Research on Teaching Object-Oriented Technology Courses

Xiaoqi Shi[1, a]

[1] Xi'an Technological University, Xi'an, Shaanxi, China

[a] shixiaoqi713@163.com

Abstract. As computer science and software engineering advance rapidly, particularly in the realm of intricate systems and extensive computer program schemes, traditional process-aimed programming approaches have increasingly revealed challenges, including maintenance difficulties and limited scalability. To address these issues, object-aimed technology emerged. This paper commences with an introduction to object-oriented technology, followed by an overview of the relevant course. The core focus of this paper lies in exploring enhancements to the teaching of the object-oriented technology course. By analyzing the course characteristics and the current state of teaching, it is identified that issues such as monolithic teaching methods, inadequate student engagement, and suboptimal resource allocation hinder the cultivation of students' practical programming skills and overall learning experience.In this regard, a series of innovative initiatives are proposed, including the reform of teaching methods based on project-driven and blended teaching, and finally, through the evaluation and analysis of teaching effects, it can be verified that the students' abilities in programming practice, problem solving, teamwork, etc. are substantially enhanced and the power of the instructional innovation. Upon conducting teaching research for the course, educators can, based on the ultimate analysis outcomes, devise a more refined and effective practical teaching framework, thereby establishing a solid groundwork for students' comprehensive development.

Keywords: Object-oriented Technologies; Course Teaching; Teaching Methods; Practical Teaching

1. Introduction

Before the object-oriented programming model became common, the main move towards to computer program development was 'purpose -aimed'. The core of this approach lies in the need to first have a comprehensive grasp and understanding of the overall function of the target system, and then, through a series of orderly steps, gradually refine and decompose these overall functions into a number of more specific and easier to manage functional modules. Although this function-based development strategy has a certain degree of rationality and feasibility in the historical context of the time, its inherent limitations are also obvious.

Specifically, when a function-based development approach is used to write software, developers often need to modify and adjust a large amount of code once the specifications of the system change or when certain new functional features need to be added. Such modifications not only involve a wide range, but also often require a relatively large adjustment to the original system architecture, which undoubtedly increases the complexity of software maintenance and upgrading [1]. More seriously, it is difficult to achieve effective reuse of software code because the function-based development approach lacks sufficient flexibility and scalability. The result of this is not only a decline in the efficiency of computer program development, but also an elevation in the expenses and hazards associated with software creation.

The aim of object-oriented technology lies in facilitating easier maintenance and reuse of software. Its core concept revolves around emphasizing individual components, enhancing their autonomy, and integrating these components to fulfill the overall functionality of the system. By increasing the independence of the components, when modifications occur, they can be reused in other systems with minimal impact [2].

Hence, prior to engaging in computer programming, students must possess a profound understanding of object-oriented technology principles. When teaching the course titled "Object-Oriented Technology (Java)," the primary objective is to nurture students with robust

foundations in object-oriented programming concepts, logical reasoning, and problem-solving abilities [3]. The ultimate goal of this curriculum is to empower students to develop exemplary programming practices, master fundamental programming techniques, and proficiently apply the programming language to address real-world challenges effectively. By combining the fundamental theories of object-oriented programming with its practical techniques, the course aims to develop students' ability to adapt to technological developments and market demands.

Object-oriented originally appeared as a programming language, and in the following 40 years, through continuous development, it has gradually been applied to various fields of development. The full picture and development process of object orientation is shown in Fig. 1.

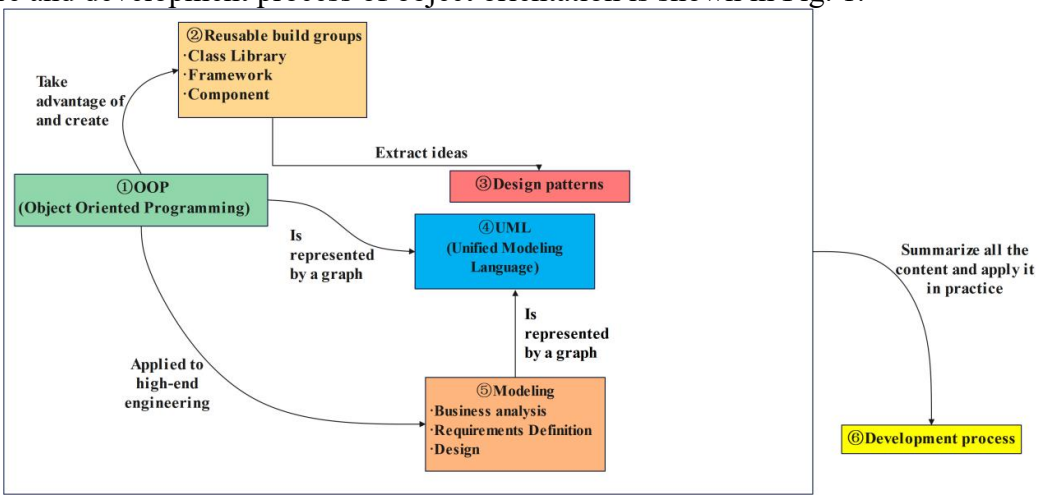

Figure 1. The whole picture and development process of object orientation

2. Course Overview of Object-Oriented Technologies

The Object-Oriented Technology course serves as a fundamental offering within computer science and technology, software engineering, and affiliated disciplines. Its objective is to equip students with proficiency in object-oriented programming concepts, methodologies, and skills, enabling them to adeptly apply object-oriented programming languages in software development endeavors.

First of all, students are allowed to study and understand the basic concepts of object-oriented in depth, including encapsulation, inheritance, polymorphism, object, class, abstract class, interface, etc. In particular, the first three are key components of Object-Oriented Programming (OOP). Their distinctive attributes are outlined in Table 1. By grasping the meanings, characteristics, and interconnections of these concepts, one can precisely elucidate the distinctions and benefits of OOP over traditional OOP. This understanding is crucial for fostering an OOP mindset [4].

Table 1 The three main elements of OPP

Three Major Elements	Encapsulation	Inheritance	Polymorphism
Explanation	Summarizes subroutines and variables, creating software components.	Achieves the commonalization of repeated class definitions.	Achieves the commonalization at the method invocation end.
Purpose	Organization	Elimination of redundancy	Elimination of redundancy
Notation	Structure that "summarizes, hides, and creates many"	Structure that creates a common main program	Structure that summarizes the common parts of classes into another class.

Secondly, once the foundational concepts of object-oriented programming are grasped, students must proceed to master the fundamental syntax and programming structure of a prominent object-oriented programming language (such as Java, C++, or Python, among others). This entails understanding data types, variables, operators, control flow statements, arrays, strings, and other essential elements. Additionally, they must become proficient in object-oriented specific features, including classes, objects, and member functions, constructor, destructor and other specific syntax and methods of using the language to write object-oriented programs of a certain scale and complexity [5], and solve practical problems. Mastering the specific syntax and application of object-oriented features is crucial, enabling students to craft object-oriented programs of varying scales and complexities to address real-world problems [5]. Ultimately, they should acquire familiarity with the fundamental techniques and procedures involved in Object-Oriented Analysis and Design (OOAD). Additionally, proficiency in the Unified Modelling Language (UML) is essential, encompassing knowledge of the graphical notations used in its commonly employed diagrams. (e.g., class diagrams, use case diagrams, timing diagrams, etc.), as well as their uses, as shown in Table 2 [6]. Students should learn to use OOAD methodology to conduct requirements analysis and system design for real-world problems, establish reasonable class hierarchies and object relationship models, furnish precise and unambiguous design schemas that serve as the foundation for subsequent coding implementations, thereby enhancing the effectiveness and quality of computer program arrangement development.

Table 2 Names of the 13 graphs defined in UML and their uses

No.	Name	Purpose
1	Class Diagram	Represents the specification of a class and the relationship between classes
2	Composite Structure Diagram	Represents the runtime structure of a class with a whole-part structure
3	Component Diagram	Represents the implementation structure of software such as files and databases, processes and threads
4	Deployment Diagram	Represents the physical structure of a system, such as a hardware, network, etc
5	Object Diagram	Represents a relationship between instances
6	Package Diagram	Represents a relationship between packages
7	Activity Diagram	Represents a control flow in a series of processes
8	Sequence Diagram	The interactions between instances are represented as time series
9	Communication Diagram	Represents the interactions between instances as an organizational structure
10	Interaction Overview Diagram	A sequence diagram that performs different actions according to different conditions is represented in an activity diagram
11	Timing Diagram	A time axis with a digital scale is used to represent state transitions and interactions between instances
12	Use Case Diagram	Represents the relationship between the functions provided by the system and the consumers
13	State Machine Diagram	Represents a state change for an instance

3. Analysis of the Current State of Teaching and Learning

The Object-Oriented Technology course plays a key role in developing students' programming skills and systems development thinking, however, the teaching process of the course currently displays a diverse array of progress, marked by both successes and hurdles. The problems in teaching the course are shown in Fig. 2.

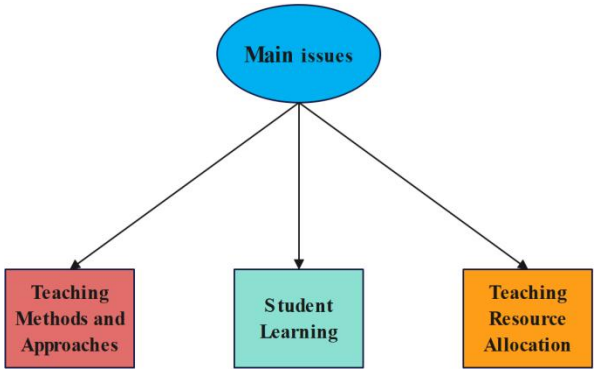

Figure 2. Curriculum Teaching Issues

3.1 Teaching Methods and Tools. Currently, the most frequently utilized teaching approach in object-oriented technology routes remains the lecture manner. Teachers systematically explain the knowledge of object-oriented concepts, principles, syntax rules and programming examples through PPT presentations and board books in the classroom. The primary benefit of this manner lies in its ability to convey extensive information within a confined timeframe, ensuring a comprehensive and systematic acquisition of knowledge. Nevertheless, its drawbacks are equally prominent. Students often find themselves in a passive position, merely receiving knowledge without active engagement or contemplation. This can result in a superficial understanding of concepts and hinder the development of practical application skills and innovative thinking among students [7].

Although practical teaching has an important position in the course, in actual teaching, the connection between practical teaching links and theoretical teaching is not close enough. The design of practical projects sometimes lacks systematicity and comprehensiveness, and fails to fully consider the actual level and ability enhancement needs of students. For example, some of the practical projects are only a simple verification of theoretical knowledge. Upon completing these projects, students gain a certain intuitive grasp of the scholarly knowledge, however, they still fall short in their capacity to independently analyze and tackle intricate practical problems. Furthermore, there is room for enhancement in the methodology employed for practical teaching instruction. Teachers may not be able to give adequate guidance and feedback to each student in time during the practical process, which affects the practical effect of students.

Case teaching method has an important role in object-oriented technology courses, but there are some problems in the selection and use of cases. Certain cases are overly outdated and disconnected from present-day industrial applications, failing to ignite students' interest and keenness in knowledge. Additionally, during case-based instruction, the depth of the teacher's analysis and guidance is inadequate. Consequently, students tend to merely follow the teacher's train of thought, lacking the opportunity for independent thinking and discussions. This hinders their ability to genuinely acquire knowledge and experience from the cases, and to enhance their problem-solving skills and object-oriented thinking abilities.

3.2 Student Learning Situation. The abstraction and complexity inherent in Object-Oriented Technology courses often lead students to encounter challenges during their learning journey, which can diminish their enthusiasm and proactive learning attitude. When it comes to learning strategies, numerous students continue to rely on traditional manners such as rote memorization and repetitive exercises when studying object-oriented technology courses, lacking systematic summarization and generalization of knowledge, and failing to form their own knowledge system and way of thinking. They tend to rely too much on textbooks and teachers' explanations in the learning process, lack the spirit of independent learning and exploration, and are not good at using network resources, reference books and other channels to acquire knowledge and solve problems.

In addition, students' programming practice ability generally needs to be improved. In the classroom experiments and post-course assignments, many students are able to complete basic programming tasks, but the code quality is not high, with problems such as code redundancy,

confusing structure and lack of comments. This shows that they still need more guidance and training in programming specifications and the development of good programming habits.

When solving complex programming problems, students' ability to analyze problems and debug programs is weak. When encountering the situation that the programmer reports an error or the running result does not meet the expectation, some students are often at a loss as to what to do, and they lack effective debugging methods and skills to locate and solve the problems quickly, which to a certain extent affects their confidence in learning and the practical effect.

3.3 Allocation of Teaching Resources. Rapid advancements are being witnessed in computer technology, accompanied by the continual evolution and expansion of object-oriented technology. New programming paradigms, frameworks, and utilities are continually emerging.However, the content of many existing textbooks is updated slowly and fails to reflect the latest developments and technology trends in the industry in a timely manner. For example, there is insufficient introduction to the characteristics of emerging object-oriented programming languages, the improvement of design patterns, and the application cases in new fields such as artificial intelligence and big data processing, which creates a certain disconnect between what students have learnt and the actual industry needs, and is not conducive to students' rapid adaptation to the technical requirements of the workplace after graduation.

Regarding hardware facilities, numerous colleges and universities have been furnished with dedicated computer laboratories tailored for the instruction of this course, but there are cases of aging equipment and uneven performance. The hardware configuration of some computers is difficult to run complex object-oriented development environments and large-scale projects smoothly, resulting in students may encounter problems such as lagging and crashing during practice, which affects the learning efficiency and practical experience. Moreover, the opening hours of the labs are limited, which often fails to meet students' needs for independent learning and practice after class, restricting students’ full use of hardware resources.

The campus network can basically meet the demand in terms of coverage, but there are deficiencies in network speed and stability. Especially during instances where students focus on utilizing the network for online learning, software updates, or collaborative team projects, network congestion becomes more prevalent. This often leads to issues such as sluggish loading of online course videos, delayed submission of code, and other related problems, thereby disrupting the smooth flow of instruct and knowledge actions. Furthermore, it impacts the prompt and seamless access of students to online educational resources.

In terms of instructor resources, the teaching staff for object-oriented technology routes collectively possess a certain degree of specialized knowledge and expertise, but there are problems of aging knowledge structure and insufficient practical experience of teachers. Some teachers have been engaged in theoretical teaching for a long time and lack of practical project development experience, which makes it difficult to combine theoretical knowledge with practical application in the teaching process, and incapable to furnish students with lifelike and authentic case studies, as well as practical directions and advice. Moreover, with the continuous updating of technology, teachers' own knowledge updating speed cannot keep pace with the development of the industry, resulting in a disconnect between the teaching content and the actual needs [8].

4. Teaching Methodology Innovation and Practice

Old-fashioned teaching approaches typically emphasize the imparting of abstract knowledge, often leading students to find the knowledge process monotonous. Applying acquired knowledge flexibly in real-world project development remains challenging. To alter this scenario, enhance the course's teaching quality, and nurture high-caliber talents capable of meeting the demands of contemporary software development is imperative, this paper will carry out an in-depth exploration of the teaching methods of the object-oriented technology course and innovative practice, which is mainly divided into the following aspects. As shown in Fig. 3.

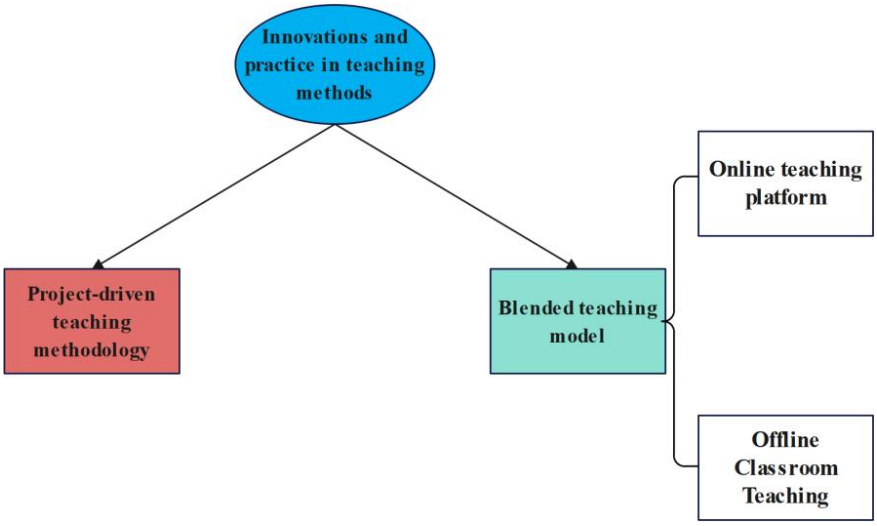

Figure 3. Innovation and practice of teaching methods

4.1 Project-Driven Teaching Methodology Based. First of all, the teacher should select a project with practical application background and covering the main knowledge points of object-oriented technology as the main teaching line of the course [9], for example, to develop a small 'library management system', 'student information management system' or 'online shoe shopping system', and so on. For example, the development of a small 'library management system', 'student information management system' or 'online shoe shopping system' and so on. Prior to embarking on the development and plan of these projects, students are required to acquire and grasp the fundamental procedures of object-oriented analysis and plan, as shown in Fig. 4.

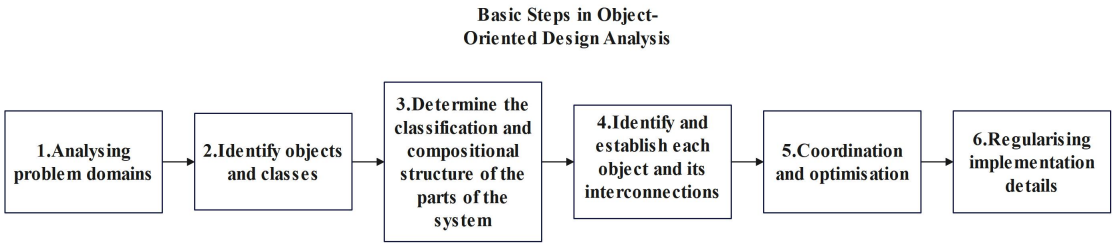

Figure 4. Innovation and practice in teaching methods

(1) Examine the problem domain thoroughly. Ascertain user needs by comprehending the business knowledge pertinent to the issue. Clearly delineate the system's user requirements, establish its responsibilities and boundaries, and investigate preliminary solutions to the problem.

(2) Recognize and define objects and classes within the system. This involves analyzing and pinpointing the entities that comprise the system, followed by abstractly categorizing these objects. Envision the system as consisting of various subsystems, and subdivide each subsystem into themes, which are constructed from collections of classes and objects.

(3) Determine the classification and composition of the various parts of the system. Firstly, ascertain the inheritance hierarchy among classes and establish the classification structure based on the general-to-specific relationship. Secondly, determine the constituents of an object according to the whole-part relationship and delineate the composition structure accordingly.

(4) Identify and establish each object and its relationship to each other. That is to identify the object based on the application, define the internal characteristics of the object (properties and methods) , establish instance connection and message connection. Message Connection embodies the communication relationship and interface protocol between objects.

(5) Harmonization and Enhancement. Further refine and optimize the performance and interactions among the diverse components of the model (encompassing class instances), honing

potential classes or objects in the process, so that the system is a minimum set of different components.

(6) Codify implementation details. Analyze and design the functional implementation details for each component of the model, including class objects, and check the consistency and integrity of the analysis model.

These projects are not only capable of sparking students' enthusiasm for learning but also fostering a profound understanding of the practical significance of object-oriented technology in real-world software development. Furthermore, instructors can decompose the project into multiple sub-tasks, progressively guiding students through the completion process in accordance with the actual progression of the course. At each stage, the instructor first elucidates the pertinent theoretical knowledge and technical aspects, subsequently allowing students to engage in practice, either in groups or individually, flexible use of knowledge to complete the arrangement of sub-tasks. For example, in the“Online Shoe City Shopping System” project, first introduced the concept of class and object, let students create“Commodity class” and“User class”; then explain inheritance and polymorphism, guide students to design different types of goods (such as sports shoes, high heels, casual shoes, etc.) and the inheritance of the corresponding order, returns and other operations of the polymorphic realization; Bring in the knowledge of database association and interface plan to faultless the purpose of the whole system.

4.2 Blended Learning Model. The blended teaching model, as its name implies, integrates online and offline instruction seamlessly to construct an educational framework tailored for object-oriented technology routes, as shown in Fig. 5.

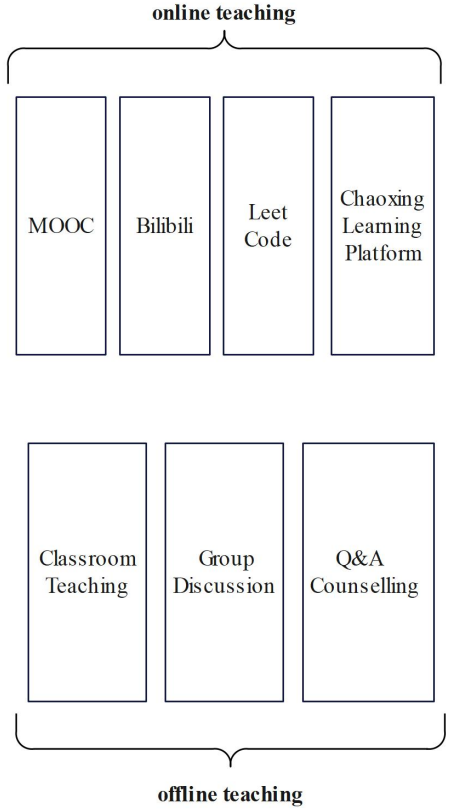

Figure 5. Blended learning model

Online teaching includes the selection of teaching platforms (e.g., course websites, online learning communities, MOOC platforms, etc.), the development and integration of teaching resources (e.g., teaching videos, e-textbooks, online test questions, discussion topics, etc.), and teachers can recommend students to use some high-quality online programming learning platforms (e.g., LeetCode, NiuKe.com, etc.), which provide a large number of object-oriented programming These platforms provide a large number of object-oriented programming practice questions and real-life project cases, and students can engage in online programming custom on the stages, thereby

enhancing their programming skills progressively through hands-on experience. The platforms also provide real-time evaluation and feedback on students' codes, pointing out errors and deficiencies in the codes and providing corresponding optimization suggestions to help students improve their programming level quickly [10].

Offline teaching, on the other hand, includes the organization and design of classroom teaching activities (e.g., classroom lectures, group discussions, project practice, etc.) Teachers can provide students with an open laboratory environment after class, equipped with sufficient computer equipment and related software resources, and arrange instructors on duty to answer the problems encountered by students in the course of practice. Based on their individual learning pace and interests, students can autonomously allocate time for programming exercises and project development in the lab, thereby fully harnessing their learning initiative and enthusiasm. Ultimately, educators can assess the students' learning outcomes and experiences before and after the implementation of the blended teaching model, confirming its advantages and viability. Subsequently, they can discuss the challenges encountered during implementation and promptly devise appropriate solutions.

5. Evaluation and Analysis of Teaching Effectiveness

To gain a comprehensive and impartial understanding of the course's teaching effectiveness and identify any issues or shortcomings in the instructional process, it is imperative to conduct a thorough evaluation of its teaching impact. This will facilitate further refinement of teaching methodologies and enhancement of educational quality. The teaching evaluation and analysis is shown in Fig. 6.

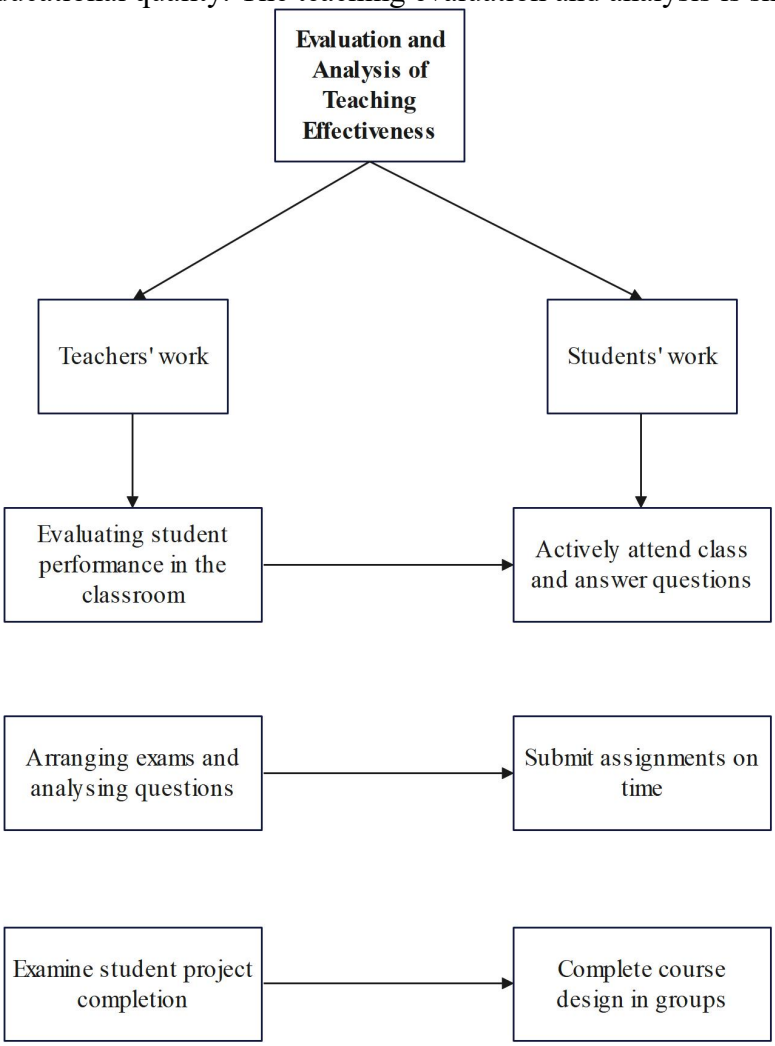

Figure 6. Teaching evaluation and analysis

Initially, educators can monitor students' in-class performance, encompassing attendance rates, the frequency of questions posed, eagerness to respond, and proactive engagement in class discussions and group activities. Students who actively participate in classroom activities often exhibit greater interest in the course and enthusiasm for learning, which also serves as an indicator of the appeal of the teacher's instructional methods and the vibrancy of the classroom ambiance. Prior to the conclusion of class, assignments closely aligned with theoretical knowledge, such as programming tasks and essays, are assigned. These assignments require students to apply their acquired knowledge to address practical challenges or respond to theoretical inquiries. By assessing these assignments, educators can gauge students' comprehension of the key concepts, their depth of understanding, and their ability to apply knowledge flexibly to solve problems [11].

Secondly, students are tasked with collaborating in small groups to complete a relatively intricate course design project, such as a small-scale information management system or game development. This project requires them to comprehensively analyze various aspects, including demand assessment, system plan, principles script, testing and debugging, as well as teamwork skills. Throughout the project completion process, educators can observe students' performance, focusing on the progress of the experimental project, the completeness and accuracy of functional implementation, and the quality of the code written (e.g., normality, readability, extensibility, etc.). In addition, the experimental projects should cover all aspects of object-oriented technology, such as class design and implementation, application of inheritance and polymorphism, exception handling, graphical user interface development, etc., To assess students' practical programming skills and their ability to comprehensively apply knowledge, teachers arrange a timely course design presentation once students have submitted their projects. An evaluation panel comprising both teachers and students presents and evaluates each group's project outcomes. Evaluation criteria encompass innovation, practicality, technical complexity, functional completeness, and teamwork skills. Additionally, a semester-end closed-book exam is conducted to evaluate students' understanding and retention of fundamental object-oriented concepts (like objects, classes, encapsulation, inheritance, polymorphism, etc.), syntax rules, programming paradigms, and pertinent algorithms and data structures. The exam questions are diversified, including multiple-choice, fill-in-the-blank, short-answer, and programming questions, to comprehensively gauge students' theoretical knowledge. Upon reviewing the exam results, teachers analyze the distribution of student scores, such as the average score, pass rate, and percentage of excellent scores, to ascertain students' overall grasp of theoretical knowledge. Furthermore, they compare these data across different classes and semesters to observe the stability of the teaching effect and identify trends in performance.

By conducting an exhaustive examination of the particular effects that diverse practical teaching components and their methodologies exert on various facets of students' practical competencies, teachers aim to comprehensively and accurately identify the significant advantages and potential shortcomings of the current teaching system, so as to provide a strong and scientific basis for the subsequent optimization and upgrading of the teaching system. In this process, data analysis tools should be fully utilized to quantitatively assess the effectiveness of different practical projects and teaching methods. Certain well-designed practical projects have shown remarkable results in enhancing students' specific abilities, which are often closely in line with the needs of the industry and focus on the in-depth integration of theory and practice, consequently, there is a significant advancement in students' practical skills. Nevertheless, it is crucial to acknowledge that certain teaching methods have encountered challenges in practical application, including insufficient interactivity and difficulty in sparking students' learning enthusiasm, which, to a certain extent, hinders the full realization of the teaching effectiveness.

By conducting such an analysis, educators can gain a clear insight into both the strengths and advantages of the existing teaching system, as well as precisely pinpoint the aspects that require enhancement and refinement, so as to provide a clear direction and goal for the subsequent teaching reform. This will help teachers to build a better and more efficient teaching system to better cultivate students' practical abilities so that they can be more confident in their future job search.

6. Summary

The problem-solving plan of object-oriented technology is to start from the factual objects in the true earth (such as persons and object), and try to use human's natural way of thinking to construct the software system, and it can be seen from the epistemological point of view that the object-oriented technology has changed the way of people's understanding of the world.

During the process of acquiring object-oriented technology, instructors ought to adapt the prevailing traditional teaching approaches based on current circumstances, embracing suitable methodologies such as project-based learning and the hybrid teaching model discussed in this paper. These innovative teaching methods should provide students with a revitalized learning environment and atmosphere. Additionally, educators should prioritize fostering students' interest in the course. Ultimately, a series of evaluative measures can be implemented to further refine and enhance the teaching system.

Under the meticulous guidance of instructors, students engage in a blend of online and offline learning to attain a profound understanding and mastery of object-oriented programming concepts, including classes, objects, inheritance, polymorphism, and encapsulation, among other fundamental principles. This mastery necessitates the integration of theoretical knowledge with practical application through a multitude of hands-on projects. By engaging in these project-based practices, students accumulate substantial programming experience, thereby enhancing their code quality and programming efficiency. Moreover, in the current environment, innovation and collaboration are particularly important. Object-oriented technology courses can cultivate students' sense of innovation, so that they can constantly explore new programming ideas and methods. And group projects can exercise students' teamwork ability, so that students understand how to play their respective advantages in the team and complete the task together.

As educational technology continually advances and the domain of object-oriented technology undergoes constant updates, the teaching of this course in the future will make more use of artificial intelligence and big data analysis technology to provide a more effective way to continue to optimize the teaching evaluation system to better meet the needs of knowledge, and to propel forward the research endeavors related to the instruction of object-oriented technology routes.

References

[1] M.J Kang, Z.L Cai and M.J Wang. Design of Object-Oriented GIS Software Development Course Model[J]. Surveying and Mapping Bulletin,2023, (S2):64-68.

[2] J. Cheng, L.J Liu and X. Chen, et al. Object-oriented equipment modelling method and its application[J]. Industrial and Mining Automation,2021,47(04):113-115+120

[3] H.M Gao, Y.X Huang and D.G Gao, etc. A study on the three-dimensional talent training model of object-oriented technology course in universities in Xizang -- taking Xizang University as an example [J]. Scientific research on the plateau,2023,7(02):105-113.

[4] H.H Li and Y.X Xiao. Implementation and numerical simulation of object-oriented elastoplastic finite element method based on Python [J]. Journal of system simulation,2024,36(05):1107 -1117.

[5] Z. Huang and T. Liu. The modularization development and practice of object-oriented program technology course under the background of “Competition” [J]. Education Information Forum,2023, (07):102-104.

[6] Z.C Yuan and Z.M Ma. Integrated classification of UML class diagrams based on semantics [J]. Computer Engineering and applications,2021,57(12):257-262.

[7] D. Dong. Management by objectives Java object-oriented programming teaching [J] . Computer Education, 2020, (08) : 9-13.

[8] S.J Wu. Research on Splitting Method and Maintainability Metrics for Microservice Architecture[D]. Anhui University,2023

[9] J. Li, Q. Xu and J.X Hu, et al. Research and practice on refining the elements of Java object-oriented programming technology “curriculum politics”[J]. China New Communication,2022,24(02):226-228.

[10] X.C Tang and X. Zhang. Exploration of full-stack knowledge point teaching method of object-oriented technology[J]. Computer Education,2021,(10):147-151.

[11] L. Wang. Research and practice of teaching mode reform for value-added evaluation in higher vocational colleges and universities--Taking ‘Java object-oriented programming’ course of software technology as an example[J]. Journal of Hunan Post and Telecommunication Vocational and Technical College,2022,21(01):45-49.