Mixed teaching of design patterns based on MOOC

Wenping Chen, Fanqing Meng* and Zhenfeng Jiang
Zaozhuang University, No.1 Bei'an Road, Shizhong District, Zaozhuang City, Shandong Province, China

*Corresponding author. Email: amo0114@163.com

Abstract. In order to improve teaching effect of the "Java Design Patterns" course, the teaching process uses cases as the main line through the pattern concept, pattern structure diagram, pattern realization, application scenarios and the advantages and disadvantages of the pattern. This teaching method combines the orientation of application-oriented undergraduate, the characteristics of students' knowledge structure and course's abstract nature. In addition, this method combines traditional classrooms and MOOC platforms, and has achieved good results in actual teaching.

Keywords: design patterns, case, MOOC platforms.

1. Introduction
Many computer majors in application-oriented universities have set up "Java Design Patterns" course. This course is an important bridge between development practice and design thinking. It is not only program optimization in the general sense, but also more helpful in cultivating students' ability to use design patterns to solve software development problems[1, 2]. With the development and wide application of object-oriented technology, design patterns have gradually become one of the basic skills to be mastered in all stages of software development [3]. GOF's definition of design patterns are a set of customized solutions to solve a general design problem in a particular context. With the help of mature design patterns, we can stand on the shoulders of giants to develop software system which have better flexibility, scalability and higher reusability [4, 5].

Through the questionnaire before the start of the class, teacher understand that most of the students have the development foundation in Java and Java Web technologies and can develop small projects. However, Due to the lack of design pattern’s thinking, the websites designed by students generally violate the object-oriented single responsibility principle, open and close principle, and composite reuse principle, etc. In order to make the software more maintainable and reusable, the teaching process combines the MOOC platform with traditional classroom teaching, allowing students to have a deeper understanding of design patterns, and improve students’ initiative and knowledge, which is crucial for subsequent large-scale project development and the work of architecture design[6-13].
2. The current situation of design patterns

2.1. Single traditional teaching mode
Design pattern is not a mature subject, because the textbooks and teaching cases are not classic enough. Most of textbooks systematically introduce the basic principles of each design pattern, but their application in the project code and the bottom layer is not detailed or even mentioned. In addition, the teachers basically use the traditional teaching mode to explain the theoretical knowledge of design patterns through the key knowledge listed on the blackboard, the PPT assisted explanation, and display the effect of code. This way leads to students in a passive state of receiving knowledge and digesting knowledge, which can not exercise students' ability to think independently, and is not conducive to integrate design patterns into development thinking.

2.2. Insufficient motivation to learn
The content of design patterns is highly abstract, and its basic ideas and basic realization are the extraction and sublimation of the common features of the software system. The development framework involved in theoretical knowledge is relatively complex, so it is too difficult for students who have not participated in large-scale project development to understand. In addition, teachers pay more attention to theoretical explanations in the learning process, and use a single teaching format. However, the injection of theoretical knowledge often leads to broad interest of students, and it is difficult for students to have a deeper understanding of design pattern thinking, which makes the quality of teaching not high.

3. The design of course teaching content
In view of the contradiction between the abstract nature of the course and the lack of students' development experience, the course mainly uses case-through methods to teach the concept, structure, realization, application scenarios and relative merits of the design pattern. Integrating MOOC platform into the traditional design patterns classroom, and make full use of the resources on the MOOC platform through reasonable teaching design to improve students' independent learning ability. In addition, the knowledge points of each chapter of the Java design pattern course have a clear structure, strong logic and continuity. Therefore, "Java Design Patterns" is more suitable for MOOC teaching.

The course adopts the super-star learning SPOC platform built by our school, and various discussion questions and teaching resources are published on the platform to consolidate and strengthen the theoretical part; The practice class of design patterns adopts case programming to consolidate the knowledge points in the class. Applying MOOC to the teaching of design mode classrooms solves the problem that students have no guidance before class and cannot be solved after class.

3.1. Case-based teaching design
Case-based teaching is a new open and interactive teaching method that combines some common scenes in real life with teaching content, so that students can more easily appreciate the meaning and value of the knowledge points they have learned. The course group designs some easy-to-understand cases based on teaching goals and teaching tasks. Students can enhance their ability to analyze and solve problems through the combination of these cases and knowledge points.

Design pattern course includes 7 object-oriented design principles and 23 design patterns, and the corresponding case of each pattern design connects fragmentary knowledge points. The selection of a suitable case first requires that the main knowledge points of the model can be included, so that students can understand the theoretical knowledge more vividly through the case; and the degree of difficulty of cases should be in line with the knowledge level of the junior students.

Taking the "responsibility chain" as an example, teacher uses the "purchase order approval" case to teach the thought, structure, implementation and application scenarios of Chain of Responsibility Pattern, different levels of personnel can approve purchase orders with different prices, and determine the roles involved in the function module according to the demand analysis. According to the demand analysis, developers determine the roles involved in the approval module: buyer, approvers (group leader,
manager, chairman), among which buyers act as purchase request class ShopRequest in Chain of Responsibility Pattern, and the approving personnel (group leader, manager, Chairman) act as a specific processor, this pattern also designed an abstract approval processor "Approver" role to meet abstract programming. Approver class uses the processRequest method to process the purchase request, and after the approval is passed, the setSuccessor method is used to set and forward it to the next approver, we can see that Approver class is the core of the responsibility chain through the division of roles. The three roles designed according to requirements conform to the idea of Chain of Responsibility Pattern. Drawing UML diagrams based on logical relationships between roles, and then let students write implementation codes according to UML class diagrams. UML diagram of chain of responsibility pattern is shown in Figure 1.

![Figure 1. UML of chain of responsibility pattern.](image)

After introducing the basic case, let students brainstorm and find small examples of similar to basic case in real life to share. Based on the idea of the model, let students refine several roles in the case they designed, which increases the participation of students. Students can extract the roles in the corresponding model by analyzing the case, establish functional modules and establish logical relationships. Students can have a deeper understanding of the knowledge learned in the classroom through this practice method, and will also fully appreciate the application value of learning design patterns.

![Figure 2. Classroom organization.](image)
The selection of cases in teaching design is divided into three categories: basic cases, which help students understand the basic ideas and structure of patterns. For example, basic case used in State Pattern: water can freeze into ice or be heated into steam, water can flow, ice can be carved, and steam can diffuse. The three states of H20 and their mutual conversion relationships are also very common in software systems. There are many objects with multiple states, which can be converted to each other under certain circumstances, and have different behaviors in different states. Synchronous training cases, consolidating the mastery of knowledge through the development of small projects. Such as the synchronous training case of state pattern is to design a small coffee vending machine, which has three states, with coffee no coin, with coffee coin and without coffee, and three states have different behaviors and can be transformed with each other. Coffee sales system designed by the students in state pattern has better scalability and maintainability. Expand the case, let students use pattern thinking to analyze OA system shown by teacher, which part of the design adopts which design pattern, so as to feel the importance of design patterns in software development in specific large-scale projects. The teaching process of three types of cases is shown in Figure 2.

3.2. Classroom management
At the beginning of the class, teachers can publish the appropriate number of questions to test and consolidate the content of the previous lesson, the results of the quiz will also be counted as regular grade.

Because students have independently studied the teaching content and completed basic test questions according to the pre-class guide sheet, they have a certain foundation for relevant pattern thinking. In the classroom, teacher will be student-centered and will not simply repeat the knowledge points. After teacher briefly introduced the pattern case, let students spontaneously form a group of 3-5 people to discuss the characteristics of the model and cite similar cases. After teacher has taught the structure and core roles of design pattern, let students initiate a brainstorm to analyze pattern case they put forward, make the core role of the case and the role of the pattern simple association, which allows the students to fully participate and summarize the whole process. Students participate in the whole process of designing and realizing the design mode, and will play a role in cultivating students' innovative and practical ability.

Teaching material is "Java Design Pattern" published by Tsinghua University Press and edited by Mr. Liu Wei [14]. According to the content of teaching materials and syllabus, teachers make teaching courseware. Teaching courseware is made according to the content of the textbook and the syllabus, Excellent pattern cases are adopted in courseware, and they are integrated into course introduction and example part, in addition, ideological and political elements of the curriculum have been added to the design patterns’ teaching process, which make design pattern curriculum more suitable for practical applications and educational significance.

Teacher uses interesting small cases to analyze the knowledge points to break through difficult points summarized in the feedback sheet. Students' cognition and mastery of knowledge points are no longer fragmented and integrated under using actual projects to assist the analysis.

Design Pattern is a subject with strong practicality. In order to prevent students from being able to grasp the basic ideas and code implementation of design pattern by focusing on theory and neglecting practice, teacher have adopted the method of teaching theory lessons in the computer room, where courseware and blackboard can be demonstrated. And students can practice the synchronized training cases and expansion cases released by the teacher in real time after learning theory.

3.3. Teaching platform
MOOC adopts the Superstar Learning platform built by our school, which can enhance the interaction between teachers and students. MOOC platform provides functions such as learning videos, check-in, online exams, homework management, teacher-student interaction, and teaching process data analysis. The teaching process of teachers combines the use of MOOC with traditional teaching to improve teaching efficiency.
Teachers in our course group have applied MOOC to the teaching of "Java Design pattern" during the course. Figure 3 shows the data statistics of 49 students in Class One of Software Development Undergraduate Class of Grade 18 who used MOOC platform in October, among which the number of times of study reached 414 times on November 19th. It is convenient to use the MOOC platform to make statistics on the completion of homework, timely review, submit and give feedback, figure 4 show the completion of homework.

Figure 3. MOOC platform usage.

Figure 4. Completion of the assignment.

Applying MOOC platform to the teaching of "Java Design Pattern" courses makes classroom activities rich and diverse, improves learning interest, and achieves the effects of pre-class preparation and after-class assistance. In addition, teachers can also manage student learning data and learning progress in a visual manner, and adjust teaching process in real time, which has a better role in promoting students' mastery of knowledge points.

4. Conclusion
To sum up, the teaching reform of Java Design Pattern that combines SPOC and traditional classroom teaching to flexibly convert teaching methods has achieved good results, which completes the transition from teacher to student-oriented, and trains students' enthusiasm and autonomous learning ability. The students have improved their algorithm learning and programming skills, laying a good foundation for the following courses.

References
[1] Z. Azimuth, Y. S. An, and P. Denny, "Evaluating an Interactive Tool for Teaching Design
Patterns," presented at the Proceedings of the Twenty-Second Australasian Computing Education Conference, Melbourne, VIC, Australia, 2020.

[2] P. C. Lotlikar and R. Wagh, "Using POGIL to Teach and Learn Design Patterns — A Constructionist Based Incremental, Collaborative Approach," in 2016 IEEE Eighth International Conference on Technology for Education (T4E), 2016, pp. 46-49.

[3] M. Cohen, "Uncoupling Alice: using Alice to teach advanced object-oriented design," ACM Inroads, vol. 4, pp. 82–88, 2013.

[4] L. Zhang, Y. Li, and N. Ge, "Exploration on Theoretical and Practical Projects of Software Architecture Course," in 2020 15th International Conference on Computer Science Education (ICCSE), 2020, pp. 391-395.

[5] R. Kazman, "Reflections on Teaching Architecture Design," in 2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE T), 2020, pp. 1-1.

[6] A. Vázquez-Ingelmo, A. García-Holgado, and F. J. García-Peñalvo, "C4 model in a Software Engineering subject to ease the comprehension of UML and the software," in 2020 IEEE Global Engineering Education Conference (EDUCON), 2020, pp. 919-924.

[7] B. Pérez and A. L. Rubio, "A Project-Based Learning Approach for Enhancing Learning Skills and Motivation in Software Engineering," presented at the Proceedings of the 51st ACM Technical Symposium on Computer Science Education, Portland, OR, USA, 2020.

[8] S. Motogna, A. Marcus, and A.-J. Molnar, "Adapting to online teaching in software engineering courses," presented at the Proceedings of the 2nd ACM SIGSOFT International Workshop on Education through Advanced Software Engineering and Artificial Intelligence, Virtual, USA, 2020.

[9] T. Kanij and J. Grundy, "Adapting Teaching of a Software Engineering Service Course Due to COVID-19," in 2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE T), 2020, pp. 1-6.

[10] L. Gren, "A Flipped Classroom Approach to Teaching Empirical Software Engineering," IEEE Transactions on Education, vol. 63, pp. 155-163, Aug 2020.

[11] B. Westphal, "Teaching Software Modelling in an Undergraduate Introduction to Software Engineering," in 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 2019, pp. 690-699.

[12] S. Chren, B. Buhnova, M. Macak, L. Daubner, and B. Rossi, "Mistakes in UML Diagrams: Analysis of Student Projects in a Software Engineering Course," in 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), 2019, pp. 100-109.

[13] V. M. Flores Fonseca and J. Gómez, "Applying Active Methodologies for Teaching Software Engineering in Computer Engineering," IEEE Revista Iberoamericana de Tecnologías del Aprendizaje, vol. 12, pp. 182-190, Nov 2017.

[14] W. Liu, Java Design Pattern. Beijing: Tsinghua University Press, 2018.