Jeong-Jin Kang
Edward J. Rothwell
Jong-Yong Lee
Jonh-Wook Jang

Kang et al.(Eds.)

Advanced and Applied Convergence Letters   **AACL 17**

Advanced and Applied Convergence

Advanced and Applied Convergence

**7th International Joint Conference on Convergence , IJCC 2021
in conjunction with ICAI 2021
Seoul, Republic of Korea ｜ February 18– February 20, 2021
Revised Selected Papers**

IPACT The International Promotion Agency of Culture Technology   **IIBC** The Institute of Internet, Broadcasting and Communication

Volume Editors:

Jeong-Jin Kang
Dong Seoul University, 423 Bokjeong-Dong, Sujeong-Gu, Seongnam, Gyunggi, 13117 Korea
E-mail: jjkang@du.ac.kr

Edward J. Rothwell
Michigan State University, 2120 Engineering Building East Lansing, MI 48824-1226, USA
E-mail: rothwell@egr.msu.edu

Jong-Yong Lee
KwangWoon University, Republic of Korea
E-mail: jyonglee@kw.ac.kr

Jong-Wook Jang
Dong-Eui University, Busan, Republic of Korea
E-mail: jwjang@deu.ac.kr

Please note that the papers in this proceeding book are neither reviewed by peer or professional editor nor accepted as official papers. The papers are working papers that the authors study their research recently.

Adapting Design Thinking to Software Design for solving ill-defined requirement problem on Creative Thinking

ChaeYun Seo and R. Young Chul Kim

SE Lab., Dept. of Software and Communication Engineering, Hongik University,
{chaeyun1, bob2}@hongik.ac.kr

Abstract

Until now, even though software engineers attempt to develop software based on customer requirements with software engineering approaches, they may have an ill-defined or unknown problem of requirements given from the customer. To tackle these problems, we adapt software engineering with design thinking at requirement and design stage because of reframe these in human-centric ways. That is, we are considering more sophistical and creative thinking at software development lifecycle. We will guarantee how to make quality of software based on creative design in 4th Industrial revolution.

Keywords: *Software engineering, Design thinking, Code education, Code education process*

1. INTRODUCTION

In 1967, Herbert A. Simon first mentioned design thinking in the Sciences of the artificial book. In the modern age, this means highly creative process to address human needs, which use design thinking to tackle ill-defined/unknown requirement specification problems. This idea will reframe in better human-centric ways than Ivan Jacobson's Use case approach of Objects Oriented Software Engineering (OOSE). We may consider creative design with Ill-defined/unknown requirement specifications given from customer needs [1]. To achieve this, we propose adapting design thinking to software design on creative thinking. First we refine requirements based on creative thinking, and enhance software design with user needs and behaviors. Second, we can generate sophistical class diagram, sequence diagram, and state diagram with creative information of design thinking.

The remainder of this paper is organized as follows. Section 2 explains the details of related works and the motivation. Section 3 introduces Adapting Design Thinking to Software Process. Section 6, experimental data are summarized and discussed. The last section presents the conclusion of this paper.

2. RELATED WORKS

2.1 SOFTWARE DEVELOPMENT LIFE CYCLE

On general software development process, the software engineer develops his own project based on customer requirements, which is very important how to well define the right requirements for a complete project. But the problems are how to rightly analyze the given natural language based requirements, and how to manage frequently changing requirements. After that, we should describe the definition of natural language based requirements for the customer needs.
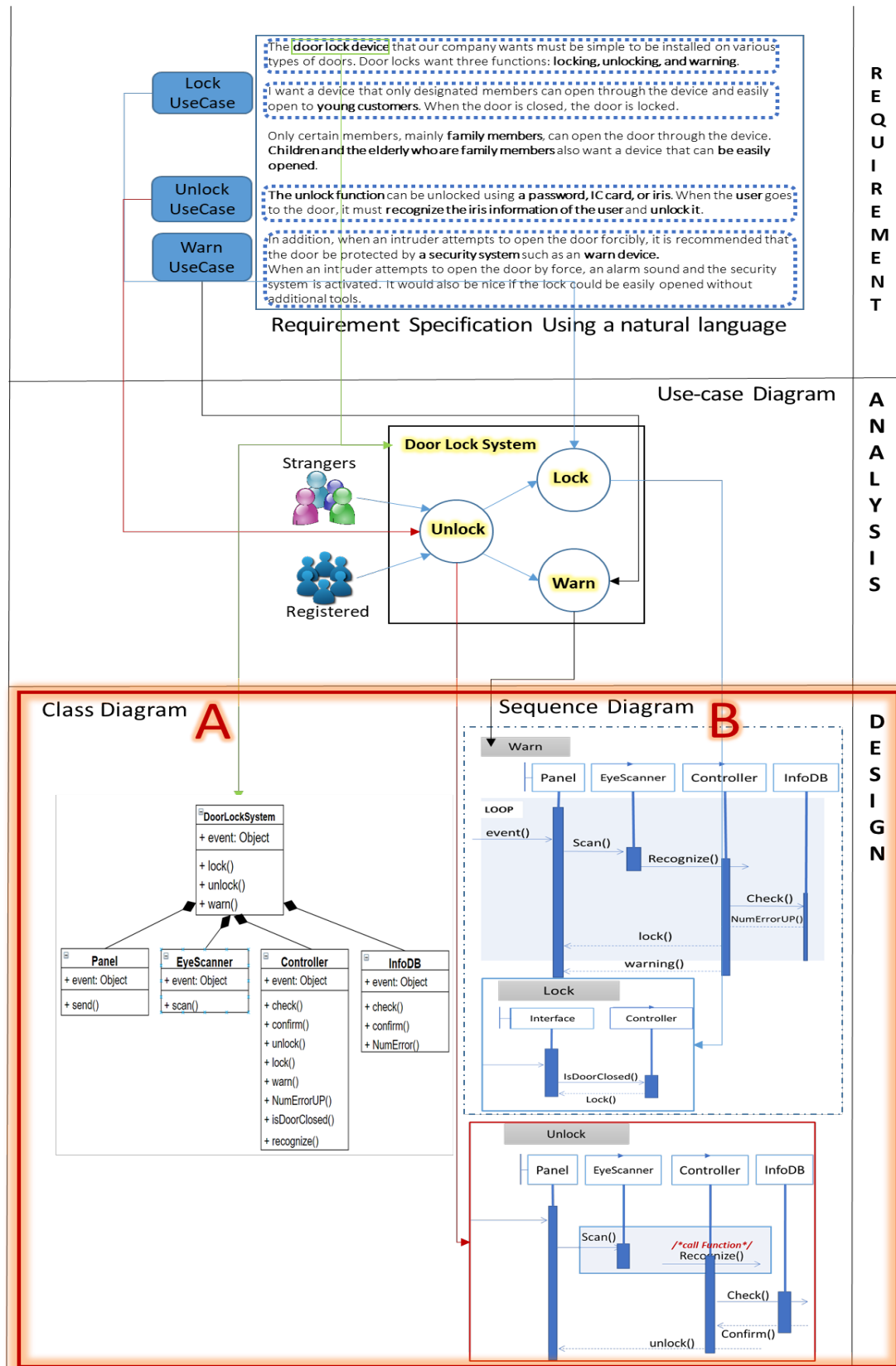
The door lock device that our company wants must be simple to be installed on various types of doors. Door locks want three functions: **locking, unlocking, and warning.**

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.

Requirement Specification Using a natural language

Figure 1. Software Development Process

We use one sample of the smart door locking system to describe our approach in this paper. We use this simple natural language based requirement specifications to develop this locking system. from the viewpoint of software engineering, the software designer typically describe software design such as typical use case, object, and sequence diagram like figure 1. In A and B of figure 1, the software developer designs typical class diagram and sequence diagram without any creative thinking [3].

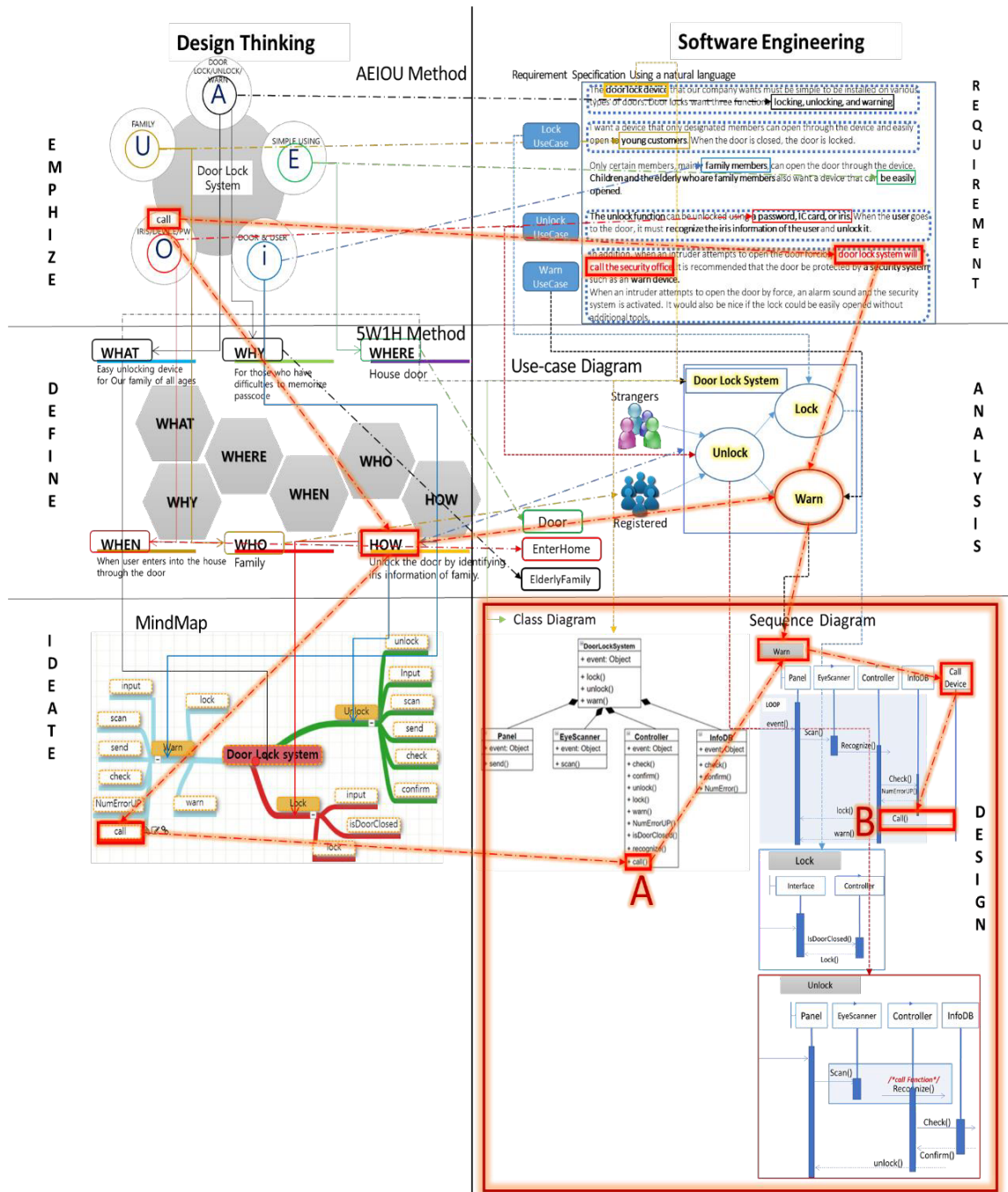# 3. ADAPTING DESIGN THINKING TO SOFTWARE PROCESS

Figure 2. Mapping on Software Engineering and Design Thinking

In 4th industrial revolution, there needs the diverse smart and intelligent software. Therefore, we should

consider how to make more sophistic software design with natural language based requirements. To do this, we suggest adapting design thinking technique to software engineering approach for creative design.

In Figure 2, we have mapped with both processes of Design thinking and software development. For assisting software development with creative thinking, we attempt to extract more information from design thinking approach for finding user-centered system requirements. Based on the extracted information with design thinking, we possibly design more sophistic concepts at design stage of software lifecycle.

In the A part of Figure 2, we can get more detailed information such as *call device and call()* on class diagram. In the B part of Figure 2, we can get more detailed information such as a message calling relationship with *call device* of the *warn* method and a *call()*.

4. CONCLUSION

Until now, even though software engineers attempt to develop software based on customer requirements with software engineering approaches, they may have an ill-defined or unknown problem of requirements given from the customer. To tackle these problems, we adapt software engineering with design thinking at requirement and design stage because of reframe these in human-centric ways. That is, we are considering more sophistical and creative thinking at software development lifecycle. We will guarantee how to make quality of software based on creative design in 4th Industrial revolution. In near future, we are going to develop a development system for majored college students.

ACKNOWLEDGEMENT

REFERENCES

[1] *https://www.interaction-design.org/literature/topics/design-thinking*

[2] *https://dschool.stanford.edu/resources/getting-started-with-design-thinking*

[3] BrucePawelDuglas, RealTimeUML, Acon, 2003

[4] A study on mapping software engineering with design thinking mechanism, chaeyun Seo, R. youngchul Kim, 2020.5

[5] A study on development of educational contents about combining computational thinking with design thinking

[6] A study on the Development of Software Education Program Based on Design Thinking for Nonmajors SW Basic Education

[7] Analysis of learning experience in design thinking-based coding education for SW non-major college students