practice has proven that this program can significantly enhance students' innovation ability, practical ability, and learning enthusiasm, while optimizing teaching quality, providing a feasible path for the cultivation of intelligent technology talents.

Acknowledge

QG/2025/BKYB0405-05772 Research on the Reconstruction of Teaching Content and Innovation of Educational Models in Programming Courses Driven by Intelligent Technology and Based on Design Thinking

References

[1] Lu Hui, Gong Zheng, Zhao Junfeng, et al.

Research on the Reform of Python Programming Experimental Teaching Based on the Mutual Infiltration of Artificial Intelligence and Ideological and Political Education in Courses [J]. Software Guide, 2025, 24(11):54-59

[2] Yang Yan, Rao Zhibo. Teaching Reform and Practice of AI-Empowered Circuit Analysis Basic Course Based on Engineering Application Ability Orientation [J]. University Education, 2025(S1):68-72

[3] Zhou Hao, Tan Qingfang. Practical Exploration of AIGC Technology Empowering Design Teaching Mode Innovation Driven by Industry-Education Integration [J]. Printing and Digital Media Technology Research, 2025(z1)

[4] Xu Qin, Qiao Yulong, Wu Zixing, et al. Innovation in Teaching Mode of Programming Courses Enabled by Artificial Intelligence [J]. Journal of China Science and Technology, 2025(1):233-239

[5] Wan Yiliang, Jin Rui, Miao Zelong, et al. Research on the "Three-Dimensional Integrated" Smart Teaching Mode of GIS Spatial Analysis Empowered by Artificial Intelligence [J]. Surveying and Mapping Bulletin, 2025(S2):319-323.

.

