JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 3, pp. 251-260, March 2020
https://doi.org/10.9708/jksci.2020.25.03.251

# Understanding about Novice Learner's Programming Conception by Prototype Theory

Dong-Man Kim*, Tae-Wuk Lee*

*Student, Dept. of Computer Education, Korea National University of Education, Cheongju, Korea
*Professor, Dept. of Computer Education, Korea National University of Education, Cheongju, Korea

## [Abstract]

In this paper, we propose to understand the conceptual structure of programming elements that learners form during the prototyping theory. To do this, we reviewed previous studies on the meaning of conception and prototype theory and conducted a course of problem-solving programming for 33 university students who had no prior experience in programming, and collected transcription materials through conceptual metaphorical writing. The conclusions of this study are as follows: 1) Identifying the conceptual structure of learners as a conceptual metaphor can enhance the effectiveness of programming education. 2) Learners need to reinforce the experience of forming abstract attributes to form mature programming concepts. 3) The concept of programming differs in the structure of multi-level concepts that students, teachers, and professional programmers have in each group. 4) Programming elements should intentionally block misconception risks in the meaning of symbols. 5) Concept evaluation tools should be developed to check whether various attributes can be applied.

▸ **Key words**: Programming Conception, Variable Conception, Prototype Theory, Conceptual Metaphor, Misconception Risk, Typicality, Prominence

## [요 약]

이 연구의 목적은 확률적 관점의 원형이론으로 학습자들이 프로그래밍 과정에서 형성한 프로그래밍 개념의 구조를 이해하는 것이다. 이를 위해 개념(conception)의 의미와 원형이론(prototype theory)을 고찰하고, 대학생 33명이 문제해결 프로그래밍 경험으로 형성한 학습자의 변수(variable) 개념을 개념적 은유(conceptual metaphor) 글쓰기로 자료를 수집하여 분석하였다. 이 연구의 결론은, 1) 개념적 은유(conceptual metaphor) 글쓰기 활동은 프로그래밍 교육 효과를 높이는 교육 방법이고, 2) 학습자의 추상적 속성 형성 경험을 보강해야 하고, 3) 프로그래밍 개념은 집단별로 수준별 개념(multi-level concept)을 형성하고, 4) 프로그래밍 교육에서 기호의 의미에서 발생 가능한 오개념 위험(misconception risk)을 의도적으로 차단해야하고, 5) 프로그래밍 평가 도구는 다양한 속성을 적용할 수 있는지를 확인할 수 있게 개발되어야 한다.

▸ **주제어**: 프로그래밍 개념, 변수 개념, 원형이론, 개념적 은유, 오개념 위험, 전형성, 현저성

• First Author: Dong-Man Kim, Corresponding Author: Tae-Wuk Lee
*Dong-Man Kim (emotionman@indischool.com), Dept. of Computer Education, Korea National University of Education
*Tae-Wuk Lee (twlee@knue.ac.kr), Dept. of Computer Education, Korea National University of Education
• Received: 2020. 02. 06, Revised: 2020. 03. 23, Accepted: 2020. 03. 23.


http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

## I. Introduction

프로그래밍은 인간이 가진 개념을 컴퓨터가 이해할 수 있는 개념으로 상상하여 작성하는 글쓰기에 해당한다. 이런 개념은 단어에 내재된 의미요소나 사물에 내재한 특성들이 아니라, 인간이 사물과 결합하는 특성으로 파악될 수 있다[1]. 즉 개념에서 기초하고 있는 의미는 언어 내재적 요소의 특성이 드러난 것이 아니라 인간의 고유한 능력이 발현된 인지적인 것이다. 그리고 우리는 개념을 생성하는 인지 과정에서 인지한 세계를 여러 갈래로 나누고, 학습하거나 의사소통할 때 필요한 정보의 양을 감소시킨다[2]. 즉 개념 형성은 인지적 절약(cognitive parsimony)을 달성하는 과정이자 목표이다[2].

인지과학적 심리학자들은 지식이 핵심 개념에 따라 구조화될 수 있고, 그 사이의 관계가 드러나게 배열할 수 있다고 본다[2]. 이러한 방법으로 학습자들이 이해하고 있는 구조를 확인하면 전달하고자 하는 개념을 쉽게 전달할 수 있다[2]. 다시 말해, 학습자들이 경험을 통해 개념을 구체적으로 파악하면, 학습을 위한 지름길을 찾아주는 일이 된다. 그리고 학습자가 이러한 방식으로 지식을 이해하는 것은 더 정교한 이해로 발전하게 하는 교수 방법이 된다[2]. 그리고 학습자는 프로그래밍에서 필요한 정보를 머릿속에서 적절히 조합하고 활용하여 결국 프로그램으로 완성하는 복잡한 인지 과정을 경험한다[3]. 학습자는 프로그래밍 과정에서 자신이 인지하고 있는 프로그래밍에 필요한 다양한 개념에 질서를 부여하고 조직하여 코드로 구체화시킨다. 그래서 효과적인 프로그래밍 교육과정을 설계하기 위해서는 학생들이 프로그래밍 요소에 대한 개념을 어떻게 형성하고 조직하고 있는지 확인할 필요가 있다.

학습자의 개념 형성을 잘 설명하는 이론으로 원형이론(prototype theory)이 있다. 이 이론은 인간이 형성하는 심리학적 개념의 범주는 확률적인 관점에서 원형(prototype)을 중심으로 구성된다고 보고, 학습자들이 형성하는 특정 개념의 구성원들이 원형과 비슷할수록 더 빨리 쉽게 분류하여 개념화된다고 말한다[5]. 그래서 초보 프로그래밍 학습자가 교육을 통해 형성된 개념이 어떻게 구성되고, 어떻게 사용되는가와 관련한 개념의 원형을 알아볼 필요가 있다. 왜냐하면, 교수자 혹은 전문 프로그래머가 형성한 원형(prototype)처럼 학습자의 원형도 비슷하게 구성할 수 있도록 교육과정을 설계한다면, 프로그래밍 교육에서 인지 부하(cognitive load)를 줄이는 방법이 되기 때문이다.

이 연구의 목적은 확률적 관점의 원형이론으로 초보 학습자가 프로그래밍 개념을 어떻게 구성하게 되는지를 알아보는 것이다. 이를 위해 프로그래밍에서 핵심 요소인 변수(variable)를 처음 배운 학습자들이 형성한 개념의 구조를 개념적 은유 방법으로 확인하고, 이를 질적(qualitative) 분석하였다.

이 연구에서 설정한 문제는 '초보 학습자들이 문제해결 프로그래밍 경험으로 형성한 변수 개념의 구조는 어떠한가?'이다.

## Ⅱ. Background

### 1. What is Conception?

개념(conception)은 개인이 갖는 사고의 기본 단위로서 새로운 개념의 형성은 모든 학습 과정에서 토대가 된다[1]. 그래서 개념은 추리, 범주화, 학습, 기억, 연역, 설명, 문제해결, 일반화, 유추, 언어이해, 언어생성 등에서 중요한 역할을 한다[1]. 그래서 교육 분야에서 개념은 인간이 무엇인가를 이해하는 정도를 확인하는 평가의 근간이 되고 있다.

개념의 범주화는 사물이나 상태를 동일화 또는 차별화하여 공통성이나 관계성에 따라 일반화함으로써 통합하는 인지활동을 말하며, 그 심리적 산물을 범주라고 부른다[4]. 그래서 범주화는 패턴 재인의 도구이며, 새로운 대상을 유목화하고 그 대상에 대해 추론하는데 사용할 수 있게 하여 정보를 효율적으로 기억하고 창조적으로 사용할 수 있는 중요한 인지 활동이다[4].

개념(conception)은 개념(concept)과 혼용해서 사용하기도 할 정도로 비슷한 의미를 갖고 있으나, 'conception'은 한 개인에게 국한된 주관적인 개념이고, 'concept'은 사회적 합의나 과학적 근거를 가진 객관적인 개념을 가리키는 영문이다[6]. 일반적으로 'concept'은 개념의 의미를 나타내는 말로써 공인된 속성 집합을 뜻한다[6]. 그래서 학교 교육의 목적은 주관적 개념(conception)이 오개념(misconception)으로 발전하지 않고, 과학적이고 객관적인 개념(concept)으로 발전하게 도와주는 것이다. 이 논문에서는 개인의 이해정도인 주관적 개념의 속성을 파악하기 위한 목적으로 수행되었기 때문에 개념의 의미를 'conception'을 중심으로 사용하고, 특별히 사회적으로 합의된 과학적인 개념을 서술할 때는 객관적 개념이라고 쓰고 'concept'으로 영문 표기하여 서술한다.

### 2. Prototype Theory and Typicality

원형이론(prototype theory)은 개념이 비슷한 속성을 가진 것들의 집합이라는 가족 유사성(family resemblance) 이론에 기초하여, 우리가 형성하는 심리학적 범주는 원형(prototype)을 중심으로 구성된다고 제안하면서, 개념은 기

억 장소에 원형으로 표현된다고 보았다[5]. 그래서 원형은 해당 개념의 범주를 대표할 만한 가장 전형적이고, 이상적이고 좋은 예(example)를 말한다[5]. 즉, 원형은 한 개념의 범주에서 가장 전형적인 세부 특징을 가지고 있다. 그래서 인간은 새로운 예시와 범주의 원형 간 비교를 통해 범주를 판단한다고 본다. 예를 들어, 과일 범주의 구성원인 사과나 오렌지는 무화과보다 더 '전형적이다'라고 설명한다[7]. 이처럼 해당 개념의 범주 속 구성원의 전형적인 성질인 전형성(typicality)과 세부 속성을 많이 공유할 수 있는 구성원이 해당 개념의 원형에 가까울 수 있다고 말한다[7]. 즉, 원형이론은 그 개념과 더 관련된 전형적인 속성에 관심을 둔다. 그리고 원형이론은 개념이 범주를 특징짓는 속성이 본질적으로 명확하기보다는 사례들(examples) 사이의 유사성을 기반으로 확률적인 범주화가 이루어진다고 본다[8]. 그래서 원형이론에서 개념은 거의 모든 속성이 예외를 가지고 있기 때문에 개념의 양호도 효과처럼, 개념을 구성하는 속성이나 특성이 본질적이기보다는 확률적이다[8]. 그래서 개념의 구성원 중에서 원형과 확률적 유사성 정도에 따라 전형성의 등급이 매겨질 수 있다. 그리고 원형이 명확하지 않은 추상적 구성원으로 범주를 이루는 개념은 유연성과 경제성을 높일 수 있는 전형성이 중요한 역할을 한다[8]. 그래서 원형이론에 근거하여 프로그래밍 개념의 구성원 중에 원형에 가까운, 대표성을 갖는, 핵심 속성을 많이 가진 것에 전형성이 높다고 평가할 수 있다. 교육에서 전형성은 수업 시 효과적인 개념 형성을 위해 제시해야 하는 예의 양호도(goodness of examples) 문제와 관련이 있다. 그래서 교사가 특정 개념을 가르칠 때, 전형성이 높은 사례를 숙지하고 적용할 수 있는 역량은 교육적 효과를 가늠할 수 있는 척도가 될 수 있다.

원형은 순간적으로 명확히 떠올릴 수 있는 것이라는 특징을 갖고 있다[8]. 그래서 원형에 가까워 높은 전형성을 가진 속성일수록 머릿속에 떠오르는 표상(representation)이 빠르고 현저하게 나타난다[8]. 그래서 특정 개념을 표상할 때 원형적인 속성은 현저성(prominence)과 상관이 있다.

이와 같이 원형이론은 개인의 개념 형성에서 특정 원형에 대한 전체적인 외적형상의 생성이 구성적 사고에 중요한 역할을 하는 것을 잘 설명하는 대표적인 이론이다[9].

## Ⅲ. Method

### 1. Process

이 연구는 변수 개념의 초기 형성 과정을 알아보기 위해 기초 프로그래밍 강좌 진행, 글쓰기를 통한 자료 수집, 질적 분석 등 크게 3단계 절차로 연구를 진행하였다.

첫째, 프로그래밍 경험이 없는 초보 학습자에게 15주간 교육용 프로그래밍 언어(educational programming language; EPL)를 활용한 문제해결 프로그래밍 강좌를 진행하였다. 구체적인 강의 내용은 Table 1.과 같고, 주당 2시간 연차시로 수업을 진행하였다.

Table 1. Contents and Procedures of the Problem-solved Programming Course

| Week | Subject | Contents |
|---|---|---|
| 1 | Orientation | -Introduction to EPL and Entry<br>-Understanding of intelligent information technology society and computational thinking(CT) |
| 2 | Sequential & Repeated structure | -Understanding of sequential structure and repeated structure<br>-Practice of drawing various shapes |
| 3 | Variable, Operator | -Understanding and practicing of variables and operator<br>-Calculator program development |
| 4 | Select structure, Event | -Understanding of select structure and event<br>-Electronic locker program development |
| 5 | List | -Understanding of list<br>-Understanding the search algorithm and creating a bucket list |
| 6 | Function | -Understanding and practicing of functions and re-factoring |
| 7 | Educational program development | -Mathematics(geometry in point symmetry)<br>-Science(parabolic movement) |
| 8 | Multimedia program development | -Development photo-album and music-player program |
| 9 | Game program development | -Developing Mole-game and Shootout-game |
| 10 | Project workshop | -Problem-solved and programming<br>-Team project Q&A |
| 11 | Problem-solved through SW #1 | -Preliminary step: Problem analysis and design(Modeling) |
| 12 | Problem-solved through SW #2 | -Progress step: programming and testing |
| 13 | Problem-solved through SW #3 | -Final step: debugging and completing tasks |
| 14 | Team project presentation | -Project Presentation and evaluation (Peer Evaluation) |
| 15 | Final exam | -Conception of programming elements |

위와 같은 한 학기 수업에서 변수(variable) 개념의 강의는 공동 연구자들이 사전에 전달할 개념에 대한 속성을 협의하여 명문화한 내용을 토대로 3주차에 진행하였다. 이때 학습자는 명시적인 내용으로 제시된 개념을 최초로 전

달받고 이후 실습 과정과 문제해결 프로그래밍 프로젝트를 진행하면서 변수를 활용하는 경험을 하였다.

둘째, 자료 수집을 위해 종강 시 기말시험과 더불어 변수에 대한 개념적 은유 글쓰기 작업을 진행하였다. 글쓰기는 문서편집프로그램으로 50분간 컴퓨터 실습실에서 작성하여 제출하게 하였고, 작성한 내용이 불실성할 경우 성적에 영향을 줄 수 있음을 작성 전에 공지하여 글쓰기 집중도를 높였다.

마지막 절차로 질적 분석을 실시하고 논문 쓰기를 진행하였다. 이 연구의 핵심 분석 방법은 미리 분석할 코드의 종류를 개발하고 사전 개발된 코드를 기준으로 수집된 자료를 분석하는, 마일스와 휴버만(1984)이 개발한 사전 목록에 의한 코딩 방법을 적용하였다[10]. 그리고 수집한 자료는 전처리 과정을 거친 전사지(transcript)를 완성하여, 2단계의 코딩작업을 진행하였다. 이를 바탕으로 의미를 분석하여 결과를 도출하고 최종 글쓰기를 진행하였다.

## 2. Participants

이 연구는 초보 프로그래밍 학습자가 구성하는 변수 개념의 구조를 파악하고자 광역시에 소재한 G대학교 1학년 학부생을 대상으로 진행하였다. 연구 참여자는 한 학기동안 문제해결과 프로그래밍 과정을 교육용 프로그래밍 언어(EPL)로 프로그래밍을 처음 경험하는 초보 학습자들이다. 연구엔 33명이 참여하였고, 모두 성실히 응답하여 회수된 전체 내용을 분석 자료로 사용하였다. 연구 참여자의 일반적인 현황은 Table 2.와 같다.

Table 2. Gender of Participants

| Gender | Frequency | Percent(%) |
|---|---|---|
| Female | 26 | 78.8 |
| Male | 7 | 21.2 |
| Total | 33 | 100.0 |

## 3. Materials

이 연구에서는 학습자가 형성한 변수 개념의 이해정도를 개념적 은유(conceptual metaphor) 글쓰기 방법을 적용하여 자료를 수집하였다.

은유(metaphor)는 해당 개념의 이해정도를 파악할 수 있는 척도로 사용될 수 있는데, 인지언어학에서는 은유를 인지적 과정으로 보고, 개념적 은유(conceptual metaphor)라는 용어를 사용한다[11]. 인식의 기제로서의 은유를 강조하는 라코프와 존슨(1980)의 개념적 은유 이론(conceptual metaphor theory)을 연구방법으로 적용한 것이 은유 분석이다[12]. 은유 분석(metaphor analysis)은 근원영역(source domain)과 목표영역(target domain) 간의 유사성을 분석하여 은유 표현이 함축하고 있는 의미를 이해함으로써, 그 표현을 창출한 개인의 인지 구조를 파악하는 탐구 방법이다[12]. 최근 국내에서는 은유 분석 방법은 개념적 은유 글쓰기로 자료를 수집하여 교사 교육 분야 및 여러 교과에서 교수자와 학생들의 신념과 태도, 인식 등을 조사하는 연구에 활용되고 있다[12-18]. 그런데 이 연구는 개념적 은유를 통해 의미하는 것을 찾는 것이 아니라 원형이론을 적용하기 위한 해당 개념의 속성을 찾기 위함으로 일반적인 은유 분석 방법 절차를 따르지 않고 개념적 은유 글쓰기를 통한 자료 수집의 방법에 한정하여 적용하였다.

개념적 은유는 목표(target)와 소스(source)를 연결하는 유사성(similarity)은 해당 개념의 속성(attribute)으로 나타낼 수 있기 때문에 개인이 가진 개념을 '{Conception(C)} is {source(S)}. Because of {attribute(A)}.'의 형식으로 표현할 수 있다. 이때 이유(because of)에 작성되는 내용이 특정 개념과 소스의 유사성으로 해당 개념이 가진 속성이 된다. 이 연구에서는 Fig. 1.과 같은 도식처럼 특정 개념이 가진 속성, 즉 개념의 구성원을 확인하였다.

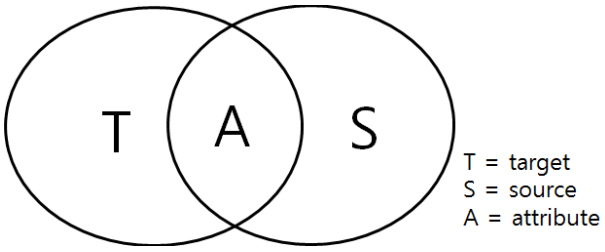


Fig. 1. Diagram of conceptual metaphor

개인이 가진 특정 개념은 그 개념과 유사성을 가진 개념 간 속성의 교집합을 합한 것과 같거나 그 이상이다. 따라서 개인이 형성한 특정 개념은 Fig. 2와 같은 식으로 표현할 수 있다.

$$C_j \geq \sum_{i=1}^{n} A_i \ , \ \ A_i = (C_j \cap S_i)$$

Fig. 2. Mathematical expression of conceptual metaphor

위 식에서 $C_j$는 개인 j가 가진 특정 개념을 말하고, $A_i$는 i번째 소스인 $S_i$가 $C_j$와 유사성을 가진 i번째 속성(attribute)을 말한다.

개념적 은유 분석을 위해서는 연구 참여자들의 언어적 은유 표현 자료를 수집하여야 한다. 그래서 이 연구는 자료 수집을 위해서 문제해결 프로그래밍을 경험한 학습자가 '{변수(variable)} is {대상(source; S)}. Because of {속성(attribute; A)}.'의 형식으로 학습자 스스로 기록지에

작성하게 하였다. 그리고 작성 시 머릿속에 먼저 떠오른 것에 따라 순차적으로 작성하고, 만약 은유한 이유가 글로 작성하기 어려우면 보류하고 나중에 이유 부분을 쓰게 안내하였다. 이런 방식을 채택한 이유는 원형의 전형과 표상 순위의 관계를 분석하고자, 이유(because of)에 해당하는 속성이 먼저 떠올랐지만, 작성의 곤란함으로 순위를 뒤로 미루어 작성될 가능성을 배제하기 위해서였다.

### 4. Analysis Method

수집된 자료는 기본적으로 '원자료(raw data)의 전처리', '코딩', '질적 해석'이라는 일반적인 질적 연구 분석 단계를 거쳤다[10].

먼저, 수집한 원자료에 대한 전처리 작업을 실시하였다. 개념적 은유 작성법을 통해 학생들에게서 얻은 이유(because of)에 해당하는 속성(attribute) 자료를 엑셀 파일로 정리한 후 내용 분석의 편리성을 위해 반복되는 단어인 '이다.', '때문에' 등을 컴퓨터 프로그램을 이용하여 일괄 삭제하는 전처리 과정을 거쳤다. 이와 같은 방법으로 반복 문구를 삭제하고 분석에 필요한 핵심 데이터만 행렬표로 추출하였다.

이어서, 코딩 과정을 진행하였다. 사전 코드를 사용하여 코딩할 경우 연구자는 만들어진 코드에 억지로 자료의 내용에 맞추어서는 안 되며, 새로운 코드를 생성할 수 있는 유연성을 갖추어야 한다[10]. 그래서 이 연구에서는 사전 코드에서 확인되지 않는 내용은 1차적으로 '해석 보류'로 코딩하게 하고, 2차적으로 연구자 3인이 함께 '해석 보류'의 내용을 보고, 패턴을 찾아, 의미를 협의한 새로운 코드를 추가 생성하였다. 그 결과, 연구자들이 미리 개발한 사전 코드(priori code) 12개에 자료 분석하는 과정에서 새로운 의미로 협의가 이뤄진 1개 코드를 추가하여, 최종 13개의 코드를 개발하였다. 이 연구의 분석에 사용된 구체적인 최종 코드 목록은 Table 3.과 같다.

Table 3. Codes of attribute content for concept of variable

| Code | Attribute contents | Time |
|---|---|---|
| A01 | Variable, Changeable | prior |
| A02 | Memorable | prior |
| A03 | Containing, Storable | prior |
| A04 | Just one thing | prior |
| A05 | Having an identifier | prior |
| A06 | Having a data type | prior |
| A07 | Only last left | prior |
| A08 | Movable, Copyable, Deleteable | prior |
| A09 | Controllable at once | prior |
| A10 | Pre-declared | prior |
| A11 | Having directional | posterior |
| A12 | Misconception risk | prior |
| A13 | Uninterpretable | prior |

마지막으로, 질적 해석 과정을 진행하였다. 질적 연구는 연구자의 주관성이 반영될 수밖에 없는데, 이를 최소화하고 신뢰도와 타당성을 높이는 노력이 필요하다[10]. 이 연구에서는 박사급 연구원 2인, 컴퓨터교육과 교수 1인 등의 연구 참여자들과 자료 분석과정을 공유하고 비평하는 작업(peer review)이 이루어졌다[10]. 이 과정은 2단계에 걸쳐 진행되었다. 먼저 1단계로 박사급 연구자 2인이 각자 수집된 자료에 Table 3.의 코드표를 보고 수작업으로 코딩하였다. 이어서 2단계로 1단계에서 각자 코딩한 행렬표를 연구 참여자 3인이 모두 모여 코딩 결과를 하나씩 교차 비교하면서 점검하고 확정하였다. 이때 코딩한 결과가 불일치한 항목은 3인의 의견 합의를 통해 재코딩하고, 의미 해석을 합의하지 못해 코딩하기 어려운 내용은 '해석 불가(uninterpretable)'로 코딩하였다. 그리고 자료의 정제를 위해 한 학습자가 중복하여 같은 개념의 속성을 여러 번 제시한 경우는 먼저 작성한 속성의 코드외에는 모두 삭제하였다. 그리고 코딩된 속성의 학습자가 작성한 순위에 따라 1순위에서 순차적으로 등위를 매기는 방식으로 개인이 가진 속성을 서열화(ranking)하였다. 학습자들이 가진 속성을 표상 순위(representation ranking)로 현저성(prominence)을 양적 분석하기 위해 학습자의 속성 제시 순위를 코딩 결과표에 추가하였다.

질적 해석은 코딩 결과표를 바탕으로 연구자 3인과 함께 원형이론을 바탕으로 의미를 해석하고, '오개념 위험(misconception risk)'으로 분류된 내용은 반복해서 읽고 각자 의미를 제시하고 협의하여, 사고 영역(dominion)을 도출하고 이를 통해 결과를 해석하였다.

이 연구의 자료 처리 및 분석 보조 도구로 Excel과 SPSS 프로그램을 사용하였다.

## IV. Result

### 1. Typicality of novice programming learners about conception of variable

이 연구에서 전체 33명의 학습자가 형성한 변수 개념의 구성원 중에 코드로 확인된 변수 개념의 속성은 78개, 오개념 위험 속성은 18개, 해석 불가한 속성은 82개 등으로 확인되었다. 그리고 코드로 확인된 변수 개념의 전형성(typicality)을 구체적인 빈도로 확인한 결과는 Table 4.와 같다.

Table 4. Typicality of novice programming learners about conception of variable

| Code | Attribute content | Yes | | No | | Total | |
|---|---|---|---|---|---|---|---|
| | | Frequency | Percent(%) | Frequency | Percent(%) | Frequency | Percent(%) |
| A01 | Variable, Changeable | 21 | 63.6 | 12 | 36.4 | 33 | 100 |
| A02 | Memorable | 5 | 15.2 | 28 | 84.8 | 33 | 100 |
| A03 | Containing, Storable | 15 | 45.5 | 18 | 54.5 | 33 | 100 |
| A04 | Just one thing | 5 | 15.2 | 28 | 84.8 | 33 | 100 |
| A05 | Having an identifier | 10 | 30.3 | 23 | 69.7 | 33 | 100 |
| A06 | Having a data type | 2 | 6.1 | 31 | 93.9 | 33 | 100 |
| A07 | Only last left | 3 | 9.1 | 30 | 90.9 | 33 | 100 |
| A08 | Movable, Copyable, Deleteable | 9 | 27.3 | 24 | 72.7 | 33 | 100 |
| A09 | Controllable at once | 2 | 6.1 | 31 | 93.9 | 33 | 100 |
| A10 | Pre-declared | 2 | 6.1 | 31 | 93.9 | 33 | 100 |
| A11 | Having directional | 4 | 12.1 | 29 | 87.9 | 33 | 100 |

확인 결과, 연구자들이 설정한 변수 개념의 속성 코드를 31명이 형성하고 있었고, 그중 29명은 교수자가 사전에 전달하고자 의도한 속성을 인지하고 있었다. 그리고 33명 중 2명은 변수 개념의 속성을 너무 일반적인 것으로 제시하거나 자칫 오개념을 발생시킬 위험 인자(risk factor)만을 갖고 있었다.

초보 프로그래밍 학습자들이 형성한 개념의 구성원은 '다양한, 변하는'의 속성이 21명(63.6%)으로 가장 많았고, '저장하는, 담는'은 15명(45.5%), '식별자(이름)를 갖는'은 10명(30.3%), '이동·복사·삭제가 가능한'은 9명(27.3%), '기억(기록)하는'과 '한 가지만'은 각각 5명(15.2%)씩, '방향성을 갖는'은 4명(12.1%), '마지막만 남는'은 3명(9.1%), '일괄 제어하는', '형식을 갖는', '미리 선언하는' 등은 각각 2명(6.06%) 등의 순으로 빈도가 높게 확인되었다.

초보 프로그래밍 학습자들이 변수 개념 형성에서 '다양한, 변하는'을 가장 대표적인 속성으로 표현하였으나 21명(63.6%)만이 제시하였고, '저장하는, 담는'의 속성도 15명(45.5%)으로 절반을 넘지 못하는 수준이었다.

이어서 학습자의 학습 상태가 비교적 성실하여 유효한 변수 개념을 형성한 학생 31명을 대상으로 각 속성이 전체에서 얼마나 차지하는지를 확인하였다. 특정 변수 속성을 표상할 수 있는 학생들의 비율을 확인한 구체적인 결과는 Table 5.와 같다.

Table 5. Ratio of programming novice about typicality of variable

| Code | Attribute content | Frequency | Percent(%) | Percent of Cases(%) |
|---|---|---|---|---|
| A01 | Variable, Changeable | 21 | 26.92 | 67.74 |
| A03 | Containing, Storable | 15 | 19.23 | 48.39 |
| A05 | Having an identifier | 10 | 12.82 | 32.26 |
| A08 | Movable, Copyable, Deleteable | 9 | 11.54 | 29.03 |
| A02 | Memorable | 5 | 6.41 | 16.13 |
| A04 | Just one thing | 5 | 6.41 | 16.13 |
| A11 | Having directional | 4 | 5.13 | 12.90 |
| A07 | Only last left | 3 | 3.85 | 9.68 |
| A10 | Pre-declared | 2 | 2.56 | 6.45 |
| A06 | Having a data type | 2 | 2.56 | 6.45 |
| A09 | Controllable at once | 2 | 2.56 | 6.45 |
| Total | | 78 | 100 | 251.61 |

'다양한, 변하는'을 개념의 속성으로 제시한 학생은 21명으로 67.74%가 갖고 있고, '저장하는, 담는'은 15명으로 48.39%, '식별자를 갖는'은 10명으로 32.26%, '이동·복사·삭제가 가능한'은 9명으로 29.03%, '기억하는'과 '한 가지만'은 5명으로 16.13%, '방향성을 갖는'은 4명으로 12.90%, '마지막만 남는'은 3명으로 9.68%, '미리 선언하는', '형식을 갖는', '일괄 제어하는' 등은 각각 2명으로 6.46%, 이와 같은 순으로 해당 속성의 형성 정도 차이가 발생하는 것을 확인하였다.

이상의 결과를 통해 확인한 초보 프로그래밍 학습자의 변수 개념의 전형성(typicality)은 '다양한, 변하는', '저장하는, 담는', '식별자(이름)를 갖는', '이동·복사·삭제가 가능한' 등은 상대적으로 높았고, '방향성을 갖는', '마지막만 남는', '일괄 제어하는', '형식을 갖는', '미리 선언하는' 등은 비교적 낮은 것으로 확인되었다.

## 2. Prominence of the Variable Conception

변수 속성의 현저성(prominence)을 알아보기 위해 다중응답 분석의 기술통계분석으로 속성의 표상 순위(representation ranking)를 확인하였다. 그 구체적인 결과는 Table 6.과 같다.

초보 프로그래밍 학습자들이 구성하고 있는 변수 개념에서 '다양한, 변하는'의 속성은 표상 빈도는 가장 높았으나 '저장하는, 담는' 속성의 표상 순위가 더 빨라 현저성이 더 높은 것으로 확인되었다. 그리고 '이동·복사·삭제가 가능한'과 '식별자를 갖는' 등의 속성도 표상 빈도는 각각 4위와 3위로 나타났으나 표상 순위는 각각 6위와 9위로 전형성에 비해 상대적으로 현저성이 낮은 속성으로 파악되었다. 그래서 변수 개념의 현저성(prominence)을 확인한 결과 초보 프로그래밍 학습자들은 형성한 변수 개념의 구조는 속성별로 더 중요하다고 판단한 주관적 가중치(weight)를 부여하고 있었다.

Table 6. Representation Ranking of the Variable Attribute

| Ranking | Code | Attribute content | N | Sum | Mean | S.D. |
|---|---|---|---|---|---|---|
| 1 | A03 | Containing, Storable | 15 | 25 | 1.67 | 1.1127 |
| 2 | A01 | Variable, Changeable | 21 | 36 | 1.71 | .8451 |
| 3 | A07 | Only last left | 4 | 7 | 1.75 | 1.5000 |
| 4 | A11 | Having directional | 4 | 7 | 1.75 | 1.5000 |
| 5 | A02 | Memorable | 5 | 10 | 2.00 | 1.0000 |
| 6 | A08 | Movable, Copyable, Deleteable | 8 | 16 | 2.00 | .7559 |
| 7 | A09 | Controllable at once | 2 | 4 | 2.00 | 1.4142 |
| 8 | A04 | Just one thing | 5 | 12 | 2.40 | .8944 |
| 9 | A05 | Having an identifier | 10 | 28 | 2.80 | 1.3165 |
| 10 | A10 | Pre-declared | 2 | 7 | 3.50 | 2.1213 |
| 11 | A06 | Having a data type | 2 | 9 | 4.50 | .7071 |

그리고 학습자들이 프로그래밍 경험으로 갖게 된 변수의 개념이 일상적인 의미에서는 '저장하는'과 '기억하는'은 변수 개념의 속성이 연관성이 존재하여 인접하게 표상되어야 할 것으로 연구자들이 확인하고 사전에 합의하여 구성하였으나, 실제 학습자들이 구성한 변수의 개념은 연관성을 갖고 인접한 순위로 표상되지 않고 연구자들이 구성한 개념과 달리 분절되어 있었다. 즉 연구자들이 합의한 변수 개념(concept)의 전형성과 현저성은 학습자가 문제해결 프로그래밍 과정으로 구성한 것과 달랐고, 속성의 일반적인 기호적 의미와 상관되어 인접성의 원리로 표상되는 환유(metonymy) 작용이 발생하지 않았다.

### 3. Misconception Risk of Variable

변수 개념화에서 오개념(misconception)을 형성시킬 수 있는 위험 속성을 가진 학습자는 33명 중 13명이었고, 이들이 작성한 속성들 중 중복되는 내용은 제거하여 오개념 형성을 유발할 인자(factor) 18가지 속성을 파악하였다. 이것을 비슷한 의미를 갖는 집단으로 구분하고, 해당 사고 영역(dominion)의 의미를 해석하였다. 이를 통해 총 3가지 의미가 결과로 도출되었다.

첫 번째는 변수가 단순히 알 수 없고, 통제 불가능한 것으로 개념화하였다.

*"예상치 못하기", "미지수처럼 모르는 값", "어디로 튈지 모르는 결과를 만들기", "어떤 일이 일어날지 모르기", "어떤 수가 나올지는 모르지만", "정해지지 않은 다른 것이 도출"*

위와 같이 학습자들은 변수를 '변하는 것'의 의미를 자칫 무작위(random)로 오인하고 있었다. 그래서 변수를 제어하지 못하는 프로그래밍 요소로 개념화 할 수 있는 위험이 존재하였다.

두 번째, 학습자들이 가진 변수와 관련된 데이터 형식을 수(number)로 한정하여 개념화하였다.

*"계산식을 만들 때 필요한", "정해진 범위가 없고 수가 무한하게 펼쳐지기", "끝이 없는 수이기", "수가 반드시 나오기", "미지수로 수식을 만드는"*

위와 같이 변수를 수(number)에 더 의미를 두고 개념화되어 있는 경향이 존재하고 있다. 그리고 특히 *"무한하게", "끝이 없는"* 처럼 변수를 무한수의 개념으로 인식한 경우도 존재하였다. 이는 기존에 갖고 있던 '수학적 경험의 변수' 개념과 간섭되는 현상으로 판단되었다.

세 번째, 학습자들이 변수를 함수가 갖는 개념 속성과 중복되거나 인접하게 개념화하여 변수와 함수를 명확히 구분하지 못하였다.

*"미지수처럼 모르는 값으로도 계산식을 만들 수", "입력한 데이터를 '변수'로 포장할 수 있기", "값이 일정하게 변하기", "값을 입력하면 다양한 코드를 실행하거나 답변을 얻을 수 있기", "input을 넣어야만 output으로 나올 수 있기", "어떠한 것을 아무렇게나 입력하여도 결과가 나오기", "입력한 것에 따라 도출되는 것이 달라지기", "값만큼 결과가 나오기", "값에 따라 최종 값이 달라지기"*

변수의 개념이 함수 개념의 구성원과 중첩되거나 구분이 모호한 속성으로 구성되어 있어서 변수와 함수, 모두에서 오개념 형성 가능성이 있었다. 그래서 변수와 함수 개념 형성 시 연계성을 고려하고 서로 공통점과 차이점을 인지하고 명확히 구분할 수 있게 세심한 지도가 필요함을 확인하였다.

## V. Discussion

이 연구의 결과에서 초보 프로그래밍 학습자들은 변수 개념의 전형성(typicality)과 표상 순위인 현저성(prominence)의 차이가 발생하였다. 이런 전형성과 표상 순위의 불일치는 학습자가 속성의 중요성을 주관적으로 판단하고 속성의 가중치(weight)를 적용하여 특정 속성을 현저하게 표상하는 구조로 개념을 형성한다는 증거이다. 다시 말해, 우리가 변수라고 하는 것을 떠올릴 때 변수의 전형과 가까운 속성을 떠올리기 보다는 최근 경험에서 자신이 의미를 더욱 부여한 속성을 먼저 떠올리는 것이다. 따라서 개인의 개념화 정도를 양적으로 확인하려면, 전술

한 개념적 은유 방법의 표현식인 Fig. 2에 속성의 전형성과 현저성에 따른 가중치를 모두 고려하게 수정해야 한다. 그리고 전형성을 가진 속성의 개념화 여부와 가중치의 차이는 개인 경험의 차이를 구분할 수 있는 방법, 즉 초보자와 전문가의 차이를 구분할 수 있는 근거가 될 수 있다.

결국, 이 연구의 결과로 프로그래밍 학습을 통해 형성한 개인의 주관적 개념(conception)은 다양한 속성의 집합이며, 각 속성은 전형성 값과 주관적 가중치 값을 가진 선형함수로 만들 수 있기 때문에, 아래의 Fig. 3과 같은 식으로 표현할 수 있다.

$$C_j = \sum_{i=1}^{n} W_i T_{ih}$$

Fig. 3. Linear function of conception by conceptual metaphor

위 식에서 $C_j$는 개인 j가 가진 인지한 개념(conception)이고, $W_i$는 i번째 속성의 현저성(prominence)으로 확인할 수 있는 속성에 대한 가중치(weight) 값이고, $T_{ih}$는 h집단의 빈도로 확인할 수 있는 i번째 속성의 전형성(typicality) 값이다. 그리고 현저성을 단순한 순위(ranking)로 확인하기보다는 객관적인 평점(ratio)으로 매기는 작업은 보다 정확한 가중치 값을 확인할 수 있을 것이다.

이처럼 개념적 은유를 통해 전문개발자나 교수들이 가진 주관적 개념(conception)의 가중치와 전형성 값을 확인하여 이를 통계적으로 종합하면, 교육을 통해 가르칠 객관적 개념(concept)을 양적으로 확인할 수 있다. 그리고 학생의 개념화 정도는 같은 방법으로 해당 학생의 주관적 개념(conception) 정도를 확인하고 합의된 객관적 개념(concept)과 양적으로 상대 비교하여 명확히 파악할 수 있다. 즉, 개인이 형성한 프로그래밍 요소에 대한 개념이 초급, 중급, 고급의 학습자 집단이나 교수자, 전문개발자 등의 전문가 집단 등 여러 집단의 특성을 수치화한 결과와 비교하여 어느 집단과 유사한지를 확인함으로서 학습자 수준을 객관적으로 평가할 수 있는 방법이 될 수 있다. 그리고 이런 방식의 평가는 학습자와 전문가 집단 간 전형의 비교를 통해 학습 결핍이나 보완할 속성을 확인하여 완숙한 개념 형성에 필요한 학습을 추천할 수 있다. 그리고 나아가 개인, 학습자 집단, 전문가 집단 등이 가진 수치화된 개념 특성을 기계학습(machine learning)시켜 분류하고 개념의 형성 정도를 평가하고 부족한 속성을 형성시키기 위한 학습 추천 시스템(learning recommendation system)을 개발할 수도 있다.

## Ⅵ. Conclusions

이 연구는 개인의 변수 개념(conception)에 대한 질적 사례연구로 학습자의 개념 구조를 하나의 사례로 심층적으로 분석하고 해석하여, 초보 프로그래밍 학습자의 개념화에 대한 총체적인 이해를 시도하였다. 그래서 이 연구로 개념적 은유 글쓰기 방법이 개인의 개념 구조를 확인할 수 있는 방법임을 확인하여 이를 선형함수(linear function)로 제시하였고, 학습자의 사고를 양적으로 확인할 수 있으면서도 질적 연구처럼 인간 사고의 내면을 파악할 수 있는 혼합 연구 방법이 될 수 있음을 확인하였다. 그리고 수집한 자료를 통해 초보 학습자가 변수 개념이 어떠한지를 전형성(typicality)과 표상 순위(representation ranking), 현저성(prominence), 오개념 위험(misconception risk) 등으로 확인하였다. 그래서 이와 같은 결과를 바탕으로 내린 결론은 아래와 같다.

첫째, 개념적 은유(conceptual metaphor) 글쓰기는 학습자의 개념 구조 확인은 프로그래밍 교육 효과를 증진하는 교육 활동이 된다. 개념적 은유는 학습자 자신의 개념 구조를 메타 인지(meta cognitive)하는 활동이며, 개념의 구성원을 원형에 가깝게 조직하는 데 도움이 된다. 그리고 수업 시 교수자는 이런 활동으로 개념의 속성과 전형성의 파악으로 학습자가 가진 오개념(misconception)을 정확히 진단할 수 있는 근거가 되고, 유사성(similarity)의 원리로 학습자의 학습 과정과 결과의 추론이 가능한 정보를 제공한다. 또한 수업 시 예의 양호도(goodness of examples) 문제를 충족시켜줄 수 있다.

둘째, 학습자가 완숙한 프로그래밍 개념을 형성하기 위해서는 추상적 특징을 가진 속성의 형성 경험을 보강해야 한다. 학습자들이 형성한 변수 개념에서 전형성은 '다양한, 변하는'의 추상적 특징을 가진 속성이 21명(63.6%), '저장하는, 담는' 의 물리적 특징을 가진 속성이 15명(45.5%) 등의 순으로 나타났으나, 현저성의 순위는 이와는 반대로 나타났다. 즉, 초보 학습자는 변수 개념을 교수자나 초보 학습자 집단에서 전형적인 것으로 파악한 추상적 특성을 가진 속성보다 물리적 특성을 가진 속성을 보다 중요한 속성(important attribute)으로 인지하고 있었다. 그래서 학습자가 완숙한 프로그래밍 개념을 형성하기 위해서는 의도적으로 추상적 특징을 갖는 속성에 대한 주관적 가중치(subjective weight)를 더 부여할 수 있도록 교육 프로그램을 구성해야겠다.

셋째, 프로그래밍 개념은 학생, 교사, 전문가 등 집단별로 갖게 되는 수준별 개념(multi-level concept)의 구조

로 다르게 형성된다. 연구자들이 전달하고자 한 개념의 속성은 A01 ~ A11 순으로 순차적으로 합의하고 수업 시에도 제시하였으나, 학습자들은 연구자들이 알려준 속성 외에도 다른 속성을 1개 더 구성하는 경향을 보였으며, A01, A03, A05, A08, A02, A04, A11, A07, A10, A06, A09 등의 순으로 나타났다. 이처럼 연구자들이 전달하고자 한 합의된 객관적 개념(concept)의 일부를 학습자들은 인지하여 주관적 개념(conception)의 속성으로 구성하고, 여기에 추가로 새롭고 유의미한 속성을 학습자 집단에서 형성하고 있었다. 즉, 교수자와 학습자는 각각의 집단이 갖고 있는 개념 구성원의 전형성과 현저성으로 인한 가중치가 다르기 때문에 수준별로 개념의 구조를 다름을 확인하였다. 따라서 프로그래밍 교육과정을 학생의 주관적 개념(conception)을 전문가들이 가진 객관적 개념(concept)으로 발전시키도록 학습의 계열과 지식의 구조를 설정하려면, 반드시 수준별 개념 구조의 확인이 필요하다.

넷째, 프로그래밍 요소의 개념화는 기호의 의미에서 발생 가능한 오개념 위험(misconception risk)을 차단하는 노력이 필요하다. 일반적으로 해당 구성요소를 지칭하는 용어가 만들어질 때 기호적 의미가 가장 많이 반영된 것이 전형성 높은 속성인 경우가 많은데, 이것이 오인되어 오개념을 형성하는 원인이 될 수 있다. 변수 개념도 마찬가지로 학습자들이 변수를 '무작위하고 통제 불가능한 수'로 개념을 형성한 경우가 많았다. 그래서 학습자들의 오개념 형성을 막기 위해서는 변수 개념을 전달할 때는 '수(number)'로 한정하지 않도록 주의해야하고, '변하는' 것이 '무작위'나 '통제 불가능' 하지 않고, '다양성'을 의미하는 것으로 개념화할 수 있는 세심한 지도가 필요하겠다.

다섯째, 프로그래밍 개념의 평가 도구는 해당 개념의 다양한 속성을 적용할 수 있는지를 확인 할 수 있게 개발되어야 한다. 변수의 개념은 일반화된 정의처럼 명시적이지 않고 맥락에서 형성된 다양한 속성의 덩어리임을 확인하였다. 그리고 특정 프로그래밍 요소를 학습자가 알고 있다는 것은 프로그래밍 특성 상 그 요소의 속성을 맥락에 맞게 정확히 사용하는가의 여부에 따라 판단해야 한다. 그래서 프로그래밍 개념의 평가 도구의 구상은 그 개념의 특정 속성을 적용할 수 있는 다양한 상황을 지정하는 노력이 수반되어야 한다.

이 연구는 초보 학습자의 프로그래밍 개념 학습 경향성을 파악하기 위한 과제이나, 학습자의 선행 개념의 차이나 학습 성취도에 영향을 주는 요인들의 배제를 하지 못하고 조사 대상이 33인으로 연구 결과의 해석과정에서 통계적인 오류를 수반할 가능성이 존재하여, 타 학습 및 평가 모델과 비교 연구하기에는 한계가 있다. 그래서 이 연구를 보완하고 연구 결과를 발전시키기 위해서, 추가 학습자의 데이터를 확보하여 프로그래밍 교육에서의 다양한 학습 및 평가 모델과 비교하는 후속 연구와 학습자의 개념 변화 과정을 이해하고, 효과적인 교육과정을 운영하기 위해 초기 학습 상태와 학습 성취도, 학습 태도 등이 개념 형성에 어떤 영향을 주는지에 대한 양적 및 질적 후속 연구를 진행하고자 한다. 또한 초급, 중급, 고급 등의 각 수준의 학습자, 그리고 교수자, 전문개발자 등 다양한 집단에서 갖고 있는 프로그래밍 요소에 대한 개념의 구체적인 속성을 파악하여 학습자의 인지적 부하를 줄여줄 수 있는 연구를 진행하고자 한다.

## REFERENCES


[1] Robert M. Gagne, "Mastery Learning and Instructional Design," International Society for Performance Improvement. Vol. 1, No. 1, pp. 7-18, Jun. 1988.

[2] Kyungsu Wang, "An Alternative View of Concept Learning and its Educational Implications," The Journal of Curriculum and Evaluation, Vol. 11, No. 1, pp. 97-118, Apr. 2008.

[3] Dong-Man Kim, Tae-Wuk Lee, "Understanding of programming thinking from Semiotics Perspective," Proceedings of the Korean Society of Computer Information Conference, Vol. 28, No. 1, pp. 275-276, Jan. 2020.

[4] Chung-sil Kim, "Application of the prototype theory in teaching vocabulary in Korean," International Network for Korean Language and Culture. Vol. 10, No. 2, pp. 31-48, Dec. 2013.

[5] Ji-ryong Lim, "Prototype Theory and Semantic Categorization," Journal of Korean Linguistics, Vol. 23, No. 0, pp. 41-68, Dec. 1993.

[6] Minsung Kim, "High School Students' Geographic Misconceptions Recognized by Teachers," Journal of the Korean Geographical Society, Vol. 48, No. 3, pp. 482-496, Jun. 2013.

[7] Eleanor Rosch, "Cognitive representations of semantic categories," Journal of Experimental Psychology General. Vol. 104, No. 3, pp. 192-233, Sep. 1975.

[8] Young Su Kwon, "Die Rolle der Typikalitat und Prototypikalitat in der Prototypentheorie," Deutsche sprach-und literaturwissensc haft, Vol. 16, No. -, pp. 143-160, Nov. 2001.

[9] Peter Gärdenfors, "Conceptual Spaces: The Geometry of Thought," : The MIT Press. Mar. 2004.

[10] Young Cheon Kim, "Qualitative research methodology 1:Bricoleur," Academypress. Aug. 2016.

[11] Ji-ryong Lim, "On Conceptual Metaphor," The Society Of Korean Semantics, Vol. 20, No. -, pp. 29-60, Aug. 2006.

[12] Seulki Lee, "Middle and High School Students' Awareness of

Writing Using Text Mining—Focus on Metaphor Analysis PDF icon," korean language education research, Vol. 53, No. 2, pp. 141-178, Aug. 2018.

[13] Kyungnam Rho, Taekkoo Kang, "Childcare Teacher's Perception of Multicultural Education Examined Through Metaphorical Analysis," The Journal of Humanities and Social science (HSS21), Vol. 10, No. 4, pp. 1461-1476, Aug. 2019.

[14] Ja-kyoung Kim, Sung-Ouk Chang, Jung-Won Shin, "Metaphor Analysis for Pre-service school Teachers' Perceptions of Inclusive Education," Korean Journal of Physical, Multiple & Health Disabilities, Vol. 62, No. 1, pp. 73-100, Jan. 2019.

[15] Chunmei Lin, "Application of Cognitive Linguistics Theory in the Semantic Teaching of Collocations-Centering on Conceptual Metaphor and Image Schema," Journal of Korean Language Education, Vol. 30, No. 2, pp. 311-332, May. 2019.

[16] Haejeon Kim, "Pre-service teacher's recognition and perspective on young children of multicultural family through metaphor analysis," The Society for Constructivist Early Childhood Education, Vol. 6, No. 1, pp. 99-121, Aug. 2019.

[17] Rim Lee, "Metaphor Analysis of Textbooks: Focused on Students Participating in the Teaching Class," Journal of Learner-Centered Curriculum and Instruction, Vol. 19, No. 2, pp. 223-247, Jan. 2019.

[18] Mi-Jin Yi, "Analysis on metaphor for cooperative learning recognized by pre-service early childhood teachers," The Journal of Korea Open Association for Early Childhood Education, Vol. 24, No. 3, pp. 121-144, Jun. 2019.

## Authors

Dong-Man Kim received the B.Ed. degree in Computer Education from Daegu National University of Education, Korea in 2002. He received the M.Ed. degree in Practical Arts Education from Gyeongin National University of Education, Korea in 2015. Mr. Kim is currently a doctoral course student in the Department of Computer Education, Korea National University of Education. He is interested in software education, maker education, and data mining.

Tae-Wuk Lee received the B.S. degree in Science Education from Seoul National University, Korea, in 1978. And he received the M.S. and Ph.D. degrees in Computer Science and Education from Florida Institute of Technology, U.S.A. in 1982 and 1985, respectively. Dr. Lee joined the Department of Computer Education at Korea National University of Education, Cheongju, Korea, since 1985. He is interested in computer education and knowledge engineering.