Journal of ICT, Design, Engineering and Technological Science (JITDETS)
VOL. 6, NO. 1, pp. 21-25, 2022
DOI: https://doi.org/10.33150/JITDETS-6.1.3

ORIGINAL CONTRIBUTION

Interdisciplinary Learning of Low-Code Development Platform Programming with Dual Coding Theory-A Case Study of Agilepoint NX

Chung-Hsiang Wang[1*], Ko-Chiu Wu[2]

[1] Doctoral Program in Design, College of Design, National Taipei University of Technology, Taipei, Taiwan

[2] Department of Interaction Design, National Taipei University of Technology, Taipei, Taiwan

***Abstract*— This paper studies the interdisciplinary learning process of programming and computational thinking by design students trying to use a cloud-based low-code development platform with an information visualization interface. We researched and set up an 8-week "AgilePoint NX" programming course, which guided students with the introduction of computational thinking and learned from dual coding theory. In image process and low-code learning, we can carry out structured thinking and problem thinking and complete project tasks practice. We comprehensively analyze and evaluate their learning effectiveness through classroom learning observation, student sampling, brief interviews, and feedback. Finally, it is found that students from different backgrounds have different cognitive learning effects on interdisciplinary learning programs. Different factors on learning pain points lead to students' low learning of programming courses. Exploring their learning process can be used as a reference for interdisciplinary learning and low-code development platform programming.**

***Index Terms*— Interdisciplinary Learning, Computational Thinking, Low-Code Development Platform, Dual Coding Theory, Information Visualization**

Received: 11 January 2022; **Accepted:** 03 April 2022; **Published:** 20 June 2022

I. INTRODUCTION

A. Interdisciplinary Learning

In the global trend of technological progress, artificial intelligence, big data, and the growth of the Internet of Things, cross-border capabilities are considered fundamental to the ability to solve modern real-world problems [1]. Through the combined influence of different fields, interdisciplinary learning and related research lead to new learning issues in education. It has become an inevitable trend to use Computational Thinking (CT) to think or write programs in teaching practice to train students to learn together across fields. For example, teaching CT practical experience courses with robotics [2]. Wing believes that in addition to the importance of CT for computer applications and evaluation, the field of computing is driven by technological innovation, social needs, and scientific issues and has influenced other disciplines from science to art [3]. Furthermore, CT has become an essential cognitive skill to develop in all fields of education [4]. Students with a design background can learn about information-related fields in an interdisciplinary way through the study of CT.

B. Dual-Code Theory

Past studies have pointed out that most students with a design background are good at using images to understand and express their inner thoughts through visual thinking [5]. After all, visual thinking and computational thinking are different modes of thinking. Based on the influence of past visual thinking training, design students are often unable to organize and transform the hierarchical and structured thinking process of programming design effectively, so what should they do? The difficulty arises when starting to learn how to apply it. However, in the past, Paivio's dual-code theory research pointed out that different coding forms, visual and verbal, can generate a referential connection through the stimulation of different representations and coding forms (graphics and texts), automatically retrieve and connect the relevant symbols of the two systems, and combine different mental models. Generate tandem, strengthen cognition, and enhance memory [6, 7]. In addition, text stimuli can only produce language coding, while image stimuli can produce double coding of image and text and have the advantage of recognition. In learning, dual encoding of images and text will produce strong memory traces, which will help semantic storage and increase encoding strength [8, 9]. And studies have found that learning materials based on dual-code theory can improve students' comprehension more than just using text [10].

C. Information Visualization

For cross-domain learning programming, more evidence shows that learning and cognition can be enhanced through charts, graphic tools, mind maps, and information visualization. Information visualization can transform complex information into a means of presenting and processing information to users [11]. The ability to represent phenomena in various forms, such as graphics, mathematics, and charts, is the ability to make multiple

[*] Corresponding author: Chung-Hsiang Wang
[†] Email: designer175cm@gmail.com

representations [12]. When learning computer multimedia, information processing can be internalized into knowledge through visual content presentation and learners' learning methods [13, 14].

D. Low-Code Development Platform (LCDP)

The LCDP in recent years can be traced back to Model-driven Engineering (MDE) [15]. Research confirms that LCD and MDE share several core principles, such as abstraction, automation, visual symbols, and agility [16, 17]. The remarkable growth of LCDPs is gaining traction in the market and attracting interest from academia. LCDP is advertised as a visual development platform that requires little or no code, typically runs in the cloud, and is aimed at non-professional programmers, providing a solution for developing and operating complex software applications and reducing the need for manual coding and scheme [18]. Low code lowers the threshold for entering the modelling technology, attracting much attention from non-professionals and bringing tremendous opportunities into new fields and areas [15].

E. The Objective of the Study

Given this, and to improve this interdisciplinary learning programming problem for the Department of Interaction Design students, this research uses a cloud LCDP-AgilePoint NX to set up an interdisciplinary course to create forms, charts, block flow charts, and low-code for programming learning. And gradually introduce concepts such as database writing, API, and IoT through enterprise project examples to reduce the design background students' rejection of programming learning through the visual presentation of information and minimal code. It is hoped that through the learning effectiveness and thinking of interdisciplinary students in this course, a reference target can be established for future research on programming design for cross-domain learning.

II. METHODOLOGY

A. "AgilePoint NX" Program Course

We cooperated with the industry to set up and teach and opened an "AgilePoint NX" programming course in the third-year course of the Department of Interaction Design of NTUT. The participants were students of the department who came from high schools with different backgrounds in the past. This course is based on dual coding theory and guides students to introduce the concept of computational thinking to carry out logical and structured thinking in the process of image process and low-code learning and complete the practice and discussion of industry projects. The main structure of the 8-week course, such as Table I.

AgilePoint NX develops and uses a single application development platform for the industry, with rapid development, agility, and a powerful process engine, which can integrate all service processes and resources within the enterprise. Cloud-based services can provide application Platform-as-a-Service (PaaS) technology on the market. For general developers, any application can be developed with low-code or no-code. Software extensibility allows professional developers to create custom functions and APIs and easily integrate them into the graphical interface in AgilePoint NX, as shown in Fig. 1-2.

B. Classroom Observation and Brief Interview Survey

Based on the progress of the course, this research observes the interaction between students' learning and classroom teachers and conducts behavioural observation records to organize the thinking and learning pain points of programming learning. We also sampled five students who took the course, including one student with basic programming skills and four first-time programming students from a design background. A brief interview was conducted with them, and the main contents were the learning status, course content arrangement, and programming learning of this programming architecture platform—the brief interview questions, such as Table II.

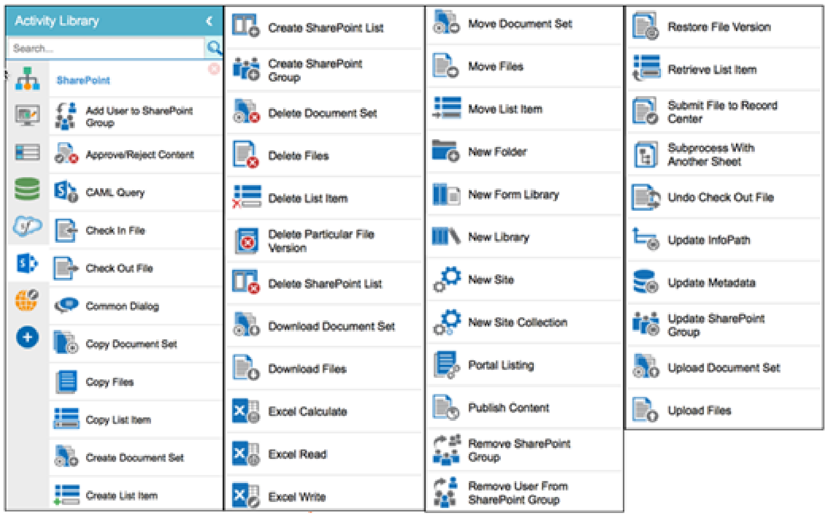

Fig. 1. Graphical Interface (Library)

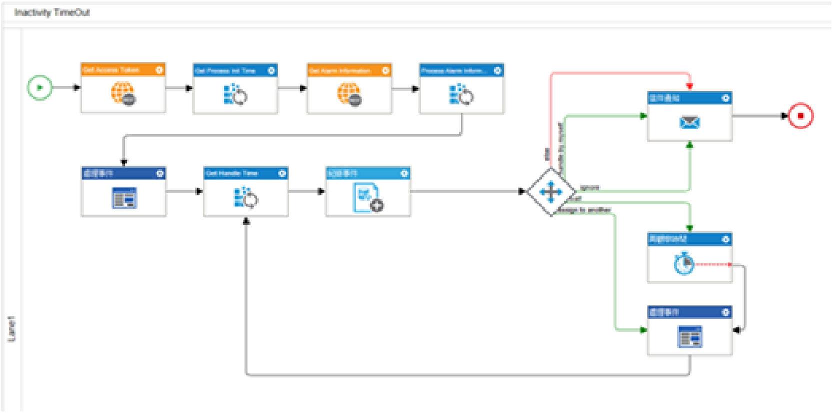

Fig. 2. Graphical Interface (Process Designer)

TABLE I
DESIGN COMPUTATIONAL THINKING COURSE

Week	Course Content	Course Content
1-2	Create a project process architecture concept	Design process and management, systematize and structure design, analyze user needs and interests, understand project purpose, and arrange process design and interaction.
3-4	AgilePoint NX Platform Operation	The AgilePoint NX platform is used to quickly construct process, business, organization, and program logic and form a graphical interface in an assembled manner.
5-6	Embedded application of external data	The AgilePoint NX platform external data link, combined with cloud computing, the Internet of Things, big data, and other technical applications.
7-8	Project practical application and operation sharing	Project discussion and results are published, and students come to the stage in groups to share the experience and results of the project.

TABLE II
BRIEF INTERVIEW QUESTIONS

Research Problem	Interview Topics
Steps and process of course implementation (students analyze ideas from activities)	• Please describe your steps or architecture when operating AgilePoint NX. • About the state of learning AgilePoint NX? • How do you evaluate your performance and understanding?
Learning difficulties and how to solve them	• Did you have any difficulty learning to AgilePoint NX programming? • Which help-seeking method did you use to learn?
The overall view of the course (the course content helps students to think)	• Summarize and organize your views and learning experiences on AgilePoint NX. • Suggestions or adjustments for course revisions.

III. RESULTS & DISCUSSION

This research aims to cultivate design students' computational and logical thinking and combine information visualization interface and cloud low-code development platform to guide programming development and carry out interdisciplinary learning processes.

According to the feedback from classroom observation and student interviews, as shown in Table III-IV, the research results indicate that students with a design background and students with a programming background have different needs and viewpoints when operating and learning the low-code development platform of visual learning. Design background students think that although the operating interface of the platform system is graphical, some operations are still biased towards engineering thinking and will be overwhelmed. For students with a programming background, the switching mode of the graphical interface is inconvenient to use when writing code. However, students generally believe that AgilePoint NX is still a development platform that can effectively integrate front-end to back-end. Establishing flowcharts and user permissions makes the system process rigorous and complete. Forms and processes can be created quickly and conveniently, and flowcharts can be configured. The operation is intuitive in image coding, and this process can be directly established and specified. If you can think about the fit of the project and it when designing the project combination and use it appropriately, I believe this can become a helpful tool. However, it is believed that this platform is not suitable for all development types. The extensive project support can be improved, the design interface design can be more intuitive and friendly, and some buttons to prevent errors can be provided.

In terms of learning pain points, there are still obstacles to students' learning of programming courses, which reduces learning motivation. The most critical factors are psychological factors and technical factors. Regarding psychological factors, students with a design background have cognitive fear and a sense of resistance to learning programming, which leads to distance and difficulty for students to resonate. Regarding technical factors, the course is from shallow to deep, but requires a more complex operation process, and also involves logical reasoning thinking, making it easier for students with psychological disabilities to withdraw and give up.

TABLE III
STUDENT LEARNING EXPERIENCE AND FEEDBACK

Background	Learning Advantage	Learning Disadvantage
Engineering	• The Build Pages function supports clicking buttons to link between different pages • Mockups can be crafted very well	• Graphical presentation and Low-Code are displayed at the same time, making it difficult to debug. • It is more inconvenient to graft external data API and must go through a third-party platform.
Design	• You can easily and quickly create web forms without writing complex code, construct an operation process, and simplify essential functions. • You can connect to the API and create a database of connections. • The primary purpose of Build Pages is to write web pages graphically. As long as you pull the Row and add content, you can create pages visually and superficially intuitively.	• The experience of using AgilePoint NX is inclined to the thinking mode of engineers. When using the “Front-end Design” function in Build Pages, it is not easy to use and operate, and there is a big gap between the vision and the finished product during production. • AgilePoint NX’s in-process setup items can sometimes confuse those without a programming background. For those who know the program, it is natural to know which project parameters to set, but those who do not have a basic knowledge of the program will not understand why to set some project parameters that do not know the function or influence.

TABLE IV
PROGRAMMING OF PAIN POINTS FOR INTERDISCIPLINARY LEARNING

Programming of Pain Points
• Students with a design background tend to have cognitive fears about program development. Students who think graphically have difficulty operating the narrative structure of programs. Although AgilePoint NX is a platform designed with a graphical interface, its experience is biased toward the engineer’s mode of thinking. • Students who are biased toward a programming background think that AgilePoint NX writes projects in a graphical drag-and-drop mode, it is difficult to modify the original data, and the operation does not conform to the habits of programming users, so it cannot generate resonance points. • Lack of understanding of the importance of the project development process and basic knowledge of concepts, resulting in a lack of interest in course content. • Students are accustomed to using their familiar software operations to solve most of the project problems, but they do not have to think deeply about the limitations and features of each software. • The AgilePoint NX platform course design is closely linked, which is more suitable for learners with basic programming knowledge. Design students with no programming concepts must master programming concepts while learning software operations. Due to the schedule of eight courses, the study time is compressed. As long as a study stage is left in the middle, keeping up with the follow-up will be difficult, causing students to withdraw and give up. • The operation of serial database connection and the grafting of external data is complex for ordinary students to absorb and cannot understand. It not only requires a more complicated operation process but also involves logical thinking. However, most students with a design background still prefer to think on the design side, and it was observed that most students gave up in the fifth week of the course.

IV. CONCLUSION

In a broad sense, programming is a series of logical instruction combinations and a systematic problem-solving process. We try to integrate design and programming thinking concepts through the graphical low-code development platform and information. The concept of process architecture guides designers to think so that designers can think more logically about the overall design architecture process so that visual thinkers can communicate more smoothly in groups after mastering the overall logical thinking and even develop the executable program.

Students’ motivation to study the course is still not high, and perfect course design is still needed to guide students into this field and build their interest. In the future, we will revise the course based on students’ experience sharing and feedback and follow-up measurement and data analysis. It is hoped that it can cultivate students’ thinking mode and combine sensibility and rationality to solve problems to connect the interface with front-end programs. Interaction design students can propose humanized and highly feasible operation interfaces and interactions.

ACKNOWLEDGMENT

We want to thank teacher Vincent Chu of agile point, Inc., and students of the third-year course of the interaction design department, National Taipei University of Technology, Taiwan, where we developed the study, agile point, Inc., for supporting the activities.

References

[1] D. Kuiphuis-Aagten, K. M. Slotman, and M. A. MacLeod, ``Interdisciplinary education: A case study at the university of twente,'' in *47th SEFI Annual Conference 2019-Varietas Delectat: Complexity is the New Normality*, Budapest, Hungary, 2019.

[2] B. Díaz-Lauzurica and D. Moreno-Salinas, ``Computational thinking and robotics: A teaching experience in compulsory secondary education with students with high degree of apathy and demotivation,'' *Sustainability*, vol. 11, no. 18, p. 5109, 2019. doi: https://doi.org/10.3390/su11185109

[3] J. M. Wing, ``Computational thinking and thinking about computing,'' *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3717-3725, 2008. doi: https://doi.org/10.1098/rsta.2008.0118

[4] P. J. Rich, G. Egan, and J. Ellsworth, ``A framework for decomposition in computational thinking,'' in *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, Aberdeen, Scotland, 2019. doi: https://doi.org/10.1145/3304221.3319793

[5] R. E. Wileman, *Englewood Cliffs, NJ: Visual communicating*. Educational Technology, 1993.

[6] A. Paivio, ``Dual coding theory: Retrospect and current status,'' *Canadian Journal of Psychology/Revue Canadienne de Psychologie*, vol. 45, no. 3, pp. 255-287, 1991. doi: https://doi.org/10.1037/h0084295

[7] A. Paivio, *Imagery and verbal processes*. New York, NY: Psychology Press, 2013.

[8] A. Paivio, *Mental representations: A dual coding approach*. Oxford university press, 1990.

[9] D. M. McBride and B. A. Dosher, ``A comparison of conscious and automatic memory processes for picture and word stimuli: A process dissociation analysis,'' *Consciousness and cognition*, vol. 11, no. 3, pp. 423-460, 2002. doi: https://doi.org/10.1016/S1053-8100(02)00007-7

[10] Suhartono, D. Mulyanti, Purwadhi, R. Agusiady, A. D. J. Suhandoko, and Hanafi, ``The integration of Science Material, Technology and Society (STS) from dual coding theory's perspective in distance education module.'' *International Journal of Advanced Science and Technology*, vol. 29, no. 5, pp. 4649-4659, 2020.

[11] G. E. Raptis, C. A. Fidas, and N. M. Avouris, ``Using eye tracking to identify cognitive differences: A brief literature review,'' in *Proceedings of the 20th Pan-Hellenic Conference on Informatics*, New York, NY, 2016. doi: https://doi.org/10.1145/3003733.3003762

[12] S. Yunus, M. Tawil, N. Muhiddin, S. Muhiddin, and M. Alim, ``Describing representation ability of prospective science teacher based on learning style,'' in *2nd Workshop on Engineering, Education, Applied Sciences and Technology (WEAST)* , Makassar, Indonesia, 2021. doi: https://doi.org/10.1088/1742-6596/1899/1/012141

[13] I. K. R. Y. Negara, ``The effect of animation in multimedia computer-based learning and learning style to the learning results,'' *Turkish Online Journal of Distance Education*, vol. 18, no. 4, pp. 177-190, 2017. doi: https://doi.org/10.17718/tojde.340409

[14] A. Q. Saeed, M. T. Sultan, and K. N. Yasen, ``Simulating mechanical engineer equations using C++ programming,'' *International Journal of Technology and Engineering Studies*, vol. 5, no. 5, pp. 163-168, 2019. doi: https://dx.doi.org/10.20469/ijtes.5.10004-5

[15] J. Cabot, ``Positioning of the low-code movement within the field of model-driven engineering,'' in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, New York, NY, 2020. doi: https://doi.org/10.1145/3417990.3420210

[16] A. Bucaioni, A. Cicchetti, and F. Ciccozzi, ``Modelling in low-code development: A multi-vocal systematic review,'' *Software and Systems Modeling*, vol. 21, no. 5, pp. 1959-1981, 2022. doi: https://doi.org/10.1007/s10270-021-00964-0

[17] K. I. D. Laurel and P. L. Fernandez, ``Flash codes with multibit update for asymmetrical write memory device,'' *Journal of Advances in Technology and Engineering Research*, vol. 3, no. 1, pp. 11-18, 2017. doi: https://doi.org/10.20474/jater-3.1.3

[18] D. Di Ruscio, D. Kolovos, J. de Lara, A. Pierantonio, M. Tisi, and M. Wimmer, ``Low-code development and model-driven engineering: Two sides of the same coin?'' *Software and Systems Modeling*, vol. 21, no. 2, pp. 437-446, 2022. doi: https://doi.org/10.1007/s10270-021-00970-2