Article

Research on Full-Cycle Teaching Reform of IoT Cloud Platform Application Development Driven by CDIO Model

Yantao He [1,*], Pengteng Huang [1], Liang Yu [1], Xiaochen Zhang [1] and Lanping Zhou [1]

[1] Department of Computer Science, Guangdong University of Science and Technology, Dongguan, Guangdong, China
* Correspondence: Yantao He, Department of Computer Science, Guangdong University of Science and Technology, Dongguan, Guangdong, China

Abstract: This research, guided by the Conceive-Design-Implement-Operate (CDIO) model, emphasizes full-cycle teaching reform and practice within the context of an Internet of Things (IoT) cloud platform application development. The primary focus is on enhancing students' engineering practice capabilities, deeply exploring and implementing the entire "ideation-design-implementation-operation" process in effective teaching methodologies. Through this research, our goal is to fully incorporate CDIO's project-driven concept into professional IoT teaching. This approach enables students to master a comprehensive skill set, ranging from demand analysis to system operation and maintenance, within real project scenarios. Consequently, it enhances their ability to tackle complex engineering problems.

Keywords: CDIO model; IoT cloud platform application development; engineering practice capabilities; project-driven teaching

1. Introduction

"IoT Cloud Platform Application Development" is a core course within the Internet of Things (IoT) engineering discipline and is closely aligned with the cultivation of essential competencies at the "cloud" layer of the IoT technological architecture [1]. The course plays a critical instructional role by bridging foundational hardware knowledge with higher-level application development. Its primary training objectives focus on enabling students to systematically understand the technical architecture, functional characteristics, and operational mechanisms of mainstream IoT cloud platforms, such as OneNet and Alibaba Cloud. At the same time, the course seeks to deepen students' comprehension of the technical principles and practical applications of commonly used cloud communication protocols, including HTTP, EDP, MQTT, and TCP transparency transmission. Through structured training, students are expected to acquire the ability to employ WiFi, 4G/5G, and other communication modules in development tasks, and to master data interaction and collaborative mechanisms between cloud platforms and underlying hardware systems. Ultimately, the course aims to prepare students to independently develop cloud-based applications on hardware devices and to effectively address practical engineering problems.

However, under traditional teaching models, this course often suffers from a pronounced "disconnect between theory and practice," in which students mainly engage in passive knowledge reception, making it difficult to form a systematic engineering mindset [2]. By introducing full-process engineering task design, the CDIO approach can guide students through the complete lifecycle of cloud platform application development, from the conception of cloud-based solutions to scheme design, development and

Published: 24 December 2025

implementation, and finally system operation and debugging. This instructional reform effectively addresses challenges such as students' difficulty in adapting to different cloud platforms and their abstract understanding of communication protocols. It also alleviates the separation between hardware-level development and cloud-level application design, thereby promoting a genuine transformation from "knowledge transfer" to "ability cultivation." Such an approach aligns closely with the course's fundamental objective of fostering composite engineering talents with both technical competence and practical problem-solving capabilities.

The CDIO model is an engineering education framework oriented toward the systematic cultivation of practical engineering abilities. The acronym CDIO represents the four stages of Conceive, Design, Implement, and Operate, and the model was proposed by leading international engineering education institutions in the early twenty-first century [3]. It was developed in response to the limitations of conventional engineering education, in which students often possess fragmented theoretical knowledge but lack integrated practical training. The core feature of the CDIO model lies in immersing students in authentic engineering contexts, enabling them to participate in the complete process of real-world project development. The iterative learning cycle inherent in the CDIO framework emphasizes "learning by doing" and "learning by using," thereby forming an educational ecosystem that integrates theory with practice.

Empirical research based on comparative teaching experiments has demonstrated that the CDIO model can significantly enhance students' practical software development abilities and strengthen their awareness of interdisciplinary collaboration [4]. Evidence from project-based teaching indicates that this approach effectively narrows the gap between academic knowledge and actual enterprise requirements. At the same time, such studies have also revealed practical challenges in implementation, including the need for adequate hardware resources and the requirement for systematic training of instructors in CDIO-related concepts. These findings provide data-based support for further optimization of engineering talent cultivation programs in higher education institutions.

Within the broader context of engineering education reform, studies integrating Outcome-Based Education (OBE) with the CDIO framework have explored diversified instructional strategies in programming and engineering-related courses [5]. Through pedagogical reforms that combine project-driven learning, case analysis, and task-oriented instruction, experimental results show notable improvements in students' practical programming skills and overall engineering thinking. These reforms contribute to closer alignment between educational outcomes and industry needs, while also highlighting the importance of optimizing teaching resource allocation to further enhance instructional effectiveness.

In addition, research focusing on the application of the CDIO framework in engineering courses emphasizes the promotion of a student-centered learning model [6]. By reorganizing course content and designing project-based learning tasks, the CDIO approach has been shown to increase students' recognition of active learning, enhance their problem-solving abilities and innovative practice skills, and strengthen interaction between teachers and students through more personalized guidance. Nevertheless, such studies also point out practical constraints, including the need to balance teaching schedules with individual differences among students, which must be carefully addressed in future teaching reforms.

2. Problems with IoT Cloud Platform Project

2.1. Challenges in Cross-Platform Adaptation of Cloud Platform Technical Architecture

Within the context of the "Internet of Things Cloud Platform Application Development" course, it is essential to guide students to become proficient in the application development for a variety of mainstream cloud platforms, such as OneNet, Alibaba Cloud, and Huawei Cloud. Each platform's autonomy and distinct technical

system design often result in operational obstacles for students when they attempt to work across different platforms. These challenges not only lengthen the Timeline for task completion, but also may reduce students 'enthusiasm for learning, thereby significantly impacting the course's objective of fostering composite engineering practice capabilities. From a technical standpoint, the device access procedures across various platforms do not adhere to unified standards. Some platforms necessitate the creation of a device model followed by access authentication, while others mandate the initial configuration of product keys before associating them with hardware devices. These discrepancies in procedural order can lead to operational confusion when transitioning between different practice platforms. Additionally, the lack of consistency in API interface specifications amplifies the challenge of adaptation. Different platforms have distinct requirements for data interaction formats, such as JSON parameter naming and request header configurations, as well as variations in interface call methodologies, including the scenarios in which GET/POST requests are applicable. As a result, students often find themselves frequently modifying code logic to accommodate the nuances of each new platform. This not only adds to their learning overhead but also increases the likelihood of interface call failures due to misconfigured parameters.

Moreover, discrepancies in rights management mechanisms across various platforms have emerged as significant impediments. Certain platforms employ hierarchical authorization to regulate the extent of data access, which includes both device-level and product-level permissions. In contrast, others depend on token timeliness to restrict operational permissions. When students do not promptly comprehend the authority configuration rules inherent to a new platform, there is a propensity for the device to malfunction due to inadequate permissions. This can lead to issues such as disrupted data transmission and abnormal data formatting resulting from incorrect authorization configurations. Such challenges significantly impede the practical utility and limit students' comprehensive understanding of the cloud platform's technical system and their ability to apply it flexibly.

2.2. Communication Protocol Abstraction Principles Are Out of Touch with Engineering Practice

In the instruction of the "IoT Cloud Platform Application Development" course, a key hindrance to the enhancement of students' technical application abilities has been the disconnection between the abstract principles of communication protocols and practical engineering applications. The course involves core theories such as the RESTful interface design of HTTP/HTTPS, the Topic subscription mechanism of MQTT, and the security authentication process of the EDP protocol. These concepts are inherently abstract. Nevertheless, traditional teaching methods often emphasize theoretical explanations, with inadequate support for real device access. This lack of practical scene support hinders students from establishing a coherent understanding between theory and practice.

From observing the distinct pain points in learning, it appears that students frequently make incorrect judgments about the application scenarios of agreements. For instance, the design of the RESTful interface emphasizes a "resource-oriented" request logic. However, students' understanding is often limited to theoretical study, making it challenging for them to comprehend its limitations in scenarios requiring high frequency and low latency, such as "device status query". Consequently, they might inadvertently select this protocol over MQTT during actual development, despite the latter being more suited for real-time communication. Furthermore, while students may grasp the basic concept of MQTT's "publish-subscribe" mechanism, their lack of exposure to real-world scenarios where multiple devices connect simultaneously hinders their understanding of how Topic level design (e.g., "device/area/number") impacts data screening efficiency. This leads to issues such as data stream interruptions or missed collections during transmission from multiple devices.

2.3. Technical Gaps in Hardware-Cloud Interactive Development

Within the practical teaching framework of the "Internet of Things Cloud Platform Application Development" course, a prominent technical disconnect persists in hardware-cloud interactive development, serving as a critical bottleneck that impedes students from establishing a seamless development pipeline spanning the "underlying hardware-cloud platform" ecosystem. This interactive development paradigm encompasses a suite of core tasks, including AT command debugging for WiFi/4G modules, serial communication configuration, and data packetization and reassembly under TCP transparent transmission mode. These tasks inherently demand the integration of dual competencies-embedded programming proficiency and hardware debugging expertise. Regrettably, a majority of students exhibit deficiencies in both domains, creating a fundamental barrier to achieving reliable connectivity and data interoperability between hardware endpoints and cloud platforms.

From the perspective of practical development workflows, students' inadequate foundational skills in hardware debugging significantly hamper the preparatory phase of projects. During the debugging of WiFi/4G modules, for instance, students are required to configure module parameters (such as baud rate and access point credentials) via AT commands. However, due to their limited hands-on experience with embedded hardware systems, they frequently encounter issues such as unresponsive command execution and erroneous parameter configuration. A typical scenario involves incorrectly setting a module's baud rate to 9600 while attempting to communicate with a microcontroller unit (MCU) at a baud rate of 115200. This mismatch leads to garbled serial data transmission, effectively preventing the establishment of an initial communication link between the module and the hardware, and stalling subsequent development progress.

3. Full-Cycle Teaching Reform Based on CDIO Model

3.1. CDIO Four-Phase Project Driving System

This study introduced CDIO model teaching, reformed Cloud Platform Application Development, and established a full-cycle-four-stage teaching system, as shown in Table 1 below.

Table 1. A full-cycle-four-stage teaching system.

Project	Conceive	Design	Implement	Operate
Mapping Association Modeling Method of Abstract Protocol Principles and Hardware Devices	Clarify teaching goals and learning outcomes, and build industry-oriented quantitative + qualitative evaluation dimensions	Plan the "protocol-hardware" mapping knowledge chain, integrate teaching resources, and design project-based/case-based/interdisciplinary teaching activities	Combine industry scenarios and enterprise project implementation teaching, integrate ideological and political affairs, and adopt flip classroom+ virtual and practical practice strategies	Build a diversified evaluation system, generate a capability radar map, and realize a closed-loop teaching through timely feedback and dynamic optimization
Scenario Case Development of "Protocol Selection-	Aiming at the pain points of cloud platform adaptation and	Build a case development knowledge chain, integrate resources, and	Combine industry scenarios with implementation	Build a diversified evaluation system,

Data Analysis-Exception Handling" Full-process	protocol application, clarify the multi-platform full-process case development goals and establish enterprise-oriented evaluation standards	design task-driven development and case sharing activities	case development, group collaboration and supporting technical support, and organize competitions to stimulate enthusiasm	integrate quantitative and qualitative indicators, and achieve a closed-loop teaching through feedback and iteration
Application Practical Assessment Mechanism Based on Fault Injection	In response to the pain point of "knowing theory but not eliminating obstacles" in hardware cloud development, clarify comprehensive ability assessment goals and establish technical + professional literacy evaluation standards	Design assessment content and scenarios in layers according to fault complexity, integrate hardware resources, and adopt a hierarchical progressive + role-playing assessment activity model	Introduce real cases to meet the operation and maintenance needs of enterprises, use virtual and real scenarios + remote monitoring to implement assessments, and cultivate independent problem-solving capabilities	Build a diversified evaluation system, integrate quantitative and qualitative indicators, and use evaluation reports and continuous improvement mechanisms to ensure that the assessment conforms to industry practice

3.2. Mapping Association Modeling Method of Abstract Protocol Principles and Hardware Devices

The primary focus of the "Conceive" stage is to address the disparity between abstract communication protocol principles and their application in engineering practice. The core objective of this course is to ensure students thoroughly understand the principles of HTTP/HTTPS, MQTT, EDP, and other similar protocols. This includes the ability to effectively map and associate these principles with hardware devices. The desired learning outcome is for students to accurately select appropriate protocols and design hardware interaction logic tailored to specific application scenarios. This will help avoid errors in protocol selection and data analysis. Furthermore, in response to the practical needs of the industry, we have established evaluation dimensions that cover comprehension of protocol principles, accuracy of mapping models, and adaptability to real-world applications. These dimensions incorporate both quantitative measures (such as model accuracy) and qualitative assessments (like scenario adaptation logic), providing a comprehensive framework for subsequent instruction.

The "Design" stage focuses on planning teaching content, resources, and activities. It establishes the "underlying principles of communication protocols" based on teaching content (encompassing RESTful interface, Topic subscription, security certification) and hardware device characteristics (such as Communication Interface and Data Interaction

Mechanism). This stage maps the knowledge chain and designs specific modules for the core logic of protocol and hardware interaction, emphasizing data format matching and interface compatibility. In resource integration, it collates official documents from the cloud platform and foundational resources like open-source protocol stack code and hardware development manuals. Additionally, it creates a virtual simulation experimental platform and incorporates practical resources, including real-world project cases. Teaching activities employ project-based learning, where groups undertake scenario projects like constructing a smart home MQTT protocol-hardware interaction model. There's also case teaching that analyzes protocol-hardware mapping error failure cases and interdisciplinary approaches.

The "Implementation" stage emphasizes the practical application and transformation of teaching methodologies. Guided by learning objectives, this stage integrates industry scenarios such as smart home technology and industrial IoT. It introduces students to the latest protocol standards, such as MQTT 5.0, and hardware technical specifications, including edge device communication interfaces. Students are encouraged to engage in the "protocol adaptation-hardware docking" tasks within actual enterprise projects. The teaching approach also incorporates ideological and political elements, fostering a rigorous engineering attitude through fault case analysis. To stimulate technological innovation, students participate in innovative modeling projects. Diversified teaching strategies are employed, including the flipped classroom model-where students learn protocol principles online before class and focus on case discussions and project practice during class. The combination of virtual and real practice, using actual equipment in the IoT laboratory and virtual simulation platforms, enhances the teaching effectiveness by enabling students to complete the full process of "protocol design-hardware docking-simulation testing."

The "Operate" stage employs a closed-loop teaching methodology, optimizing evaluation and continuous improvement to construct a diversified assessment system. This system utilizes formative evaluations, such as project progress tracking and experimental report assessments, in conjunction with summative evaluations like project defenses and actual equipment testing, which are integrated into the students' experiences. It includes a multi-dimensional evaluation of self-assessment, peer mutual evaluation, and teacher assessment. By combining quantitative measures (e.g., protocol principle test scores, mapping model accuracy) with qualitative assessments (e.g., innovative thinking, problem-solving capabilities), a comprehensive "personal ability radar map" is generated to highlight students' areas for improvement. Based on this, a continuous improvement mechanism is established, reinforced by timely feedback following monthly phased reviews.

3.3. Scenario Case Development of "Protocol Selection-Data Analysis-Exception Handling" Full-Process

The "Conceive" stage addresses the prevalent industry challenges of "difficult cross-platform adaptation of cloud platforms" and the "error-prone application of communication protocols." It underscores the primary objective of the course: enabling students to autonomously manage end-to-end case development on multiple cloud platforms, including Alibaba Cloud and Tencent Cloud. The curriculum ensures that students gain proficiency in cloud platform device access, API specifications, and rights management, thereby mitigating the risks associated with configuration errors. Concurrently, we align our case development curriculum with real-world enterprise requirements, establishing assessment criteria that encompass the comprehensiveness of case functions, the precision of protocol selection, and the robustness of exception handling. This provides a clear direction and standard for evaluating case development instruction.

The "Design" stage is structured around the content, resources, and tasks involved in case development. The foundational knowledge chain encompasses "multi-cloud platform features, protocol selection logic, data analysis methods, and exception handling strategies." This chain is constructed upon the teaching content, categorized as follows: Basic cases (single platform and single protocol, e.g., Alibaba Cloud MQTT data analysis) and Comprehensive cases (cross-platform and multiple protocols, e.g., cross-cloud device data interoperability). In terms of resource integration, document resources such as official documentation and open-source code from the cloud platform are combined with practical resources like cloud experimental environments and enterprise exception cases. The approach to case development activities employs a task-driven model, segmented into three primary tasks: selection, resolution, and exception handling. Furthermore, a case sharing mechanism is implemented to facilitate students' efficient mastery of the development process and to encourage the exchange of experiences.

The "Implementation" stage emphasizes the execution of case development practices that align with enterprise requirements, integrating real-world industry scenarios such as smart agriculture and industrial monitoring into the case design. Students are required to adhere to enterprise specifications, including document templates and code review processes. The goal is to ensure that the outcomes align with the actual project, while simultaneously fostering communication skills and a sense of responsibility through cross-group collaboration. Practically, groups of 3-4 students collaborate, with roles defined for task allocation, development, and exception handling. Online Q&A support and development manuals are provided for cloud platform engineers. Furthermore, full-case development competitions are organized to encourage student enthusiasm, focusing on functionality, performance, and innovation.

The "Operate" stage constitutes a closed-loop teaching approach, characterized by continuous iteration and optimization evaluation, thereby establishing a diversified evaluation system. This system seamlessly integrates formative evaluations, such as code normative reviews and phased achievement evaluations, with conclusive evaluations like case functional testing and performance assessment. Moreover, it involves the participation of enterprise engineers in the review process. The evaluation also incorporates both quantitative indicators, such as case function realization rates, and qualitative indicators, like program innovation. This holistic integration facilitates specialized training addressing prevalent issues. Based on this foundation, a mechanism for continuous improvement has been instituted, ensuring timely optimization of the development process via weekly code problem feedback and monthly case reviews. To ensure the alignment of teaching methodologies with enterprise practices, both case content and development resources are periodically updated, reflecting the latest industry technology advancements and evaluation data insights.

3.4. Application Practical Assessment Mechanism Based on Fault Injection

Addressing the technical gap characterized by a theoretical understanding without practical troubleshooting skills in interactive hardware and cloud development, this study sets a clear core assessment goal. This goal is to evaluate students' abilities in hardware debugging, protocol communication, and cloud data processing, requiring them to quickly identify and rectify issues in fault injection scenarios such as module failures or data abnormalities. The assessment not only meets the operational and maintenance needs of enterprises by establishing technical dimensions like fault location accuracy, repair time, and system recovery integrity, but also incorporates professional qualities including team collaboration efficiency, communication clarity, and responsibility. This multidimensional approach provides clear directions and scientific standards for effective combat assessment.

The assessment content is structured in layers, with each layer corresponding to the complexity of the fault. Basic faults encompass errors such as WiFi/4G module debugging

and serial communication configuration exceptions. Comprehensive faults involve more intricate issues, including superposition of TCP transparent data exceptions and protocol parsing errors. Frontier faults are even more advanced, incorporating 5G module delays and edge node synchronization exceptions. These scenarios are designed to emulate the real-world operations and maintenance activities of an enterprise. To facilitate this, we offer ESP32, STM32 development boards, and a variety of communication modules. Together, these form a fault injection testing platform capable of network disconnection, data tampering, cloud account integration, enterprise case libraries, and standard document templates. Our assessment activities follow a hierarchical and progressive model that transitions from individual screening to group repair. This approach integrates role-playing and report submission to enhance understanding post-assessment.

The practical assessment implementation aligns closely with the operational and maintenance requirements of enterprises. It incorporates real-world scenarios, such as factory gateway protocol abnormalities, and mandates that students adhere strictly to enterprise operational procedures and hardware and protocol investigation guidelines. Concurrently, unpredictable faults are introduced to evaluate Incident Response Service and decision-making skills. The pedagogical approach leverages both virtual and real elements: the fault injection platform simulates virtual faults and is equipped to thoroughly verify actual hardware faults. Real-time remote monitoring of the operational process provides analytical and troubleshooting insights. These only offer directional hints, rather than direct solutions, aiming to foster students' independent problem-solving abilities.

The assessment and evaluation processes together establish a diversified system. Formative evaluation concentrates on operational standardization, analytical thought processes, and collaborative performance. On the other hand, conclusive evaluation merges results of repairs, the duration of assessment, and quality scores of documents. It also introduces a mutual evaluation model involving school teachers, business engineers, and students. Concurrently, quantitative indicators such as fault location time and repair success rate are coupled with qualitative indicators like cognitive logic and innovation ability. These combined aspects generate an operation and maintenance capability evaluation report which serves as a guide for improvement. Based on these evaluations, a continuous improvement mechanism is established. This involves conducting training sessions on prevalent issues, updating fault types in accordance with industry technology, adjusting the proportion of assessments, and regularly upgrading platforms and tools to ensure that the assessments remain aligned with industry practices.

4. Conclusions

This study, guided by the CDIO model, addresses the prevalent issue of a 'disconnect between theory and practice' in the course of IoT cloud platform application development. It pinpoints three major areas of difficulty: cross-platform adaptation in cloud platforms, the disconnection between communication protocol principles and their real-world application, and the lack of development technology in hardware-cloud interaction. To address these issues, a comprehensive teaching reform system has been constructed. The reform system incorporates three core projects spread across four stages. The first project, "Abstract Protocol and Hardware Mapping Related Modeling," bridges the gap between protocols and hardware. This is achieved through a clarification of industry goals and evaluations, the planning of knowledge chains, the implementation of scenario-based teaching, and the establishment of diverse, closed-loop evaluations. The second project, "Protocol Selection-Data Parsing-Exception Handling Full Process Case Development," integrates resources from multiple cloud platforms driven by specific tasks. It also introduces corporate standards and competitions, and aligns with the evolving needs of the industry. Finally, the third project, "Fault Injection Practical Assessment Mechanism," designs assessment content in layers. It combines both virtual and real assessments while

merging technical and literacy evaluations. By integrating the CDIO concept of project-driven learning into the teaching approach, this reform guides students to master skills across the entire process. It enhances their ability to solve complex engineering problems and shifts the emphasis from mere 'knowledge transfer' to 'ability cultivation'. This aligns with the requirements for the training of composite engineering talents.

Funding: This work is supported by the Quality Engineering Project of Guangdong University of Science and Technology under Grant GKZLGC2025028.

References

1. K. Fufon, S. Puengsungwan, and K. Chomsuwan, "Integrating CDIO and Problem Based Learning Frameworks for Industrial Internet of things Training Course Development," In *2024 9th International STEM Education Conference (iSTEM-Ed)*, July, 2024, pp. 1-6. doi: 10.1109/istem-ed62750.2024.10663126
2. G. Ortegat, D. Grolaux, E. Riviere, and J. Vanderdonckt, "Engineering the transition of interactive collaborative software from cloud computing to edge computing," *Proceedings of the ACM on Human-Computer Interaction*, vol. 6, no. EICS, pp. 1-31, 2022. doi: 10.1145/3532210
3. E. F. Crawley, D. R. Brodeur, and D. H. Soderholm, "The education of future aeronautical engineers: Conceiving, designing, implementing and operating," *Journal of Science Education and Technology*, vol. 17, no. 2, pp. 138-151, 2008. doi: 10.1007/s10956-008-9088-4
4. B. Tanveer, and M. Usman, "An Empirical Study on the Use of CDIO in Software Engineering Education," *IEEE Transactions on Education*, vol. 65, no. 4, pp. 684-694, 2022. doi: 10.1109/te.2022.3163911
5. X. Yuan, J. Wan, D. An, J. Lu, and P. Yuan, "Multi-method integrated experimental teaching reform of a programming course based on the OBE-CDIO model under the background of engineering education," *Scientific Reports*, vol. 14, no. 1, p. 16623, 2024. doi: 10.1038/s41598-024-67667-6
6. S. J. Fusic, N. Anandh, A. N. Subbiah, and D. B. K. Jain, "Implementation of the CDIO framework in engineering courses to improve student-centered learning," *Journal of Engineering Education Transformations*, pp. 19-26, 2022.