JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 11, pp. 277-286, November 2020
https://doi.org/10.9708/jksci.2020.25.11.277

A Study on Programming Concepts of Programming Education Experts through Delphi and Conceptual Metaphor Analysis

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 a new educational approach to help learners form concepts by identifying the properties of programming concepts targeting a group of experts in programming education. Therefore, we confirmed the typical properties of concepts by programming education experts for programming learning elements through conceptual metaphor analysis, which is a qualitative research method, and confirmed the validity through the delphi method. As a result of this study, we identified 17 typical properties of programming concepts that learners should form in programming education. The conclusions of this study are that need to compose the educational content more specifically for the conceptualization of learners' programming as follows: 1)the concept of a variable is to understand how to store data, how to set a name, what an address has, how to change a value, various types of variables, and the meaning of the size of a variable, 2)the concept of operator is to understand how to operate the four rules, how to deal with it logically, how to connect according to priority, meaning of operation symbols, and how to compare, 3)the concept of the control structure is to understand how to control the execution flow, how to make a logical judgment, how to set an execution rule, meaning of sequential execution, and how to repeat executing.

▸ **Key words**: Programming Concepts, Conceptual Metaphor Analysis, Delphi, Typical Properties, Programming Education

[요 약]

이 연구의 목적은 전문가 집단을 대상으로 프로그래밍 교육에 필요한 프로그래밍 개념의 구성원을 도출하여 학습자의 개념화를 돕는데 있다. 이를 위해 변수, 연산자, 제어 구조 등의 프로그래밍 요소를 질적 연구 방법인 개념적 메타포 분석 방법을 적용하여 프로그래밍 교육 전문가 집단이 형성한 개념의 전형적 속성을 확인하고 델파이 연구 방법을 통해 타당성을 확보하였다. 이 연구의 결과로 프로그래밍 개념의 전형적 속성 17가지를 도출하였다. 이를 바탕으로 내린 결론은 프로그래밍 교육에서 학습자가 1)변수 개념은 데이터를 저장하는 방법, 이름을 정하는 방법, 변수의 주소가 갖는 의미, 값을 변하게 하는 방법, 변수의 다양한 형태, 변수가 갖는 크기의 의미 등, 2)연산자 개념은 사칙 연산 방법, 논리적인 처리 방법, 우선순위에 따른 연결 방법, 연산 기호의 의미, 비교하는 방법 등, 3)제어 구조 개념은 실행 흐름 제어 방법, 참/거짓으로 논리적 판단 방법, 실행 규칙 설정 방법, 선택적 실행 방법, 순차적 실행의 의미, 반복적 실행 방법 등을 이해하도록 교육 내용을 보다 구체적으로 설정해야 한다는 것이다.

▸ **주제어**: 프로그래밍 개념, 개념적 메타포 분석, 델파이, 전형적 속성, 프로그래밍 교육

• 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. 09. 21, Revised: 2020. 11. 09, Accepted: 2020. 11. 14.
• This paper is an extension of the paper presented, Understanding of programming thinking from Semiotics Perspective, at the Korea Society of Computer and Information Winter Conference 2020, Daejeon, Korea.

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

I. Introduction

프로그래밍은 인간의 문제해결 아이디어를 구체적 코드로 완성하는 과정이다. 그래서 학습자는 자신의 추상적 생각을 구체적인 코드를 이용해 논리적 조합으로 프로그램을 완성한다. 문제해결의 아이디어에서 발생한 추상적 요소를 특정 프로그래밍 요소와 연결하기 위해서는 그 해당 프로그래밍 요소가 갖고 있는 특성과 논리적으로 연결 가능한 프로그래밍 요소의 특성까지도 파악하고 있어야 한다. 그래서 프로그래밍에 대한 개념은 사용되는 맥락과 사용하는 방법에 대한 이해까지 포함한 것으로 프로그래밍 능력의 핵심적 역할을 한다.

프로그래밍 학습자는 다양한 인지적 경험을 통해 전문가로 성장하면서 프로그래밍 개념을 형성한다. 정란(1996), 플뢰리(2000), 에커달 외(2005), 로사민 외(2012) 등은 초보자가 경험하는 실수나 오개념(misconception)의 형성은 해당 개념의 형성 여부가 아니라 개념화 정도가 부족하여 발생하는 경우가 많다고 주장하였다[1-4]. 다시 말해, 프로그래밍 초보자가 전문가로 성장하기 위해서는 프로그래밍에 필요한 개념이 충분한 영역에서 깊이 있는 이해가 필요하다. 그래서 프로그래밍 교육에서는 학습자가 각 프로그래밍 요소가 갖고 있는 속성을 전체적으로 이해하여 개념을 형성하도록 도와야 한다. 만약 프로그래밍 학습으로 형성할 개념의 구체적인 결과를 학습자가 이해할 수 있는 방법으로 제시할 수 있다면, 보다 효과적이고 효율적인 교육 전략을 설계할 수 있다. 그래서 학습의 목표와 같은 전문가가 형성한 프로그래밍 개념은 어떠한지, 해당 개념의 실체를 구체적으로 확인할 필요가 있다. 그러나 컴퓨터 교육 분야에서는 이런 개념화에 대한 연구가 부족하고, 특히 오개념 형성을 막고 올바른 개념을 형성시키기 위한 체계적인 노력이 없었다. 그래서 학습자가 형성해야할 프로그래밍 개념은 무엇이며 개념화로 프로그래밍 능력 향상을 구체적으로 돕는 새로운 방식의 연구 접근이 필요하다.

이 연구의 목적은 전문가 집단을 대상으로 프로그래밍에 필요한 개념의 속성을 도출하여 학습자의 개념화를 돕는데 있다. 이를 위해 프로그래밍에서 핵심 요소인 변수(variable), 연산자(operator), 제어 구조(control structure) 등의 프로그래밍 학습 요소를 프로그래밍 교육 전문가 집단이 형성한 개념의 전형적 속성을 질적(qualitative) 연구 방법인 개념적 메타포 분석으로 확인하고 델파이를 통해 타당성을 확보하였다.

II. Background

1. Programming Concepts

개념(concepts)은 사고 혹은 생각이 갖는 내용의 구성단위이다[5]. 그리고 하나의 개념은 추상된 둘 이상의 다른 대상들로 구성된 개념의 외연(extension)과 이런 외연이 공유하는 속성들인 개념의 내포(intension)를 갖는다[6]. 그래서 개념의 내포는 바로 개념을 정의하는 전형적 속성들(typical properties)의 집합이며, 바로 이것이 개념의 의미, 개념의 구조를 형성하는 내용이다[7].

교육 분야에서 개념은 전문가 집단이 선택하여 제시한 것으로 학습자가 형성해야 할 학습 목표가 된다. 그리고 개념은 학습의 이해 수준을 확인하는 근거가 되기 때문에 개념의 속성을 제시하고 설명할 수 있다면 해당 개념을 이해하고 있다고 판단할 수 있다. 그래서 프로그래밍 교육은 학습자가 전문가 집단이 갖고 있는 개념의 내포를 유사하게 형성하는 과정이고, 학습자가 형성한 개념의 내포 확인은 개념화 정도를 확인할 수 있는 방법이 된다.

프로그래밍은 명령어의 의미(semantics)와 연결하여 발생하는 의미를 설계할 수 있는 결합 기술(skills)을 필요로 한다[8]. 그래서 프로그래밍에 필요한 개념은 프로그래밍 요소의 의미와 사용법에 대한 이해를 바탕으로 형성된다[9]. 이와 같이 프로그래밍 경험으로 형성한 개념은 필요한 요소들을 어떻게 적용하고 이용해야 하는지, 어떤 명령어를 선택해서 결합할지를 결정할 수 있게 한다. 따라서 프로그래밍에 필요한 개념은 프로그래밍 요소의 의미와 사용법에 대한 이해를 바탕으로 형성된다.

이 연구에서 프로그래밍 개념(programming concepts)은 프로그래밍에 필요한 개념으로 해당 프로그래밍 요소가 갖는 의미를 이해하고 다른 요소와 관련지어 사용할 수 있는 방법에 대한 이해까지 내포로 구성한 것으로 조작적 정의를 내린다.

2. Conceptual Metaphor Analysis

개념적 메타포 분석(conceptual metaphor analysis)은 인식의 기제(mechanism)로서 메타포를 강조하는 라코프 외(1980)의 개념적 메타포 이론(conceptual metaphor theory)을 연구 방법으로 적용한 것이다[10].

개념적 메타포 분석에 대해 불로우 외(1995)는 의식과 행동 간에 존재하는 긴밀한 관계에 통찰력을 제공한다고 하였고[11], 모저(2000)는 개인의 암묵적, 인지적 표상의 체계적인 구조를 이해하기 위한 양적 및 질적 분석 연구 방법이라고 하였다[12]. 그리고 슈미트(2005)는 개념적 메

타포 분석을 통해 개인 혹은 특정 집단의 사고와 행동유형이나 패턴을 알 수 있다고 말하였다[13].

개념적 메타포 표현은 표현하고자 하는 대상인 목표(target) 영역을 유사성(similarity)을 갖고 있는 근원(source) 영역으로 표현하는 방법이다[14]. 그래서 개념적 메타포 글쓰기는 근원 영역과 목표 영역의 연결 이유(because of)를 해당 개념이 가진 속성(attribute)이 되게 글로 표현하는 방법이다[14]. 예를 들어, '컴퓨터는 연필이다. 왜냐하면, 컴퓨터는 생산성을 갖고 있기 때문이다.'로 표현된 경우는 목표 영역인 '컴퓨터'를 근원 영역인 '연필'로 표현하였고, 그 연결의 이유를 '생산성을 갖는'다고 제시한 것이다. 이때 '컴퓨터'와 '연필'이 갖는 유사한 속성인 '생산성을 갖는'은 '컴퓨터'와 '연필' 등에 모두 갖고 있는 속성이라는 이유로 표현된 결과이다. 이렇게 표현한 개인이 형성한 컴퓨터의 개념에는 '생산성을 갖는'이 속성으로 포함되어 있음을 확인할 수 있다. 이와 같은 개념적 메타포의 표현을 자료로 확보하면 구조적 메타포(structured metaphor) 표현에 해당하는 것으로 체계적인 분석이 가능하다.

이처럼 개념적 메타포 분석은 인간의 학습과 같은 경험을 통해 형성된 개념 체계와 지식의 이해 체계를 해석하고 설명하는 도구가 될 수 있는 혼합 연구 방법이다. 그래서 개인 및 집단이 갖고 있는 인지적 지식의 개념 덩어리를 개념적 메타포 분석을 통해 확인하고 이해할 수 있다.

3. Delphi

델파이(delphi)는 전문가 집단의 경험적 지식을 통해 문제해결이나 미래 예측을 위한 설문 방법이다[15].

델파이가 비대면으로 개인의 의견이 전달되기 때문에 응답자 편향과 같은 집단의 부정행위를 피할 수 있고, 이전 설문 결과에 대한 피드백을 제공하여 조사 대상자에게 질문에 대한 합의를 유도해 내는 집단협의 방식이라는 특징을 갖는다[16].

델파이 절차는 순차적 설문지 형태로 추가 분석을 위해 판단을 통계적으로 요약하고, 해당 그룹으로 다시 전달하는 과정이 반복되어 최종 합의된 결정에 이르게 된다.

델파이를 통해 전문가 의견의 합의 정도를 확인하기 위해 안정도(stability)라는 방법을 사용한다[17]. 이 방법은 반복되는 설문 과정에서 전문가들의 설문 응답 값 차이가 작아 응답의 일치성이 높으면, 안정도가 확보되어 더 이상의 설문이 필요하지 않음을 확인하는 방법이다. 델파이는 설문 발송 및 회신의 과정을 반복하는데, 이를 라운드(round)라고 한다. 그래서 델파이 진행 중에 안정도가 높으면 추가 라운드 없이 설문을 종료할 수 있다.

일반적으로 안정도는 표준편차(StDev)를 산술평균(mean)으로 나눈 값인 변이계수(coefficient of variation; CV)가 사용된다[18]. 이 값이 0.5 이하인 경우 추가적인 라운드가 필요 없으며, 0.5~0.8인 경우 비교적 안정적이라고 판단할 수 있다[18].

델파이의 주목적은 해당 조사 내용에 대해 전문가의 의견이 합의되어 타당성을 확보 받는 것이다. 래쉬(1975)는 전문가를 대상으로 설문 문항의 타당성 정도에 대한 의견을 확보할 수 있는 내용타당도비율(content validity ratio; CVR)을 제안하였다[19]. 문항이 '타당하다'고 동의한 전문가들의 최소 내용타당도비율(CVR)을 만족하면, 타당성이 확보된 것으로 판단하는 방식이다. 그가 제안한 내용타당도비율(CVR)은 아래와 같은 식으로 구할 수 있다[19].

$$CVR = \frac{n_e - N/2}{N/2}$$

위 식에서 n_e는 '타당하다'로 응답한 수이고, N은 전체 전문가의 수이다.

에어 외(2014)는 래쉬(1975)가 제안한 내용타당도비율(CVR) 임계 값을 이항(binomial) 확률로 보다 정확하게 Table 1.과 같이 제시하였다[20].

Table 1. Part of the CVR value of minimum consent experts by Ayre C. et al.[20]

N(Panel Size)	Minimum number of consenting experts	CVR critical exact values
5	5	1.00
6	6	1.00
7	7	1.00
8	7	.750
9	8	.778
10	9	.800
11	9	.636
12	10	.667
13	10	.538

앤더슨(1977)은 델파이에서 10~15명의 소집단의 전문가만으로도 유용한 결과를 얻을 수 있음을 규명하였다[16]. 그리고 송선진 외(1992)는 델파이에 참여하는 전문가의 수가 작은 그룹이 더 효과적이라고 주장하였다[21]. 그래서 델파이 전문가 집단은 소규모 집단의 구성으로도 충분히 유효한 결과를 확보할 수 있다.

III. Methods

1. Delphi Process

이 연구의 전체 연구 방법은 델파이로 진행하고 델파이 1라운드에서 수집된 개방적 글쓰기 자료에 개념적 메타포 분석 방법을 적용하였다. 개념적 메타포 분석의 목적은 전문가 집단이 가지고 있는 프로그래밍 개념의 전형적 속성을 도출하는 것이고, 전문가 집단의 권위를 통해 도출된 분석 결과의 타당성을 확보하는 것이 델파이 설문의 목적이다.

이 연구의 델파이를 통한 연구 절차와 해당 절차의 각 라운드의 진행 목적은 Fig. 1.과 같다.

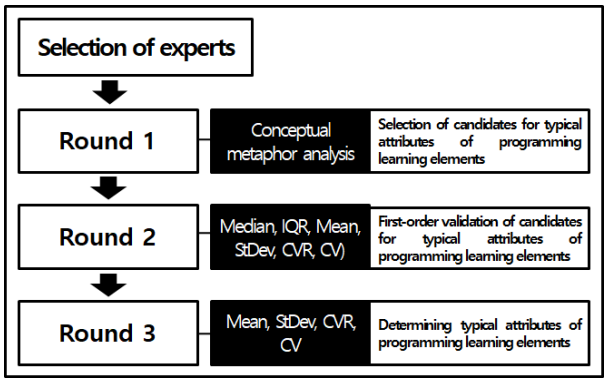

Fig. 1. Delphi procedure

델파이는 전문가 선정을 시작으로 전체 3개 라운드를 진행하였다. 1라운드는 개념적 메타포 분석을 통해 전문가가 형성한 프로그래밍 개념의 전형적 속성을 도출하고, 2라운드는 해당 속성의 1차 타당도를 확인하여 패널(panel)에게 자신의 의견을 수정할 수 있게 정보를 제공하여, 최종 3라운드에서는 해당 프로그래밍 개념의 전형적 속성을 확정하는 과정을 진행하였다.

1.1 Selection of experts

이 연구는 앤더슨(1977)과 송선진 외(1992)의 연구 주장을 근거로, 비확률적 표집의 하나인 의도적 표집 방법으로 15명 이하의 소규모 전문가 집단을 구성하였다[16][21].

이 연구의 참여자는 컴퓨터 교육학 박사 학위소지자, 대학 프로그래밍 교육 강사, 10년차 이상의 정보교사 등을 기준으로, 이 기준에 1가지 이상 해당하는 자를 추천받아 전자메일로 연구에 대한 설명을 송부한 후 참여를 희망한 자로 선정하였다. 이렇게 선정된 전문가 집단은 컴퓨터 교육과 교수 5명, 대학 프로그래밍 강사 3명, 정보교사 3명 등 총 11명으로 구성되었다.

1.2 Round 1

1라운드는 개념적 메타포 글쓰기 방법으로 설문을 진행하였고, 수집된 자료는 개념적 메타포 분석으로 변수, 연산자, 제어 구조 등 프로그래밍 개념의 전형적 속성 후보를 결과로 도출하였다. 도출된 결과는 다음 라운드에 적용할 구조화된 설문의 타당성을 확인하는 문항으로 사용되었다.

1라운드의 개념적 메타포 분석은 고우크 외(2010)가 제안한 5단계 분석 절차에 따라 진행하였다[22]. 그리고 연구의 타당도와 신뢰성을 확보하기 위해 질적 연구 경험이 있는 전문가 1인과 병렬로 분석을 진행하였다. 구체적인 5단계 절차와 분석 방법은 다음과 같다.

1단계는 메타포 판별(identification of metaphors)의 단계로 메타포로 표현한 내용을 목록으로 구성하고 타당하지 않는 표현을 제거하였다.

2단계는 메타포 범주화(categorization of metaphors)의 단계로 범주화를 위해 메타포 표현 내용을 분석하기 위해, 메타포 표현의 단어를 바탕으로 범주를 정하고 유사한 의미를 포함하고 있는 것을 범주에 하나씩 포함시키는 작업을 진행하였다.

3단계는 범주 개발(category development)의 단계로 메타포 표현의 이유인 유사성을 근거로 묶여진 내용의 의미를 공통적으로 나타낼 수 있는 대표적 단어를 바탕으로 검토하여 범주를 개발하였다. 그리고 범주명은 개념의 속성을 파악하는 것으로 사물의 성질이나 상태를 나타내는 품사인 형용사적 표현 종결방식을 취하였다.

4단계는 타당도와 신뢰도 검토(providing the validity and reliability)의 단계로 3단계까지 질적 연구 전문가 1인과 병렬로 분석 진행한 결과를 바탕으로 상호 비교 및 외부 전문가 2인이 분석 결과를 검토하여 분석자 간 내부 검토와 외부 전문가 집단의 검토를 통해 타당도와 신뢰도를 확보하였다.

5단계는 양적통계 분석(transfer the data to analysis program)을 진행하였다. 이 단계에서는 범주와 분류가 이루어진 후 결과에 대한 빈도와 백분율을 산출하였다.

1.3 Round 2

2라운드는 1라운드의 비구조화된 분석 결과를 바탕으로 문항을 작성하고 해당 문항의 타당성을 전문가 응답으로 확인하였다. 2라운드 설문으로 수집된 자료는 각 문항의 평균(mean), 표준편차(StDev), 중위 값(median), 사분위 수범위(IQR), 내용타당도비율(CVR) 등을 구하고 결과를 분석하였다. 델파이는 타인의 의견을 청취하여 자신의 의견을 수정할 기회를 제공한다. 그래서 2라운드에서 확인된

각 문항의 결과 값은 3라운드 설문 안내문과 같이 전자메일에 첨부하여 전송하였다.

1.4 Round 3

3라운드에서 델파이 라운드 종료 결정을 위해 의견의 합의 정도를 변이계수(CV)로 판단하였다. 이 연구는 3라운드에서 응답이 안정적인 것으로 확인되어 추가 라운드 없이 델파이를 종료하였다. 그리고 3라운드에서 문항의 내용타당도비율(CVR)을 최종적으로 확인하고 이를 통해 프로그래밍 개념의 전형적 속성을 확정하였다.

2. Data Collection

자료 수집 기간은 2020년 5월 18일부터 동년 7월 19일까지, 약 2개월 간 진행되었다.

이 연구에서 개념적 메타포 분석에 사용된 자료는 프로그래밍 교육 전문가 집단을 대상으로 변수, 연산자, 제어구조 등에 대한 개방적 글쓰기 방식인 개념적 메타포 글쓰기를 통해 수집하였다.

그리고 델파이 설문의 타당성을 확인하기 위한 문항은 4점 리커트 척도(4 point likert scale) 방식을 적용하였다. 일반적으로 사용하는 5점 리커트 척도는 '보통'으로 표현되는 중간 항은 응답 시 양극단을 피하고 중간을 선택하는 응답자의 경향이 있어 해당 항목의 선택 가능성이 높다[23][24]. 그래서 이 연구에서는 보다 정확한 전문가 의견으로 내용타당도비율(CVR)을 측정하기 위해 중간 항을 제외한 4점 리커트 척도 방식을 적용하였다.

델파이 과정을 진행하기 위해 자료 조사 도구로 구글 온라인 설문조사 폼(google forms), 전자메일 등을 이용하였고, 필요한 파일을 첨부하여 전송하고 회신 받는 등 전체 자료 수집 과정은 온라인으로 진행하였다.

응답 자료는 선정된 전문가 패널 11명이 모두 각 라운드에서 100% 회신하였고, 이렇게 수집된 자료는 모두 분석을 위해 사용되었다.

3. Analysis Method

개념적 메타포 분석을 체계적으로 진행하기 위해 질적 연구 도구 프로그램인 파랑새 2.0(Bluebird 2.0)을 사용하였다. 파랑새 2.0은 한국형 질적 연구 플랫폼으로 웹(web)에서 전사, 코딩, 주제 생성으로 이어지는 핵심적인 질적 연구 과정을 수행할 수 있는 도구이다[25]. 파랑새 2.0을 이용한 질적 분석은 1단계-자료 입력, 2단계-코딩 지정, 3단계-코딩 구조 만들기(개념화), 4단계-주제 생성(범주화), 5단계-글쓰기 등의 단계로 안내되어 있다[25]. 이 연구는 개념적 메타포 글쓰기를 통해 수집한 자료를 바탕으로 각 프로그래밍 요소가 갖는 속성의 범주화가 목적이므로 자료를 입력하고, 코딩 지정하여 코딩 구조를 만들어, 주제인 범주를 생성하는 과정까지만 사용하였다. 이 연구에서 파랑새 2.0을 활용한 진행과정에서 코딩 구조화와 범주화를 위한 주제 생성 예시는 Fig. 2., Fig. 3. 등과 같다.

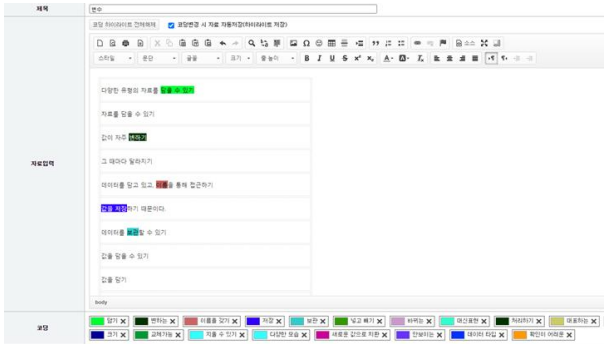

Fig. 2. Coding structure example webpage of Bluebird 2.0

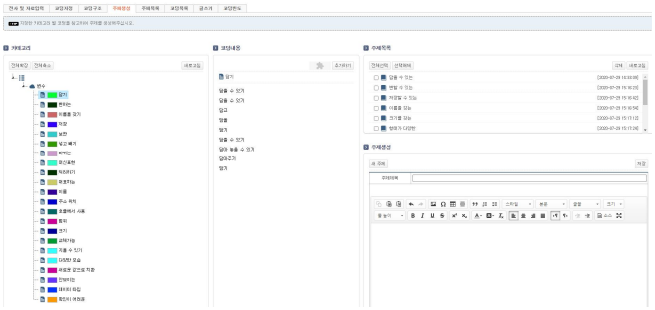

Fig. 3. Subject creation example webpage of Bluebird 2.0

델파이 설문 문항을 통해 수집된 자료의 타당성을 확인하기 위한 통계 분석 도구로 SPSS 프로그램과 엑셀(excel) 프로그램을 사용하였다.

IV. Results

1. Results of Round 1

1라운드 델파이는 전문가 집단의 개념적 메타포 글쓰기 자료를 수집하여 개념적 메타포 분석을 진행하였다. 그 결과, 프로그래밍 교육 전문가 집단의 프로그래밍 요소에 대한 개념적 메타포 표현(conceptual metaphor expression)은 2명 이상이 중복되게 표현하여, 군집화가 가능한 전형적 속성을 갖고 있음이 확인되었다. 그 구체적인 결과는 Table 2.와 같다.

Table 2. Conceptual metaphor analysis result of programming elements for experts

Elements	Code	Typical attribute candidate	n(%)
Variable	V1	Containable	11(100)
	V2	Changeable	10(90.9)
	V3	Storable	8(72.7)
	V4	Named	6(54.5)
	V5	Sized	5(45.4)
	V6	Diverse in form	4(36.3)
	V7	Frequently used	3(27.2)
	V8	Invisible	3(27.2)
	V9	Addressed	3(27.2)
	V10	Easy to delete	3(27.2)
	V11	Temporary	3(27.2)
Operator	O1	Converting	10(90.9)
	O2	Arithmetic	6(54.5)
	O3	To compared	6(54.5)
	O4	Logically processed	6(54.5)
	O5	Connected	5(45.4)
	O6	Controlling	4(36.3)
	O7	Symbolic	3(27.2)
	O8	Finding data	3(27.2)
	O9	Making a relationship	2(18.1)
	O10	Distinguishing form	2(18.1)
	O11	Prioritized	2(18.1)
Control structure	C1	Logically judged	10(90.9)
	C2	To control the flow	7(63.6)
	C3	Finding the way	6(54.5)
	C4	Repeating	6(54.5)
	C5	Observing the rules	5(45.4)
	C6	To choose	4(36.3)
	C7	To solve the problem	4(36.3)
	C8	Assembled	4(36.3)
	C9	Made simple	2(18.1)
	C10	Executed in sequence	2(18.1)
	C11	Overlapping	2(18.1)

전문가 집단이 개념적 메타포로 표현한 프로그래밍 개념의 범주화된 속성은 변수 속성 11가지, 연산자 속성 11가지, 제어 구조 속성 11가지 등이 확인되었다. 이와 같이 총 33가지 속성은 각 프로그래밍 요소의 개념이 갖는 전형적 속성 후보가 된다.

2. Results of Round 2

2라운드 델파이 조사는 1라운드 델파이 결과를 바탕으로 범주로 도출되었던 33개의 프로그래밍 요소가 갖는 전형적 속성 후보를 대상으로 진행하였다. 2라운드 설문은 각 문항으로 구성된 전형적 속성의 후보들이 해당 요소의 전형적 속성으로 타당한지를 평가하는 것이다. 응답 자료의 분석을 통해 각 문항의 평균(mean), 표준편차(StDev), 중위 값(median), 사분위수범위(IQR), 내용타당도비율(CVR), 변이계수(CV) 등의 값을 도출하였다.

프로그래밍 요소가 갖는 전형적 속성 후보 33개를 대상으로 전문가 집단에게 4점 리커트 척도로 타당성 정도를 확인받은 구체적인 결과는 Table 3.과 같다.

Table 3. Expert opinion on candidates for typical attributes of programming learning elements

Elements	Code	Mean	StDev	Median	IQR	CVR	CV
Variable	V1	3.64	.674	4	1	0.818	0.185
	V2	3.55	.820	4	1	0.636	0.231
	V3	3.73	.467	4	1	1	0.125
	V4	3.45	.688	4	1	0.818	0.199
	V5	3.27	.647	3	1	0.818	0.198
	V6	3.27	.786	3	1	0.636	0.240
	V7	2.91	.701	3	1	0.455	0.241
	V8	2.55	.688	2	1	-0.091	0.270
	V9	3.36	.505	3	1	1	0.150
	V10	2.36	.505	2	1	-0.273	0.214
	V11	2.91	.831	2	1	0.273	0.286
Operator	O1	3.36	.809	4	1	0.636	0.241
	O2	3.55	.688	4	1	0.818	0.194
	O3	3.27	.786	3	1	0.636	0.240
	O4	3.00	.632	3	1	0.636	0.211
	O5	2.64	.924	3	1	0.273	0.350
	O6	3.36	.674	3	1	0.818	0.201
	O7	2.55	.820	3	1	0.273	0.322
	O8	2.91	.831	3	1	0.818	0.286
	O9	2.82	.751	3	1	0.273	0.266
	O10	3.18	.603	3	1	0.818	0.190
	O11	3.18	.603	3	1	0.818	0.190
Control structure	C1	3.64	.505	4	1	1	0.139
	C2	2.91	.944	3	2	0.455	0.324
	C3	3.36	.674	3	1	0.818	0.201
	C4	3.55	.522	4	1	1	0.147
	C5	3.36	.505	3	1	0.818	0.150
	C6	3.00	.632	3	1	0.636	0.211
	C7	2.36	.809	2	1	-0.091	0.343
	C8	2.55	.820	3	1	0.091	0.322
	C9	2.64	.809	3	1	0.455	0.306
	C10	3.27	.467	3	1	1	0.143
	C11	2.73	.647	3	1	0.455	0.237

2라운드는 변이계수(CV) 값이 0.1251 ~ 0.35 사이에 모두 존재하여 응답이 안정적이라고 볼 수 있다.

2라운드 델파이 결과, 내용타당도비율(CVR)을 통해 타당성이 확인된 문항은 변수에서 7가지, 연산자에서 8가지, 제어 구조에서 6가지 등으로 총 21가지가 확인되었다.

Table 3.과 같은 2라운드 결과는 3라운드 설문 참여 안내를 위한 전자메일 전송 시 첨부하여 전달하였다.

3. Results of Round 3

3라운드에서도 전문가 11명 모두가 델파이 설문에 참여해서 응답하였다. 응답 결과, 변이계수(CV)가 0.5이하로 모두 안정적으로 파악되어 추가 라운드는 진행되지 않아도 되었다. 그래서 3라운드 응답 결과를 바탕으로 변수, 연산자, 제어 구조 등 프로그래밍 개념의 전형적 속성을 확정하였다.

3.1 Typical properties of variable

변수 개념의 전형적 속성 후보 11개를 대상으로 전문가 집단에게 4점 리커트 척도로 타당성 정도를 확인받은 결과는 Table 4.와 같다.

Table 4. Results of validation about typical properties of variable concepts

Code	Mean	StDev	Validity frequency	CVR	CV
V1	3.27	.905	8	0.455	0.276
V2	3.55	.688	10	0.818	0.194
V3	3.82	.405	11	1	0.106
V4	3.64	.505	11	1	0.139
V5	3.27	.786	9	0.636	0.240
V6	3.45	.688	10	0.818	0.199
V7	2.91	.831	6	0.091	0.286
V8	2.73	.786	6	0.091	0.288
V9	3.18	.405	11	1	0.127
V10	2.82	.603	8	0.455	0.214
V11	2.91	.701	8	0.455	0.241

변수 개념에서 설문 응답의 변이계수(CV)는 0.106~0.288 사이의 값으로 합의가 안정되게 이루어졌다.

변수 개념의 전형적 속성 후보 중 타당성을 확보한 것은 모두 6개로 나타났다. ‘V3: 저장할 수 있는’, ‘V4: 이름을 갖는’, ‘V9: 주소를 갖는’ 등은 11명 모두 긍정적 응답을 제시하여 내용타당도비율(CVR) 값은 1로 가장 높은 결과를 보여주었다. 그리고 ‘V2: 변할 수 있는’, ‘V6: 형태가 다양한’ 등은 긍정적 응답이 10명, 내용타당도비율(CVR) 값은 0.818로, 다음 순으로 타당성을 확보하였다. 또한, ‘V5: 크기를 갖는’도 타당하다고 응답한 전문가는 9명, 내용타당도비율(CVR)은 0.636으로 타당성이 확보되었다. 이상의 6가지 속성은 전문가 집단의 합의를 통해 타당성이 확보된 것으로 변수 개념이 갖는 전형적 속성으로 확정되었다.

3.2 Typical properties of operator

연산자 개념의 전형적 속성 후보 11개를 대상으로 전문가 집단에게 4점 리커트 척도로 타당성 정도를 확인받은 결과는 Table 5.와 같다.

연산자 개념에서 설문 응답의 변이계수(CV)는 0.139~0.307 사이의 값으로 합의가 안정되게 이루어졌다.

연산자 개념의 전형적 속성 후보 중 타당성을 확보한 것은 5개로 나타났다. ‘O2: 사칙 연산하는’, ‘O4: 논리적으로 처리하는’, ‘O11: 우선순위가 있는’ 등은 11명이 모두 긍정적 응답을 제시하여 내용타당도비율(CVR) 값은 1로 가장 높은 결과가 나타났다. 그리고 ‘O3: 비교하는’과 ‘O7: 기호를 가지는’은 각각 10명의 전문가가 타당다고 응답하여 내용타당도비율(CVR) 값은 0.818로 그 다음 순으로 타당성이 확보되었다. ‘O7: 기호를 가지는’의 속성은 2라운드에서 내용타당도비율(CVR) 값이 0.273이었으나 3라운드에서는 0.818로 높아졌고, 변이계수(CV)값이 2라운드의 0.322에서 3라운드는 0.190으로 안정도가 높아져, 긍정적인 합의로 수정된 결과가 나타났음을 확인하였다. 이상의 5가지 속성은 전문가 집단의 합의를 통해 타당성이 확보되어 연산자 개념이 갖는 전형적 속성으로 확정되었다.

Table 5. Results of validation about typical properties of operator concepts

Code	Mean	StDev	Validity frequency	CVR	CV
O1	3.09	.831	8	0.455	0.269
O2	3.64	.505	11	1	0.139
O3	3.64	.674	10	0.818	0.185
O4	3.45	.522	11	1	0.151
O5	3.00	.775	8	0.455	0.258
O6	2.64	.809	5	-0.091	0.307
O7	3.18	.603	10	0.818	0.190
O8	2.55	.688	5	-0.091	0.270
O9	3.00	.775	8	0.455	0.258
O10	2.82	.603	8	0.455	0.214
O11	3.55	.522	11	1	0.147

3.3 Typical properties of control structure

제어 구조 개념의 전형적 속성 후보 11개를 대상으로 전문가 집단에게 4점 리커트 척도로 타당성 정도를 확인받은 결과는 Table 6.과 같다.

Table 6. Results of validation about typical properties of control structure concepts

Code	Mean	StDev	Validity frequency	CVR	CV
C1	3.55	.688	10	0.818	0.194
C2	3.73	.467	11	1	0.125
C3	3.00	.775	8	0.455	0.258
C4	3.36	.809	9	0.636	0.241
C5	3.27	.647	10	0.818	0.198
C6	3.73	.647	10	0.818	0.174
C7	2.73	.647	7	0.273	0.237
C8	2.64	.505	7	0.273	0.191
C9	2.64	.674	6	0.091	0.256
C10	3.45	.688	10	0.818	0.199
C11	2.91	.701	8	0.455	0.241

제어 구조 개념에서 설문 응답의 변이계수(CV)는 0.125~0.275 사이의 값으로 합의가 안정되게 이루어졌다.

제어 구조 개념의 전형적 속성 후보 중 타당성을 확보한 것은 모두 6개로 확인되었다. ‘C2: 흐름을 제어하는’은 11명 모두가 타당하다고 응답하여 내용타당도비율(CVR)이 1로 나타났고, ‘C1: 논리적으로 판단하는’, ‘C5: 규칙을 지키는’, ‘C6: 선택하는’, ‘C10: 순서대로 실행하는’ 등은 10

명이 긍정적 응답을 제시하여 내용타당도비율(CVR) 값이 0.818로 다음 순으로 높게 나타났다. 다음으로 'C4: 반복하는'은 9명이 타당하다고 응답하여 내용타당도비율(CVR) 값이 0.636으로 타당성을 확보하였다. 다만, 'C4: 반복하는'은 2라운드 보다 3라운드 결과에서 긍정적 타당성 평가 응답이 11명에서 9명으로 줄고 변이계수(CV)가 0.147에서 0.241로 커져 합의정도가 나빠진 것으로 확인되었다. 이상의 6가지 속성은 전문가 집단의 합의를 통해 제어 구조 개념이 갖는 전형적 속성으로 확정되었다.

V. Discussions

이 연구의 결과는 프로그래밍 교육 전문가 집단이 표현하고 합의된 프로그래밍 개념의 객관적인 전형적 속성이다. 그래서 변수의 6가지, 연산자의 5가지, 제어 구조의 6가지 등 프로그래밍 요소의 17가지 전형적 속성에 대한 이해(understanding)는 프로그래밍에 필요한 지식(knowledge)으로 말할 수 있다. 즉, 전형적 속성으로 확인된 17가지에 대한 의미와 조작 방법의 이해는 변수, 연산자, 제어 구조 등의 프로그래밍 요소를 사용할 때 필요한 개념에 관한 지식이다. 그래서 이와 같은 프로그래밍 개념에 관한 지식은 학습자가 프로그래밍 경험을 통해 형성해야할 교육 내용의 목록이 될 수 있다. 이를 바탕으로 교육과정을 구성하고 적절한 프로그래밍 경험을 제공한다면 개념화에 보다 효과적인 교육과정이 될 것이다.

또한 프로그래밍 교육과정의 내용을 계열화할 때 필요한 개념의 전형적 속성을 학습 환경과 학습자의 수준에 맞게 선정하면 학습자 수준을 고려한 나선형 교육과정 편성에 도움이 될 것이다. 예를 들어, 프로그래밍 입문자 과정에서 주로 사용하는 비주얼 방식의 교육용 프로그래밍 언어(educational programming language)의 환경에서는 변수의 속성 중 주소의 의미나 변수의 선언에서 자료구조에 따른 제약에 대한 이해가 필요 없지만, 텍스트 방식을 경험하게 되면서 해당 개념의 속성이 필요하다. 따라서 전문가로 성장하기 위해서는 더 다양한 속성의 이해가 필요하기 때문에 해당 개념 형성에 필요한 속성을 적절히 선정하여 제시하고 개념을 확장할 수 있게 수준별 교육과정을 구성할 수 있다.

프로그래밍 오개념(misconception)은 해당 개념의 충분한 이해가 부족하여 발생한다[1-4]. 즉, 프로그래밍 개념화는 프로그래밍 환경에 맞게 개념이 갖는 전형적 속성에 대한 총체적인 이해가 반드시 필요하다. 만약 이와 같은 프로그래밍 개념이 갖는 속성의 이해가 부족하다면 해당 환경에서 개념 사용에 문제가 발생하여 초보자는 프로그래밍에 대한 어려움을 겪게 된다. 예를 들어, 연산자가 우선순위를 갖고 있고 조합 방법에 대한 이해가 없으면 그것은 연산자에 대한 오개념 형성의 원인이 된다. 그리고 이런 부분적인 이해로 인해 결핍된 속성은 연관된 프로그래밍 요소와 조합될 때 연쇄적으로 문제를 일으킬 것이다. 결국은 개념의 특정 속성의 이해 결핍은 프로그래밍에서 오류를 발생시키는 원인이 된다.

그래서 이 연구의 결과로 확인된 프로그래밍 요소의 전형적 속성은 프로그래밍 교육에서 학습자의 오개념이나 프로그래밍 오류의 발생 원인을 진단하거나 프로그래밍 결과가 어떻게 될 것인지를 추론할 수 있는 근거로 사용할 수 있다. 학습자가 발생시킨 프로그래밍 오류는 복잡하고, 오개념이 형성될수록 더욱 창의적인 오류를 발생시킬 수 있다. 학습자가 발생시킨 구문 오류는 찾아 수정하기는 쉽지만, 그렇게 생각한 원인을 찾기는 어렵다. 그런데 해당 개념의 전형적인 속성에 대한 이해 결핍을 확인한다면 오류 발생의 원인을 이해할 수 있고, 이에 대한 학습을 추천한다면 올바른 개념화를 위한 학습 처방이 될 것이다. 그래서 이 연구에서 확인된 프로그래밍 개념의 전형적 속성을 평가 근거로 활용하면 학습자가 발생시킨 오류를 추론할 수 있고 체계적인 학습 진단과 처방을 내릴 수 있는 평가 도구를 개발할 수 있다.

그리고 프로그래밍 개념의 전형적 속성에 대해 총체적으로 이해하고 구체적으로 설명할 수 있는 능력은 프로그래밍 교육자에 필요한 효과적인 내용교수지식(pedagogical content knowledge; PCK)이 될 수 있다. 예를 들어, 변수 개념의 속성인 '저장할 수 있는' 속성은 저장하는 그릇과 같은 형태에 대한 개념이 아니라 그릇이 갖는 속성인 저장할 수 있는 속성을 떠올리면서 그릇이 사용되는 방법과 원리에 대한 이해를 바탕으로 개념화하고 사용할 수 있게 된다. 그래서 프로그래밍 개념의 전형적 속성을 전문가 집단이 갖고 있는 개념의 속성으로 이해하는 것은 해당 개념의 원리가 포함된 사용에 대한 이해를 수월하게 형성할 수 있게 도와준다. 그래서 이 연구의 결과로 확인된 프로그래밍 개념의 전형적 속성에 대한 의미와 방법적 이해는 프로그래밍 교육에서 개념화를 위해 학습자에게 예의 양호도(goodness of examples)가 충족된 형태로 제시할 수 있는 교육자의 지식이 될 수 있다. 학습자가 구체적인 명령어의 의미와 사용 방법에 대한 이해를 갖기 위해서 학습자가 이해 가능한 구체적인 속성의 예를 제시한다면 보다 빠르고 정확한 이해와 개념 형성에 도움이 될 것이기 때문이다. 그

래서 이와 같은 개념에 관한 지식은 프로그래밍 교육활동에 필요한 내용교수지식(PCK)이 될 수 있다.

따라서 이 연구로 확인된 총 17가지 프로그래밍 개념화에 필요한 전형적 속성은 프로그래밍에 필요한 개념에 관한 지식으로 프로그래밍 교육과정 내용을 구성하는데 중요한 역할을 할 수 있고, 개념 형성 정도를 확인할 수 있는 구체적인 근거로 평가 도구 개발에 활용될 수 있으며, 프로그래밍 교육자가 형성해야할 교수내용지식(PCK)으로 활용될 수 있다.

VI. Conclusions

프로그래밍에서 필요한 개념은 사용(using)의 상황에서 논리적으로 연결하는 방법에 대한 이해를 포함한다. 그래서 학습자가 형성해야할 개념은 프로그래밍 언어의 요소나 명령어가 갖는 의미(semantics), 그리고 이런 요소와 명령어를 서로 연결하는 방법적인 스킬(skills)을 포함하여 이해한 결정체이다.

이 연구는 델파이를 통해 프로그래밍 교육 전문가가 인식하고 있는 프로그래밍 개념에 대해 질적 및 양적 연구로 학습자의 올바른 개념화를 돕기 위한 프로그래밍 개념의 속성을 구체적으로 파악하는 시도를 하였다. 그래서 프로그래밍 전문가 집단이 형성한 프로그래밍 개념의 속성은 중복되는 표현이 확인되어 전형적인 것으로 군집화하고 범주명을 붙여서 타당성을 확보하였다. 그 결과, 프로그래밍 교육에서 학습자가 형성해야할 변수, 연산자, 제어 구조 등의 프로그래밍 요소의 객관적인 17가지 전형적 속성을 파악하였다. 그리고 프로그래밍 개념이 갖는 전형적 속성의 중요성과 활용 가능성에 대해 논의하였다. 이상의 결과와 논의를 바탕으로 내린 결론은 다음과 같다.

첫째, 학습자는 변수 개념의 속성으로 데이터를 저장할 수 있는 방법, 이름을 정하는 방법, 변수 주소가 갖는 의미, 값을 변하게 하는 방법, 다양한 형태, 변수 크기의 의미 등에 대해 이해할 수 있어야 한다.

둘째, 학습자는 연산자 개념의 속성으로 사칙 연산 방법, 연산의 논리적 처리 방법, 우선순위에 따른 연결 방법, 연산자 기호의 의미, 연산자로 비교하는 방법 등에 대해 이해할 수 있어야 한다.

셋째, 학습자는 제어 구조 개념의 속성으로 실행 흐름 제어 방법, 참/거짓의 논리적 판단 방법, 실행 규칙 설정 방법, 선택적 실행 방법, 프로그램의 순차적 실행의 의미, 반복적 실행 방법 등에 대해 이해할 수 있어야 한다.

넷째, 프로그래밍 개념의 평가도구는 해당 개념의 구체적인 속성에 대한 이해를 확인할 수 있게 개발되어야 한다. 즉, 올바른 개념화는 개념이 갖는 속성의 이해 결핍이 없어야하기 때문에 프로그래밍 평가 도구는 이를 확인할 수 있도록 구상되어야 한다. 그래서 해당 개념의 다양한 속성의 이해 정도를 확인하고 결핍된 속성에 대해 학습 처방해주는 것은 학습자의 오류 발생 원인을 진단하고 발생 가능한 프로그래밍 어려움을 미리 방지하고 프로그래밍 개념화를 직접적으로 도와줄 수 있는 평가 방법이 될 것이다.

그런데, 개인의 개념의 구조를 파악하고 평가하는 과정은 질적 연구 방법으로 시간과 노력이 많이 든다. 이와 같은 방법으로 학습자의 개념화 수준을 평가하는 것은 비효율적인 방법이다. 그래서 학습자의 개념화를 구체적으로 파악하는 방법을 자동화할 수 있는 연구가 필요하다.

이 연구의 추후 과제로 프로그래밍 개념의 전형적 속성이 학습 경험의 차이로 인해 발생하는 가중치의 변화를 확인하여 개념화 정도를 자동화된 방법으로 평가할 수 있는 연구를 진행하고자 한다.

REFERENCES

[1] Lan Jeong, “Analysis of characteristics of novice programmers and teaching schemes,” Journal of Theses Collection. Vol. 29, No. 1, pp. 388-411, 1996.

[2] Fleury, A. E., “Programming in Java: student-constructed rules,” ACM SIGCSE Bulletin. Vol. 32, No. 1, pp. 197-201, Jan. 2000.

[3] Eckerdal, A. and Thune´, M., “Novice Java programmers’ conceptions of object and class, and variation theory,” ACM SIGCSE Bulletin. Vol. 37, No. 3, pp. 89-93, Jun. 2005.

[4] Siti Rosminah, M. D. and Ahmad Zamzuri, M. A., “Difficulties in learning programming: Views of students,” 1st International Conference on Current Issues in Education, Sep. 2012. DOI: 10.13140/2.1.1055.7441

[5] Young-Ah Jo, “Prototype Theory And Problem of Compositionality,” Journal of Korean philosophical society. Vol. 132, No. -, pp. 151-184, Dec. 2014.

[6] Bongju Kim, “Conception-Foundation of Semantics,” Seoul: Han-Shin Media, Jan. 1996.

[7] 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, May. 2008.

[8] Linn, M. C. and Dalbey, J., “Cognitive consequences of Programming Instruction: Instruction, Access, and Ability,” Educational Psychologist. Vol. 20, No. 4, pp. 191-206, Sep. 1985.

[9] Kwon, K., “Student’s misconception of programming reflected on

problem-solving plans," International Journal of Computer Science Education in Schools. Vol. 1, No. 4, pp. 14-25, Oct. 2017. DOI: 10.21585/ijcses.v1i4.19

[10] Lakoff, G. and Johnson, M., "Metaphors we live by," Chicago: University of Chicago Press, Dec. 2008.

[11] Robert Bullough and Gitlin, A, Becoming a Student of Teaching: Methodologies for Exploring Self and School Context. Teaching Education. Vol. 8, No. 1, pp. 167-170, Jul. 2006. DOI: 10.1080/1047621960080123

[12] Moser, K. S., "Metaphor analysis in phychology-method, theory, and fields of application," Forum: Qualitative Social Research(on-line journal). Vol. 1, No. 2, pp. 22-31, Jun. 2000.

[13] Schmitt, R., "Systematic metaphor analysis as a method of qualitative research," The Qualitative Report. Vol. 10, No. 2, pp. 358-394, Jun. 2005.

[14] Dong-Man Kim and Tae-Wuk Lee, "Understanding about Novice Learner's Programming Conception by Prototype Theory. The Journal of the Korea society of computer and information. Vol. 25, No. 3, pp. 251-260, Mar. 2020.

[15] O. Helmer and N. Rescher, "On the epistemology of the inexact sciences," The Journal of Management Science. Vol. 6, No. 1, pp. 5-52, Oct. 1958.

[16] Anderson, E. T., "Important distance education practices: A delphi study of administrators and coordinators of distance education programs in Higher Education," Ph. D. Thesis. University of Idaho, Jan. 1997.

[17] Yong-Joo Kang, "Understanding and application examples of Delphi technique," Seongnam: Korea Employment Agency for the Disabled, Jul. 2008.

[18] Seung-Yong Noh, "Delphi technique: predicting the future with expert insight," The Journal of Korea Research Institute For Human Settlements. Vol. 299. No. 9, pp. 53-62, Sep. 2006.

[19] Lawshe, C. H., "A quantitative approach to content validity," Personnel Psychology. Vol. 28, No. 4, pp. 563-588, Jul. 1975.

[20] Ayre C. and Andrew John Scally, "Critical Values for Lawshe's Content Validity Ratio. Measurement and Evaluation in Counseling and Development. Vol. 47, No. 1, pp. 79-86, Dec. 2013.

[21] Sung-Jin Song and Do-Keun Yoon, "A Study on the prospection of Long Term Care facilities by Delphi Technique," The Journal of Architectural Institute of Korea. Vol. 8, No. 7, pp. 85-93, Jul. 1992.

[22] Gők, B. and Erdorğam, T., "Investigation of pre-service teachers' perceptions about concept of technology through metaphor analysis," The Turkish Online Journal of Educational Technology. Vol. 9, No. 2, pp. 145-160, Apr. 2010.

[23] In-hwan Oh, "Social Research Methodology-Focusing on Error Factors," Paju: Nanam Publishing, Jul. 1995.

[24] Kyung Sun Oh and Seong Jin Ahn, "A study on the relationship between difficulty in learning to program and Computational Thinking," The Journal of Korean association of computer education. Vol. 18, No. 5, pp. 55-62, Sep. 2015.

[25] Young Beom Oh, Hyun Cheol Lee and Sang Won Jeong, "Qualitative Data Analysis Bluebird 2.0 Software," Seoul: Academy Press, Apr. 2016.

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 programming education and artificial Intelligence.

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.