KIPS Trans. Comp. and Comm. Sys.
Vol.10, No.4 pp.107~116
ISSN: 2287-5891 (Print), ISSN 2734-049X (Online)

https://doi.org/10.3745/KTCCS.2021.10.4.107

Applied Practices on Codification Through Mapping Design Thinking Mechanism with Software Development Process

Chae Yun Seo† · Jang Hwan Kim†† · R.Young Chul Kim†††

ABSTRACT

In the 4th Industrial Revolution situation it is essential to need the high quality of software in diverse industrial areas. In particular current software centered schools attempt to educate the creative thinking based coding to non-major students and beginners of computer. But the problem is insufficient on the definition and idea of the creative thinking based software. In addition in a aspect of coding education for non-major and new students we recognize to have no relationship between creative thinking methods and coding. In other words we should give them how to practically code and design through learning the creative thinking. To solve this problem we propose the codification of design thinking mechanism without the knowledge of software engineering through mapping creative thinking with software development process. With this mechanism we may expect for students to have some coding ability with the creative design.

Keywords : Design Thinking, Software Engineering, Software LifeCycle, Coding Education Process, Creative Thinking

소프트웨어개발 프로세스와 디자인씽킹 메커니즘의 접목을 통한 코딩화 적용 사례

서 채 연† · 김 장 환†† · 김 영 철†††

요 약

4차 산업혁명 시대가 도래함으로써 수많은 영역에 다양한 소프트웨어의 고품질화가 필수적이다. 특히 비전공자 및 기초 전공자들에게는 창의적 사고 기반으로 코드 할 수 있는 능력이 요구된다. 하지만 문제는 창의적 사고 기반의 소프트웨어에 대한 정의 및 아이디어가 부족하다는 것이다. 또한, 비전공자 및 기초 전공자를 위한 코딩 교육 영역에서, 창의적 사고 기반 디자인씽킹과 코딩화는 괴리가 존재한다. 즉 실질적으로는 창의적 사고기법을 통해서 소프트웨어 설계 및 코딩이 가능해야 한다는 점이다. 이런 문제를 해결하기 위해, 창의적 사고 기법과 소프트웨어 개발 프로세스 기법의 접목을 통해 비전공자의 소프트웨어공학 개념 없이도, 디자인씽킹 메커니즘 기반의 코드 템플릿을 제시한다. 이를 통해 창의적 설계의 코딩화를 기대한다.

키워드 : 디자인씽킹, 소프트웨어공학, 소프트웨어 생명주기, 코딩 교육 프로세스, 창의적 사고

1. 서 론

4차 산업혁명 시대 소프트웨어는 지능형, 맞춤형으로 진화되고 있다. 특히 지능 정보와 정보 통신 기술 융합 환경에서는 창의성, 사고력, 정보 수집, 처리, 활용 능력, 문제 해결력의 중요성이 필요하다[1]. 앞으로 창의적 인재양성은 문제 발견과 확장된 사고, 그리고 논리적으로 분석하고 의미에 따라 표현하는 사고를 해야 한다. 또한, 해결책을 찾기 위해 융/복합하는 사고가 필요하다[2].

비전공자/전공자들을 위한 창의적 문제 해결 방법인 디자인씽킹과 컴퓨테이셔널 씽킹을 통해 문제를 해결하려는 시도가 매우 필요하다. 문제는 비전공자들은 코딩을 어려워한다는 것이다. 창의적 생각은 실현 가능할지 모르나, 코딩과 연결 고리를 찾지 못하고 있다. 이 문제를 해결하기 위해, 본 논문에서는 소프트웨어공학적 디자인씽킹 기반 코딩 개발 방안을 제안한다. 이 방법은 소프트웨어공학 개발과 디자인씽킹 메커니즘을 접목한다. 디자인씽킹 관점의 창의적 사고기법을 통해 비전공자/기초 전공자들은 정확한 문제 인식 및 해결 능력을 배양하고 창의 융합적 문제 해결이 가능하게끔 코딩

※ 본 논문은 2020년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2020R1I1A1A01072928).
※ 이 논문은 2020년 한국정보처리학회 춘계학술발표대회에서 “디자인 씽킹 메카니즘과 소프트웨어공학 접목에 관한 연구”의 제목으로 발표된 논문을 확장한 것임.
† 정 회 원 : 홍익대학교 소프트웨어융합학과 강사
†† 준 회 원 : 홍익대학교 소프트웨어융합학과 석사과정
††† 정 회 원 : 홍익대학교 소프트웨어융합학과 교수
Manuscript Received : July 28, 2020
First Revision : October 29, 2020
Second Revision : January 18, 2021
Accepted : February 3, 2021
* Corresponding Author : R.Young Chul Kim(bob@hongik.ac.kr)

성숙 능력을 향상 시킬 수 있다.

본 논문의 2장에서는 관련 연구로서 소프트웨어개발 프로세스와 디자인씽킹을 소개한다. 3장에서는 소프트웨어개발 프로세스와 디자인씽킹 프로세스의 접목을 제안하고 소개한다. 4장에서는 결론과 함께 향후 연구 방향에 관해 서술한다.

2. 관련 연구

2.1 Software Engineering (SE)

소프트웨어가 발전함에 따라 소프트웨어개발의 규모가 점점 커지면서 보다 체계적이고 시스템적인 개발이 요구되어왔다. Fig. 1은 소프트웨어공학 개발 프로세스 기반 도어락 시스템을 보여준다[3]. 소프트웨어공학 개발 프로세스의 수직적 매핑관계이다. 도어락 시스템을 만들 때, 필요한 소프트웨어 개발 프로세스의 수직적 매핑관계를 보여준다.

Table1은 소프트웨어 개발 주기의 각 단계별 산출물이다.

1) 요구사항

소프트웨어 개발 프로세스의 첫 번째 단계인, 요구사항단계이다. 요구사항이란 시스템 개발 분야에서 어떤 과제를 수행하는데 필요한 조건이나 능력을 말한다[4]. 고객이 자연어로 기술한 내용을 기반으로 요구사항 스펙을 정의 한다. Fig. 2는 도어락 시스템 개발을 위해 자연어로 작성한 요구사항 스펙정의서이다.

Step1. 요구사항 스펙 정의

요구사항단계에서는“Step1요구사항스펙정의“를 작성한다.

2) 분석

두 번째 단계인, 분석단계는 시스템 파악 및 문제점을 분석하고 사용자 인터뷰를 통해 새로운 요구사항들을 도출하여 수집한다. 고객의 요구사항으로부터 유즈 케이스를 추출 한다[5]. 시스템 상위 수준에서 환경과 사람들 간의 관계 그리고 모든 이해 관계자들이 시스템을 이해하는 것이 중요하다[5]. 이 단계에서는 Step1의 요구사항 스펙 정의를 기반으로 개발시스템의 유즈케이스를 시스템 기능으로 분할한다.

Step2. 유즈케이스 다이어그램 추출

Fig. 3은 개발시스템의 기능 분석을 통해 작성한 유즈케이스 다이어그램이다.

Table 1. Horizontal Mapping Relationship

SoftwareEngineering	Artifact
Requirement	Requirement Specification
Analysis	Use-case Diagram
Design	Class Diagram Sequence Diagram
Implementation	Code Template

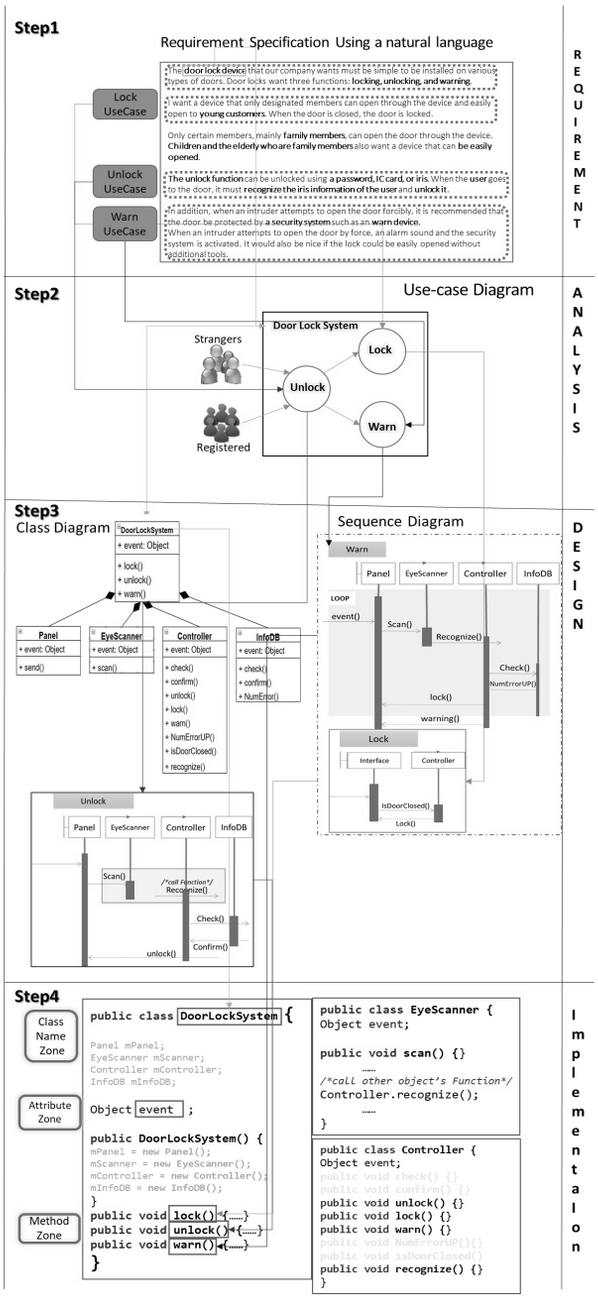


Fig. 1. Software Development Process

The **door lock device** that our company wants must be simple to be installed on various types of doors.
Door locks want three functions: **locking, unlocking, and warning**.

I want a device that only designated members can open through the device and easily open to **young customers**.
When the door is closed, the door is locked.

Only certain members, mainly **family members**, can open the door through the device.
Children and the elderly who are family members also want a device that can **be easily opened**.

The unlock function can be unlocked using **a password, IC card, or iris**.
When the **user** goes to the door, it must **recognize the iris information of the user** and **unlock it**.

In addition, when an intruder attempts to open the door forcibly,
it is recommended that the door be protected by **a security system** such as an **warn device.**
When an intruder attempts to open the door by force, an alarm sound and the security system is activated.
It would also be nice if the lock could be easily opened without additional tools.

Fig. 2. Requirement Specification Using a Natural Language

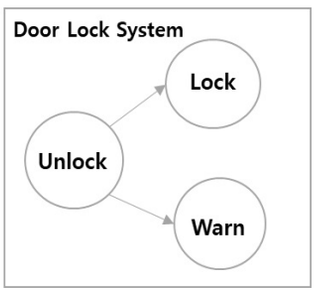


Fig. 3. Use-case Diagram

분석단계에서는 Step1의 요구사항 스펙 정의를 기반으로 도어락 시스템의 기능을 분석하여 Step2. 유즈케이스 다이어그램을 추출한다. 유즈케이스 다이어그램으로 추출한 도어락 시스템의 기능은 'Lock', 'Unlock','Warn'이다.

3) 설계

설계단계에서는 분석단계에서 표현한 도구들을 바탕으로 시스템 및 구조를 설계한다[6]. Fig. 4는 도어락 시스템의 클래스다이어그램이다. 도어락 시스템은 입력정보를 오브젝트 타입으로 받고, 도어락 기능으로 lock(), unlock(), warn()기능이 있다. 도어락 시스템의 객체로 Panel, EyeScanner, Controller, InfoDB가 있다. 각각의 기능을 수행하기 위해 필요한 속성과 메소드를 확인한다[7]. 설계단계에서 Step3. 클래스다이어그램 추출한다.

Step3. 클래스다이어그램 추출 단계 (Fig. 4)

Step4. 유즈케이스 순차 분석

유스 케이스로 작성된 각 순차 다이어그램을 분석한다.

Fig. 5는 도어락 시스템 Unlock기능의 순차 다이어그램이다. a) Unlock Sequence Diagram : Unlock()기능의 데이터 흐름과 메소드 호출 관계를 분석한다[8].

Fig. 6은 도어락시스템 Warn기능 순차 다이어그램이다.

b) Warn Sequence Diagram : 시스템에서 Warn() 기능이 어떻게 호출되는지 알 수 있다[9].

Fig. 7은 도어락시스템 Lock기능의 순차 다이어그램이다.

c) Lock Sequence Diagram : 자물쇠가 잠기는 흐름도를 알 수 있다.

4) 구현

구현단계는 코드를 구현하는 단계이다. 시스템의 모든 코드를 작성하지 않고, 설계단계에서 추출된 기능들을 스켈레톤 코드 형태로 반자동 발생한다. 클래스다이어그램으로부터 스켈레톤 코드를 반 자동으로 발생한 코드 템플릿이다[10]. 설계 단계에서 추출된 Step3, Step4를 기반으로 Step5. 스켈레톤 코드 템플릿을 작성할 수 있다.

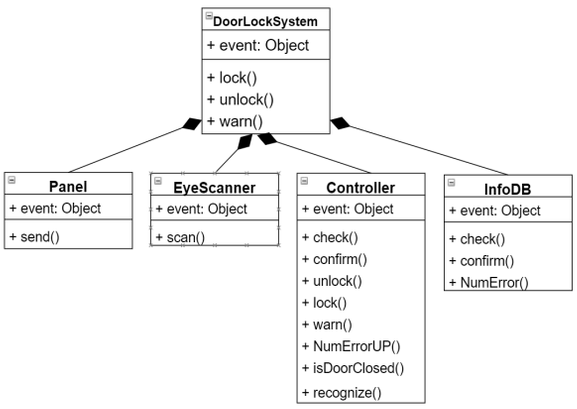


Fig. 4. Class Diagram

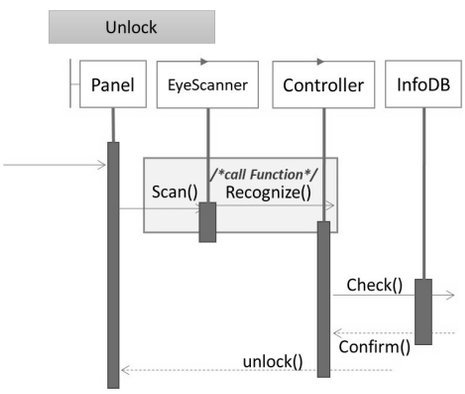


Fig. 5. Unlock Sequence Diagram

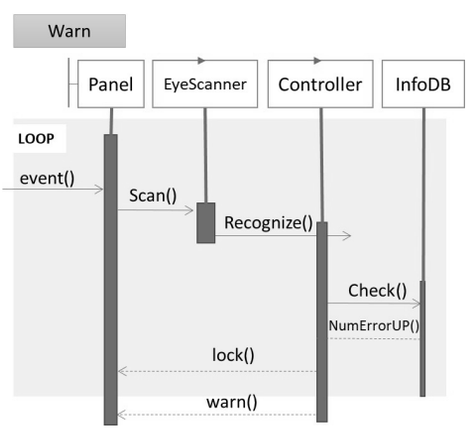


Fig. 6. Warning Sequence Diagram

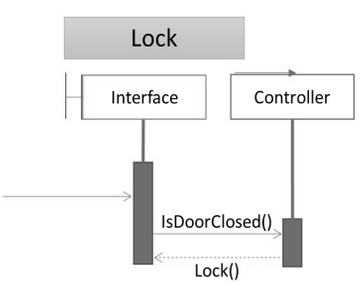


Fig. 7. Lock Sequence Diagram

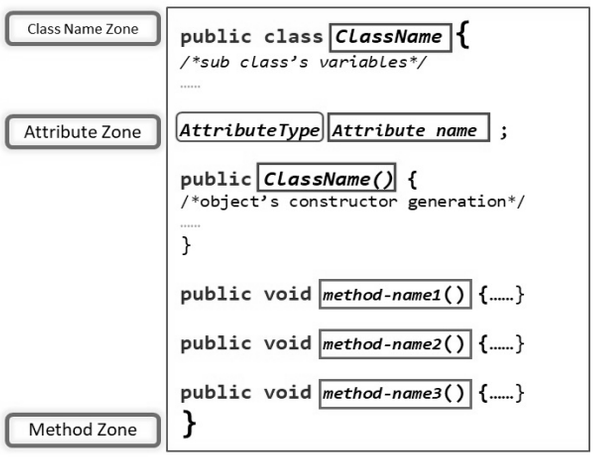


Fig. 8. Skeleton Code Template of Composition Relationship

Step5. 스켈레톤 코드 템플릿

Fig. 8은 설계 단계에서 추출된 기능들을 스켈레톤 코드로 자동 발생한 코드 템플릿이다.

ClassNameZone에는 시스템의 클래스네임(*ClassName*)을 적는다. AttributeZone에는 변수 타입(*AttributeType*), 변수 이름(*Attribute name*)을 적는다. MethodZone에는 클래스의 메소드를 적는다. *method-name1(),method-name2(),method-name3()*과 같이 작성한다. 다음은 도어락 시스템 개발의 디자인씽킹 과정을 언급한다.

2.2 Design Thinking (DT)

디자인 씽킹의 목표는 창의적으로 문제를 해결하기 위해 확산적 사고와 수렴적 사고를 결합하여 다양한 사고 능력을 키우는 것이다[11]. Fig. 9는 디자인씽킹 프로세스이다. 디자인씽킹은 공감과 정의를 기반으로 창의적 아이디어 발상, 창의적 아이디어 표현을 한다. 아이디어를 제품으로 구현하고, 시제품 테스팅 후, 피드백을 받고 개선해 나간다.

Fig. 9는 디자인씽킹의 수직적 매핑관계를 보여준다.

1) 공감

공감 단계에서 AEIOU관찰법을 사용하여 공감한다[12].

Fig. 10은 AEIOU관찰법에 사용되는 요소이다. AEIOU구성은 활동, 환경, 상호 작용, 개체 및 사용자 5가지 요소를 의미한다.

- Activity-활동, 사람들이 달성하고 싶은 행동, 목표
- Environment-활동이 일어나는 전체의 영역, 전체공간의 기능과 특징
- Interaction-사람간 또는 그 밖의 것들의 활동
- Object-기능, 객체, 장치와의 연관성있는 활동
- User-사용자, 환경 및 관찰이 요구되는 사람들

Table 2처럼, 공감단계에서 사용자가 원하는 도어락 시스템을 만들기위해 AEIOU관찰법을 활용한다. 도어락 시스템

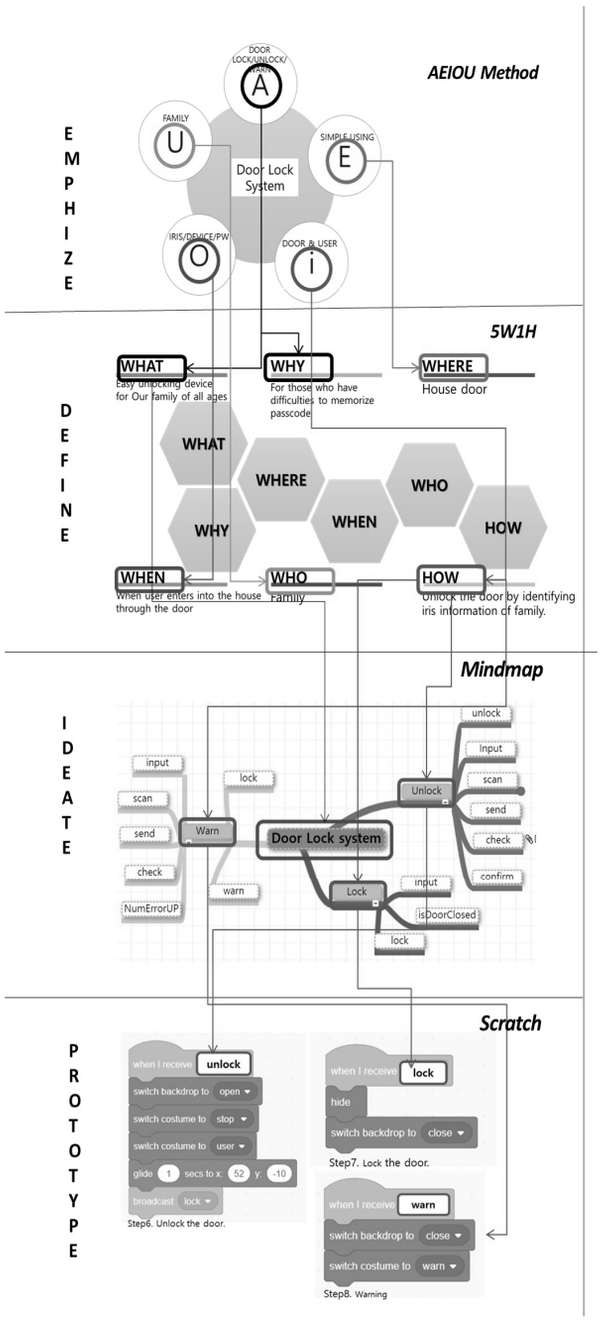


Fig. 9. Design Thinking Process

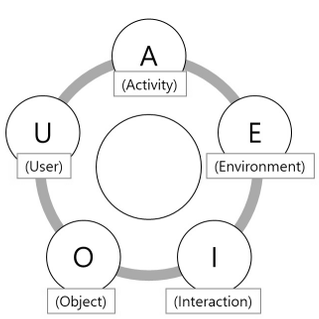


Fig. 10. DT Empathy "AEIOU"

Table 2. AEIOU Method

The AEIOU observation method is very helpful in solving the problem for those in contact (observing the user). *A: Activity* *Users can lock or unlock the door.* *E: Environment* *Door unlock can be use only family.* *I: Interaction* *Door interacts with family members.* *O:Objects* *Door lock system consists of Lock/Unlock/Warning* *U:User*

Table 3. Defined 5W1H Method

Empathy AEIOU	Define 5W1H
Activity	What,Why
Environment	Where
InterAction	How
Object	When
User	Who

의 AEIOU공감을 통해 다음 요소들을 추출할 수 있다.

2) 정의

정의 단계는 사용자와 공감한 문제를 정의하는 단계이다.

우리는 문제를 정의하기 위한 기법으로 5W1H를 사용한다[13]. Table 3은 공감 단계에서 사용한 AEIOU요소들을 정의단계에서 5W1H와 매핑하여 문제정의를 한다.

도어락 시스템이 갖고있는 문제를 5W1H기법을 활용하여 정의한다. 필요한 장소, 사용자, 어떻게, 언제, 어떤 시스템을 만들 것인지 정의할 수 있다. 정의 단계에서 도출한 도어락 시스템은 ”등록한 가족만 Unlock 기능을 수행할 수 있고, 문이 열리면 자동으로 Lock기능이 실행되고, 등록되지않은 사용자가 Unlock을 시도할 경우, Warn이 실행되는 시스템“이다.

- What-목적, 다양한 작업 또는 기능 영역
- When-사용기한 또는 시간
- Where-사용 위치, 기능 위치
- Why-사용 이유
- Who-이해 관계자, 영향을 받는 사람 또는 책임자
- How-기능, 명령 구조, 흐름, 기술 및 사용 절차

Fig. 11은 5W1H기법으로 도어락 시스템을 정의한다. 정의단계는 5W1H기법을 활용하여 도어락 시스템을 정의할 수 있다.

3) 아이디어

아이디어 단계는 정의한 문제를 해결하기 위한 단계이다. 정의 단계에서 도어락 시스템을 5W1H기법으로 정의한 결과를 아이디어 기법 중 마인드 맵을 활용하여 적용한다.

Fig. 12에서는 시스템 정의가 마인드맵을 활용하여 추출한다.

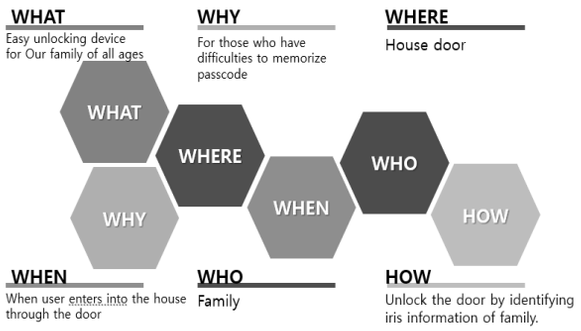


Fig. 11. DT Define for 5W1H Method

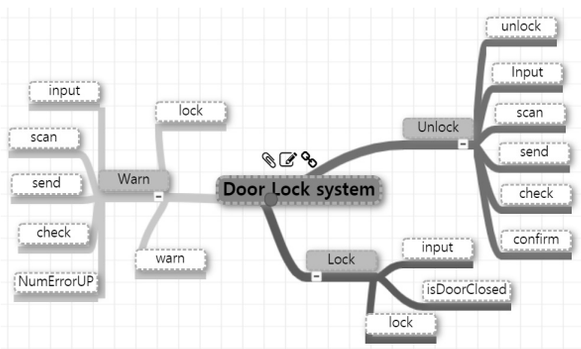


Fig. 12. DT Ideate for Mindmap (System Function)

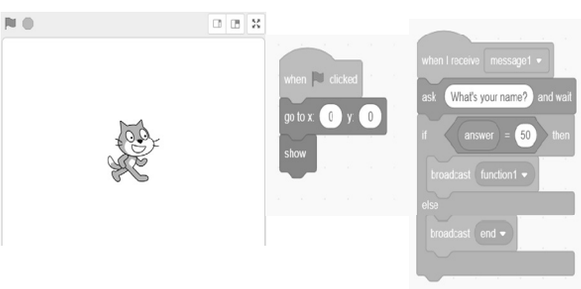


Fig. 13. Scratch Script for Prototype

4) 프로토타입

프로토타입 단계는 마인드 맵으로 작성한 시스템 기능을 스크래치로 시뮬레이션한다. 즉, 개발 전에 시스템을 가상으로 시뮬레이션 할 수 있다. 프로토타입 단계에서 스크래치 스크립트를 작성하여 도어락 시스템 기능을 시뮬레이션 할 수 있다.

Fig. 13은 프로토타입 단계에서 개발 시스템 기능을 시뮬레이션 할 수 있는 스크립트 코드이다.

5) 테스트

테스트 단계는 프로토타입이 잘 되었는지 확인하는 단계이다. 만약, 수정 작업이 필요하다면 앞 단계로 돌아가서 충분한 작업을 거친 후 다음 작업을 한다.

3장에서 나오는 디자인씽킹과 소프트웨어공학 매핑 예제는 도어락 시스템을 예제로 든다.

3. 디자인씽킹(DT)과 소프트웨어공학(SE) 개발 접목

우리는 디자인씽킹 프로세스를 알면 소프트웨어공학 개발 절차를 모르더라도 쉽게 코딩할 수 있는 매핑 방법을 제안한다. 디자인씽킹 프로세스의 각 단계별 요소를 소프트웨어공학 개발 주기의 각 단계와 매핑하여 쉽게 코딩 가능한 매핑 프로세스이다. Fig. 14는 소프트웨어개발 라이프사이클과 디자인씽킹 프로세스를 매핑한 소프트웨어공학 기반 디자인씽킹 코딩 프로세스이다. Fig. 14에서는 두 개의 프로세스 간 매핑 관계는 각 단계를 따른다. 소프트웨어공학은 SE로, 디자인씽킹은 DT로 표기한다.

3.1 DT Empathy & SE Requirement

첫 번째 단계는, DT “공감”과 SE “요구사항” 매핑이다.

먼저, 디자인씽킹 공감 “AEIOU관찰법” 요소를 추출한다. 추출된 DT의 AEIOU 관찰법 요소들을 사용해서 소프트웨어공학의 요구사항스펙을 작성한다. Table 3은 SE의 요구사항 단계와 DT의 공감 단계를 매핑한 것이다[14].

디자인씽킹의 공감 관찰법인 ‘AEIOU’는 Activities(행동), Environments(환경), Interactions(상호작용), Objects(사물), User(사람)의 첫 글자를 딴 관찰법으로써 데이터를 요소별로 구분하여 관찰하거나 조사한 데이터들을 각각의 요소별로 나누어 분석할 수 있게 한다[14]. 이 각각의 요소들을 사용해서 소프트웨어공학의 요구사항스펙을 만든다[15].

Table 4는 디자인씽킹 AEIOU 관찰법으로 도어락 시스템의 필요성을 공감하고, 디자인씽킹의 요소를 사용하여 소프트웨어공학의 요구사항정의서를 만든다.

Table 5는 도어락 시스템의 필요성을 공감한 요소들을 사용하여 소프트웨어공학 첫 번째 단계인 요구사항정의를 자연어로 매핑한 결과이다[16].

Fig. 15는 디자인씽킹 공감 AEIOU관찰법 요소로부터 추출된 소프트웨어공학 요구사항명세서이다. 첫 번째 매핑 과정이 끝나면, 디자인씽킹의 공감을 통해 소프트웨공학의 요구사항을 명세할 수 있다[17]. Fig. 14는 디자인 씽킹과 소프트웨어공학 개발 프로세스의 수직적 수평적 매핑관계이다.

Table 4. Mapping DT with SE at Step1

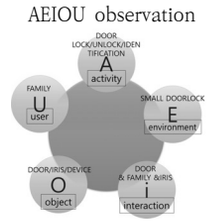
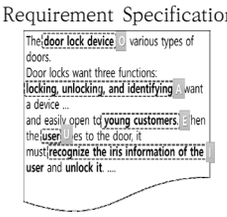
	Design Thinking		Software Engineering	
Empathy	AEIOU observation 		Requirement Specification 	Requirement

Table 5. Each Related Elements between DT and SE with an Example of DoorLockSystem at Step1

DT AEIOU	SE RequirementSpec.
Activity	Functions of DoorLockSystem
Environment	Use the Family
Interaction	DoorLockSystem&User
Object	Iris/ICcard/Keypad
User	Registered Family

3.2 DT Define & SE Analysys

두 번째 단계는, DT “정의”와 SW “분석” 매핑이다.

매핑 1단계에서 추출된 요소를 기반으로 두 번째 단계에서는, 디자인씽킹 정의 “5W1H”요소를 추출한다. 그리고, 추출된 DT의 5W1H요소를 활용해 소프트웨어공학의 유즈케이스 다이어그램을 작성한다[18].

Table 6은 디자인씽킹 정의단계 5W1H기법으로 도어락 시스템을 정의하고, 각 요소들을 소프트웨어공학의 분석 단계에서 도어락 시스템 기능으로 정의해 유즈케이스 다이어그램을 그린다.

Table 7은 디자인씽킹의 5W1H 기법을 활용해 What을 도어락 시스템으로 정의한다. Why와 Who를 시스템 사용자로 정의, Where은 현관문으로 정의, When은 집에 들어갈 때, How는 도어락 시스템 기능으로 정의한 후, 소프트웨어공학 분석 단계의 유즈케이스 다이어그램을 그린다. 두 번째 매핑 과정이 끝나면, 디자인씽킹의 정의를 통해 소프트웨공학의 유즈케이스 다이어그램을 명세할 수 있다.

Table 6. Mapping DT with SE at Step2.

	Design Thinking		Software Engineering	
DEFINE	5W1H method • WHAT-DoorLockSystem • WHY-forTheElderlyfamily • WHERE-HouseDoor • WHEN-EnterHome • WHO-Family • HOW-Unlock/Lock/Warn		Use-case Diagram 	ANALYSYS

Table 7. Each Related Elements between DT and SE with an Example of DoorLockSystem at Step2.

DT 5W1H	SE Use-case Diagram
What	DoorLocksystem
Why	forTheElderlyFamily
Where	HouseDoor
When	EnterHome
Who	Registerd Family
How	Unlock/Lock/Warn

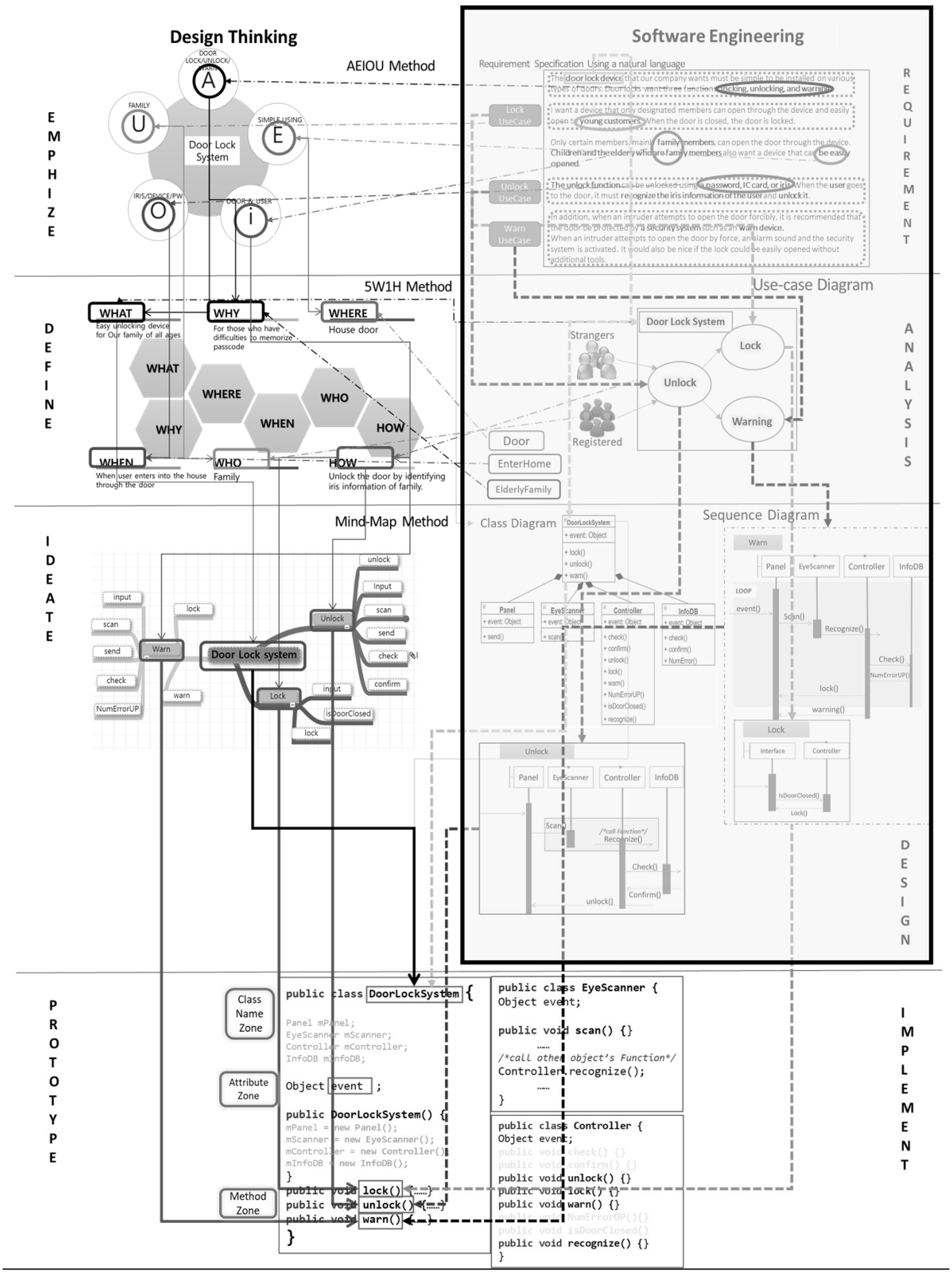


Fig 14. A Coding Development Process Based on Design Thinking and Software Engineering

The **door lock device** that our company wants must be simple to be installed on various types of doors.
Door locks want three functions: **locking, unlocking, and warning**.

I want a device that only designated members can open through the device and easily open to **young customers**.
When the door is closed, the door is locked.

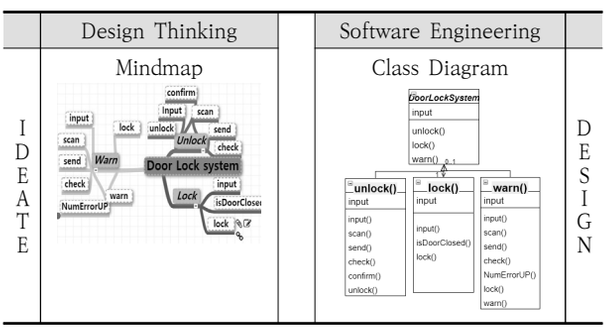
Only certain members, mainly **family members**, can open the door through the device.
Children and the elderly who are family members also want a device that can **be easily opened**.

The unlock function can be unlocked using **a password, IC card, or iris**.
When the **user** goes to the door, it must **recognize the iris information of the user** and **unlock it**.

In addition, when an intruder attempts to open the door forcibly,
it is recommended that the door be protected by **a security system** such as an **warn device.**
When an intruder attempts to open the door by force, an alarm sound and the security system is activated.
It would also be nice if the lock could be easily opened without additional tools.

Fig. 15. Software Engineering Requirements Specification Extracted from Design Thinking AEIOU Observation Method

Table 8. Mapping DT with SE at Step3.



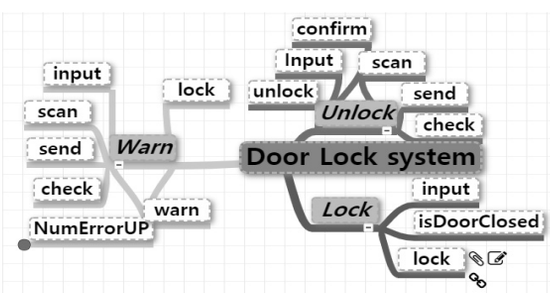


Fig. 16. DT “Ideate” Mindmap (Example Door Lock System Function)

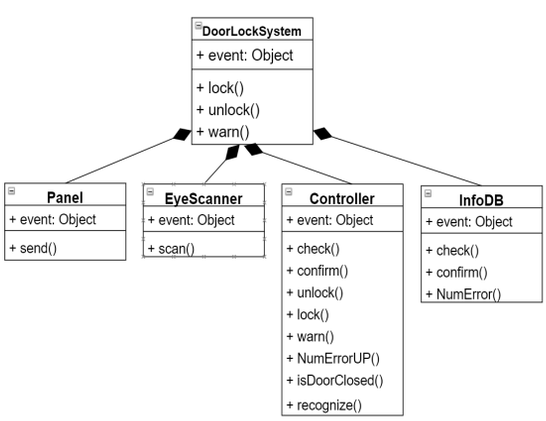


Fig. 17. “Design” Class Diagram in SE (Door Lock System)

Table 9. Each Related Elements between DT and SE with an Example of DoorLockSystem at Step3.

DT Mindmap	SE Class Diagram
DoorLockSystem	DoorLockSystem
Unlock	Unlock
Lock	Lock
Warn	Warn

3.3 DT Ideate & SE Design

세 번째 단계는, DT “아이디어”와 SE “설계”의 추상설계 매핑이다.

SW 설계단계를 추상설계와 상세설계로 나누어 매핑한다. 세 번째 매핑단계 과정은, 마인드맵과 클래스다이어그램이다. 매핑 2단계, “How“에서 추출된 도어락 시스템 Unlock/Lock/Warn 기능을 디자인씽킹 아이디어 단계 “마인드맵”으로 그린다. 마인드 맵 표현 시, 각 기능이 갖을 수 있는 작은 기능을 표현한다. 이렇게 표현된 기능을 기반으로 도어락 시스템의 전체 구조를 소프트웨어공학의 클래스다이어그램으로 작성한다. 추상설계는 소프트웨어공학 설계단계의 클래스다이어그램을 그린다.

Table 8은 도어락 시스템의 기능을 디자인씽킹 아이디어 단계 마인드기법으로 작성한 후, 기능요소를 기반으로 소프트웨어공학 디자인 단계 클래스다이어그램과 매핑한다.

Fig. 16은 디자인씽킹 아이디어 단계 마인드맵으로 표현한 도어락 시스템 기능이다. Door Lock system을 가운데로 시스템 주요 기능 Unlock,Lock,Warn을 표현한 후, 기능 안에 들어갈 수 있는 메소드를 표현한다.

Fig. 17은 디자인씽킹 마인드맵으로 작성된 도어락 기능의 클래스다이어그램이다.

Table 9는 디자인씽킹 마인드맵으로 표현된 도어락 기능과 소프트웨어공학 클래스다이어그램 매핑요소이다. 마인드맵을 구조화하여 클래스다이어그램으로 표현한다. 각 기능에 들어갈 수 있는 작은 기능을 기술하여 작성한다.

세 번째 매핑 과정이 끝나면, 디자인씽킹의 아이디어를 통해 소프트웨공학의 클래스다이어그램을 명세할 수 있다.

3.4 DT Prototype & SE Design in Detail Design

네 번째 매핑단계는, DT “프로토타입”과 SE “설계”의 상세설계 매핑이다. 상세설계는 순차다이어그램, 상태다이어그램을 표현하고, 스크래치를 활용하여 개발시스템 기능을 작성한다. 매핑 3단계에서 추출된 도어락 시스템 각 기능을 소프트웨어공학의 순차다이어그램으로 작성 후, 스크래치 스크립트를 작성한다. Fig. 18은 매핑 3단계에서 추출된 추상설계의 도어락 시스템 기능별 순차다이어그램이다. 클래스다이어그램으로 표현된 각 기능별 순차 다이어그램을 작성한다.

Fig. 18은 클래스다이어그램으로 표현된 Unlock기능의 순차다이어그램이다. 순차다이어그램으로 각 기능별 시스템

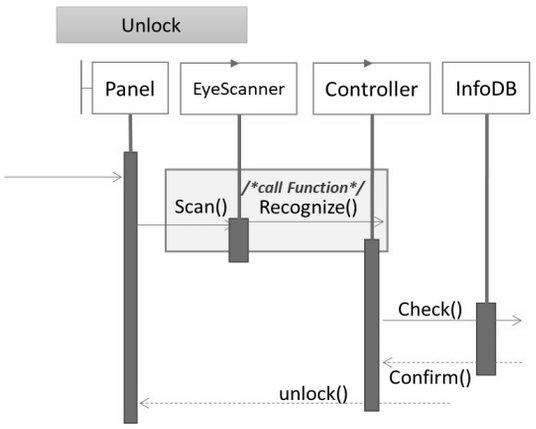
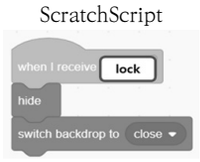
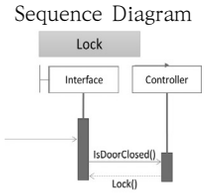


Fig. 18. Sequence Diagram Unlock Function

Table 10. Mapping DT with SE in Step4.

	Design Thinking		Software Engineering	
PROTOTYPE	ScratchScript 		Sequence Diagram 	DESIGN

흐름을 알 수 있다. 순차 다이어그램은 시스템 기능별로 작성한다. Lock/Warn기능 순차다이어그램도 작성한다.

Table 10은 소프트웨어공학의 순차다이어그램으로 도어락 시스템의 기능 및 상태를 스크래치 스크립트로 표현한 것이다. 순차 다이어그램의 흐름도를 스크래치로 표현한다. 네번째 매핑 과정이 끝나면, 디자인씽킹의 프로토타입으로 작성된 스크래치 스크립트를 작성하고, 순차다이어그램을 작성할 수 있다.

3.5 DT Test & SE Implementatin & Test

다섯 번째 매핑단계는, DT “테스트”와 SE “구현”“테스트”의 매핑이다. 매핑 4단계에서 스크래치로 작성된 도어락 시스템 기능 스크립트를 기반으로 코드화 적용하여 스켈레톤 코드로 매핑한다.

Fig. 19는 도어락 시스템 기능을 스크래치로 구현한 스크립트를 간략히 표현한 것이다. 도어락 시스템 주요기능을 보여준다. 도어락 해제는 Unlock기능을 실행하고, Unlock기능으로는 CHECKIRIS, PRESSBUTTON, ICCARD가 있다.

도어락 해제 실패 시, Lock기능을 수행한다. 그리고, 여러번의 불법 시도가 있을 시, 경고 기능이 수행되는 구조이다.

Table 11은 스크래치 스크립트 기반으로 작성된 도어락 시스템 전체 구조의 스켈레톤 코드 템플릿이다. 스크래치로 표현된 도어락 시스템의 각 기능을 템플릿화하여 코드를 생

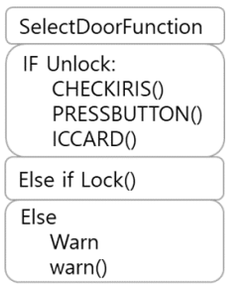


Fig. 19. Scenarios of DoorLock System Prototype based on Scratch Script

Table 11. Skeleton Code Template

```
public class ClassName {

AttributeType Attribute name;
public ClassName() {
   /*object's constructor generation*/
}
public void method-name1() {...}
public void method-name2() {...}
public void method-name3() {...}
```

성한다. ClassName은 구현할 소프트웨어의 이름으로 설정하고, 그 밑에 각 클래스를 생성한다.

디자인씽킹의 프로토타입으로 작성된 스크래치 스크립트와 순차다이어그램을 기반으로 템플릿 코드 작성이 가능하다.

4. 결 론

현재 소프트웨어중심대학에서 학생들에게 창의적 문제 해결기반으로 디자인씽킹을 교육한다. 문제는 창의적 설계는 가능하지만, 코딩과는 전혀 무관한 실정이다. 이 문제를 해결하기 위해 디자인 씽킹 기반의 코딩화를 제안한다. 비전공자들에게 소프트웨어공학개발관점은 매우 어렵다. 본 논문에서는 첫째, 소프트웨어개발프로세스 요구사항에서 구현 코드까지의 수직적 연관성을 제시하고, 둘째, 디자인씽킹 전체공정의 수직적 연관성을 보여준다. 셋째, 두 공정단계의 수평적 매핑을 통해 공통적인 요소사항을 식별한다. 넷째, 공통적인 요소사항이 코드 템플릿 내 위치에 적용한다.

이를 통해 비전공자들은 소프트웨어공학 개념 없이도, 디자인씽킹 메커니즘의 코딩화를 제시한다. 이를 통해 창의적 설계의 코딩화를 기대한다. 현재, 코딩 기반 공정 프로세스를 개발 연구 중이다.

References

[1] T. R. Song and J. H. Lee, “Design thinking to improve problem solving ability,” Seoul: Hanbit Academy, 2019.

[2] Design Thinking [Internet], https://dschool.stanford.edu/resources/getting-started-with-design-thinking.
[3] C. S. Kim, "Easy to learn Software Engingeenring," Seoul: Hanbit Academy, 2020.
[4] C. S. Kim, "Software Engineering Easy to Learn," Seoul: Hanbit academy, 2020.
[5] Karl Wegers, "Software Requirement," Seoul: Wikibooks, 2017.
[6] Y. J. Kwak and J. B. Oh, "UML object-oriented design for beginners," Seoul: Information Culture History, 2006.
[7] B. P. Duglass, "RealTimeUML," Seoul: Acon, 2003.
[8] G. J. Kim and N. G. Cho, "UML Components," Seoul: Intervision, 2000.
[9] H. Y. Yoo, "SoftwareEngineering," 4rd ed., Seoul: SciTech, 1998.
[10] E. M. Choi, "Object Oriented Software Engineering," Hanbit Academy, 2017.
[11] S. Jeong, "Design Thinking Program Development," Chonbuk National University Graduate School of Education, 2015.
[13] J. E. Choi, "Study on program learning outcome setting and evaluation system development of design and software convergence education," Graduate School of Inha University, 2019.
[14] Jeon, S. "A study on Development Maker Education Program for Software Education," Seoul National University of Education Graduate School of Education, 2018.
[15] H. E. Ju and J. S. Lee, "A Study of Open Collaboration Creative Thinking System Based on Design Thinking," *Digital Design Studies*, Vol.12, No.3, pp.179-190, 2012.
[16] C. Y. Seo and R. Y. Kim, "A Study on mapping Software engineering with Design thinking mechanism," *The KIPS Spring Conference 2020(online)*, Vol.27, No.1, pp.349-351, 2020.
[17] H. J. Yoon, "Creative Thinking and Convergence Education for Innovative Design KCI," *A Journal of Brand Design Association of Korea*, Vol.11, No.4, pp.79-92, 2013.
[18] Y. H. Shin, H. J. Jung, and J. S. Song, "Analysis of Learning Experience in Design Thinking-Based Coding Education for SW Non-Major College Students," *Journal of Digital Contents Society*, Vol.20, No.4, pp.759-768, 2019.
[19] K. S. Oh, E. K. Suh, and H. J. Chung, "A study on development of educational contents about combining computational thinking with design thinking," *Journal of Digital Convergence*, Vol.16 No.5, p.65-73, 2018.
[20] H. S. Shin, E. K. Suh, K. S. Oh, H. J. Chung, and S. H. Park, "A Study on the Development of Software Education Program Based on Design Thinking for Nonmajors SW Basic Education," *The Korean Association of Computer Education Conference*, Vol.22, No.2, pp.125-128, 2018.
[21] H. S. Shin, H. J. Jung, and J. S. Song, "Analysis of learning experience in design thinking-based coding education for SW non-major college students," *The Journal of Contents Computing*, Vol. 20, No.4, pp.759-769, 2019.

서 채 연
https://orcid.org/0000-0003-4449-5781
e-mail : chaeyun@hongik.ac.kr
2014년 홍익대학교 전자전산공학(박사)
2019년 ~ 현 재 홍익대학교 소프트웨어융합학과 강사
2020년 ~ 현 재 선문대학교 IT교육학과 강사
관심분야 : 소프트웨어공학기반 디자인 씽킹 접목, 소프트웨어 프로세스, 소프트웨어 가시화, 소프트웨어 코딩 교육, 웹 프로그래밍 연구

김 장 환
https://orcid.org/0000-0003-0185-4406
e-mail : janghwan@selab.hongik.ac.kr
2014년 ~ 2019년 Computer Science at Idaho State University(학사)
2020년 ~ 현 재 홍익대학교 소프트웨어융합학과 석사과정
관심분야 : 소프트웨어 공학, 인공지능, NLP, Software Visualization, Deep Learning, Test Automation

김 영 철
https://orcid.org/0000-0002-2147-5713
e-mail : bob@hongik.ac.kr
2000년 ~ 2001년 LG산전 중앙연구소 Embedded System 부장
2001년 ~ 현 재 홍익대학교 소프트웨어융합학과 교수
관심분야 : MDA기반 메타모델링, 소프트웨어 프로세스 가시화, 정적 분석 연구