Journal of The Korean Association
of Information Education Vol. 22, No. 2,
April 2018, pp. 251-262

http://dx.doi.org/10.14352/jkaie.2018.22.2.251

미국 컴퓨터교육 프레임워크 분석을 통한 Computational Thinking의 개념과 교육과정 편성의 시사점 분석

신승기* · 배영권**

대구교육대학교 글로벌교육혁신연구소* · 대구교육대학교 컴퓨터교육과**

요 약

본 연구에서는 우리나라의 소프트웨어 교육과정 편성과 Computational Thinking에 대한 개념정립을 위하여 미국의 K12CS에서 제시한 컴퓨터교육 프레임워크를 분석함으로써 시사점을 도출하고자 하였다. 첫 번째로 우리나라에서 Computational Thinking에 대한 용어를 컴퓨팅 사고력으로 사용하는 부분에 대해서 미국의 교육과정에 제시된 개념과 비교하여 살펴보았으며, 컴퓨팅 사고력과 Computational Thinking은 문제 해결의 초점과 범위가 다름을 살펴볼 수 있었다. 두 번째로, 우리나라의 소프트웨어 교육과정은 학교급 및 학년에 따른 위계가 고려되지 않는다는 점을 고려하여 미국의 K12CS에서 제시하고 있는 핵심 역량을 토대로 재구성하여 교육과정 체계를 제시하였다.

키워드 : 컴퓨팅 사고력, 소프트웨어 교육, 교육과정, 컴퓨팅, 컴퓨터교육

The Concept of Computational Thinking through Analysis of Computer Education Framework in the United States and its Implications for the Curriculum of Software Education

Seungki Shin*, Youngkwon Bae**

Institute of Global Education for the Advancement of Innovative Learning,
Daegu National University of Education*,
Dept. of Computer Education, Daegu National University of Education**

ABSTRACT

In this study, we conducted to derive some implications by analyzing the computer education framework proposed by K12CS in the United States in order to organize the software curriculum and conceptualization of computational thinking in Korea. First, we discussed the use of the term Computational Thinking as a Computing Thinking in Korea and compare it with the concept presented in the US curriculum. we derived that Computing Thinking and Computational Thinking are different in the focus and scope of problem solving. Second, considering the fact that Korean software

교신저자 : 배영권(대구교육대학교 컴퓨터교육과)
논문투고 : 2018-04-19
논문심사 : 2018-04-23
심사완료 : 2018-04-24

curriculum does not consider the hierarchy according to the school and the grade, we reconstructed the curriculum based on the core practices and concepts which were suggested by the organization of K-12 Computer Science in the United States.

Keywords : Computational Thinking, Software Education, Curriculum, Computing, Computer Education

1. 서론

소프트웨어 교육은 4차 산업혁명의 시대의 도래에 따라 미래의 인재를 길러내기 위한 핵심 교과로 자리매김하고 있다. 이에 따라 2015개정교육과정에서는 소프트웨어 교육이 필수교육과정으로 도입이 되어 2018년부터는 중학교과정에 독립교과로서의 필수 교육과정으로 반영이 되고 2019년부터는 초등학교의 5~6학년에 제시되어 교육과정의 영역에 반영이 될 예정이다[10][11][12].

소프트웨어 교육의 핵심 사고전략은 컴퓨팅 사고력으로 일컬어지는 Computational Thinking이며, 이는 문제해결력을 의미하는 체계적인 사고과정이다[11][12]. 우리나라의 교육과정에서도 문제해결력을 기반으로 하는 Computational Thinking을 향상시키기 위한 교수학습체제가 구성이 되어있고 교수학습모형이 안내가 되어 있으나, 처음 교육과정으로 도입이 되는 교과이며 관련 개념과 프레임워크 등이 정립이 되어가는 단계라는 점을 고려하여 볼 때, 컴퓨터교육과정이 이미 편성되어 운영되고 있는 해외의 다양한 사례를 살펴본다면 우리나라의 교육과정 편성과 추후 개정교육과정에 반영해야할 요소와 시사점을 도출할 수 있을 것이다.

해외의 여러 나라들은 소프트웨어 교육을 도입함에 있어서 교육과정 편성과 함께 프레임워크 구성을 위한 연구와 Computational Thinking의 개념 정립을 위한 노력을 기울이고 있다. 특히, 미국의 경우 Papert(1996)에 의한 Computational Thinking에 대한 개념 도입에서부터[13][14] Wing(2006)의 개념 확립의 단계 이르는 과정까지 소프트웨어 교육의 기저 철학과 사고과정에 대한 연구가 실시되었으며[22], 이후 CSTA(Computer Science Teachers Association)와 IEEE(Institute of Electrical and Electronics Engineers) 및 ACM(Association for Computing Machinery) 등은 교육과정의 편성과 체제구성을 위한 노력을 기울여왔다[4].

미국의 경우 연방국가라는 점을 고려하여 우리나라나 영국 등과 같이 국가수준의 교육과정을 제시하지 않기 때문에 교과교육과 관련하여 국가수준의 지침을 제시할 뿐 세부적인 교육과정 편성 및 영역에 대한 내용은 제시하지 않고 주(State)별로 세부 내용을 편성하여 교육과정을 구성하고 있다. 이에 따라 미국에서는 컴퓨터교육과 관련된 연합체(K-12 Computer Science)[1]가 구성이 되어 미국의 전체 지역에 지침이 될 수 있는 교육과정을 편성하고 보급함으로써 각 주의 교육부에 참고가 될 수 있도록 제시하고 있다[7]. 교육과정 설계의 단계에서 14개의 주와 4개의 교육청이 참여하고 국가수준의 연구기관이 주도함으로써 실질적인 국가수준의 교육과정 지침으로써 내용이 편성되어 있다[7].

또한 우리나라의 소프트웨어 교육 운영 지침을 비롯하여 영국, 핀란드, 에스토니아 등 소프트웨어 교육을 필수 교육과정으로 운영하는 국가에서는 미국의 CSTA 및 ISTE에서 주도한 교육과정 편성을 국가수준교육과정 편성에 참고하고 있다는 점을 고려한다면 최근에 개정 및 완성된 미국의 소프트웨어 교육과정 편성에 대한 내용을 살펴본다면 우리나라의 향후 소프트웨어 교육과정 운영과 개선에 필요한 시사점을 도출할 수 있을 것이다.

본 연구에서는 실질적인 미국의 국가수준 교육과정이라고 할 수 있는 컴퓨터교육 연합체(K-12 Computer Science)에서 제시한 교육과정 편성과 세부 내용을 살펴봄으로써 우리나라의 소프트웨어 교육 안착과 교육과정 편성 및 Computational Thinking에 대한 시사점을 도출하고자 한다.

1) ACM, Code.org, CSTA, Cyber Innovation Center, National MATH+SCIENCE Initiative로 구성됨

2. 선행연구 분석

2.1 해외의 소프트웨어 교육관련 교육과정 분석

인도는 세계적으로 IT강국이라고 불리며 전 세계의 소프트웨어 관련 주요 핵심 인력을 배출하는 소프트웨어 교육의 우수 국가 중 하나로 일컬어진다. 인도의 컴퓨터 교육과정은 CMC(Computer Masti Curriculum)이라고 불리며 즐거운 컴퓨터 교육과정이라는 의미를 갖는다. 인도의 컴퓨터 교육과정은 학년별로 범위와 내용이 정해져 있으며, 세부 내용은 아래의 <Table 1>과 같이 나타난다[16][20].

<Table 1> Computer Education Curriculum in India

Grade1	Grade2	Grade3	Grade4	Grade5	Grade6
1. Categorizing and organizing information					
	■	■	■	■	■
2. Algorithmic thinking					
		■	■	■	■
3. Logical Reasoning / Problem solving					
			■	■	■
4. Gathering Information					
				■	■
5. Decision making					
				■	■
6. Brainstorming, analysis of ideas and synthesis					
					■
7. Using multiple representations					
				■	■

에스토니아는 e-Estonia로서 정부차원의 전자정부로 지향점을 구축하고 이를 토대로 짧은 시간에 급속도로 성장한 나라 중 하나이다. 이는 산업 전반뿐만 아니라 교육에도 영향을 미치게 되었으며, ProgeTiger라는 프로젝트를 통해 정부차원에서의 소프트웨어 교육과정을 추진하고 있다. 에스토니아의 소프트웨어 교육관련 교육과정의 세부내용을 살펴보면 아래의 <Table 2>와 같이 제시되었다[6][19].

<Table 2> Computer Education Curriculum in Estonia

Grade	Coding Curriculum
1st to 4th	1) First steps in programming - learning how to use the mouse and keyboard - playing games that involve logic and thinking 2) Graphic programming language - Kodu, Logo, and Scratch
5th to 9th	1) Lego Robots - From NXT-G to NXC 2) Web Programming - Website and web application
10th to 12th	1) Web Programming - Website and web application

핀란드는 명실상부한 교육선진국으로서 우리나라와 역사적 배경이 유사하고 인재 양성을 통한 국가 경쟁력을 확보하기 위해 교육에 집중적인 투자를 실시한 교육복지국가라고 할 수 있다. 핀란드의 경우 2016년부터 개정교육과정이 도입되면서 소프트웨어 교육을 실시하고 있으며, 다음의 <Table 3>과 같이 학년 군별로 세부 교육과정을 편성하여 운영하고 있다[8][9][18].

<Table 3> Computer Education Curriculum in Finland

Grades	How to study
1-2 Grade	Play for learning the strategies and solving the problems
3-6 Grade	Visual programming language as an educational programming language
7-9 Grade	Actual programming language

호주는 국가수준교육과정으로 소프트웨어 교육을 필수정규교육과정으로 반영할 것을 2015년 결정하였고, 학년 군별로 아래의 <Table 4>와 같이 교육과정을 편성하여 운영하고 있다[1][15][17].

<Table 4> Computer Education Curriculum in Australia

Grades	Programming Resources
F-2 Grade	Robotic Devices
3-4 Grade	Visual Programming Languages
5-6 Grade	Visual Programming Languages
7-8 Grade	General-purpose Programming Language
9-10 Grade	Object-oriented Programming Language

이와 같이 소프트웨어 교육을 운영하는 해외의 국가들은 교육과정의 편성단계에서 학년 혹은 학년군별로 구체적인 기준과 소프트웨어 교육을 위한 도구의 범위를 제시하고 있다. 해외의 교육과정을 분석하여 우리나라 소프트웨어 교육과정에 대한 시사점 분석 연구를 살펴보면 프로그래밍 언어의 선택 기준 및 학년 군에 따른 위계 구성 등에 대한 연구는 진행되어 있으나, Computational Thinking의 개념 정립과 사고과정에 대한 활동에 초점을 두고 위계를 구성하여 교육과정을 편성한 연구가 이루어질 필요가 있음을 살펴볼 수 있다[2][17].

2.2 우리나라의 소프트웨어 교육과정

소프트웨어 교육이 목표가 문제해결력을 기르기 위한 사고과정에 초점을 두고 있음을 고려하더라도 컴퓨터에서 이루어지는 절차적 사고를 기르기 위한 핵심 활동이 프로그래밍 활동이라는 점을 고려하였을 때 해외의 교육과정에 편성된 프로그래밍 언어의 선택 기준과 이를 토대로 수행되어야 하는 활동이 명확하게 제시되어야 한다.

교육과정의 편성과 실제가 국가수준의 교육과정을 토대로 이루어진다는 점을 고려하여 '2015개정교육과정'에 제시되어 있는 교육과정 편성과 기준을 살펴보면 [9정04-01]의 성취기준 해설과 교수학습방법 및 유의사항에서 아래와 같이 프로그래밍 언어의 선택 기준과 활동이 제시하고 있다[11].

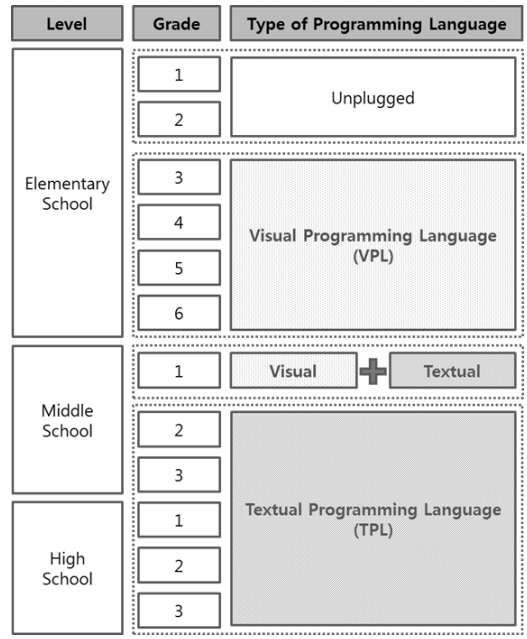
사용할 프로그래밍 언어의 개발 환경 및 특성을 이해한다(p.229)

학습자 수준에 적절한 교육용 프로그래밍 언어를 선택한다(p.230)

특정 프로그래밍 언어의 기능 습득에 치중하지 않도록 유의하고 문제 해결을 위한 프로그램 설계 및 개발 과정을 통해 컴퓨팅 사고력을 신장하는데 초점을 둔다(p.230)

컴퓨팅 사고력을 신장시키기 위한 활동으로 프로그래밍 언어를 활용하도록 안내가 되어 있고, 학습자의 수준에 따라 적절한 교육용 프로그래밍 언어를 선택하도록 제시되어 있으나, 학습자의 발달단계에 따른 구체적인 기준이 마련될 필요가 있으며, 교육용 프로그래밍 언어의 종류에 따른 가이드라인이 제시되는 것이 개정교육과정의 적용에 따른 안정적인 시행이 이루어질 것이다.

이에 따라, 신승기, 배영권(2015)는 소프트웨어 교육과정 운영을 위하여 다음 [Fig. 1]과 같이 학교 급별 프로그래밍 언어와 교육과정의 편성 기준을 제시하고 있다. 초등학교 저학년에서는 언플러그드 활동을 실시하고 중학년과 고학년에서는 비주얼 기반의 프로그래밍 언어를 활용하도록 하였다[17]. 중등과정에서는 텍스트 기반의 프로그래밍 언어를 활용하되, 중학교 1학년 과정에서 전이(Transfer)가 이루어질 수 있도록 비주얼 기반의 프로그래밍 언어와 텍스트 기반의 프로그래밍 언어를 함께 학습하도록 구성하고 있다[17].



(Fig. 1) Suggestion for Curriculum of Korea

소프트웨어 교육의 핵심 사고전략이 컴퓨팅 사고력의 향상이라는 점을 고려하였을 때, 우리나라의 교육과정에

서는 컴퓨팅 사고력 향상을 위한 세부 활동의 제시가 명확하지 않다. 컴퓨팅 사고력에 대한 정의와 주요 요소 등에 대한 내용들은 제시가 되어 있으나, 구체적인 교수학습모형과 컴퓨팅 사고력 향상을 위한 활동이 보완될 필요가 있다[17].

배영권, 신승기(2017)은 Computational Thinking의 해외 적용 사례 분석을 통해 국가별 교육 환경과 철학적인 배경에 따라 개념 정립과 세부 활동 및 교육과정 편성이 달랐음을 제시하면서 '절차적으로 제시된 일련의 사고 과정'이라는 용어와 '알고리즘을 기반으로 문제해결방법을 체계적으로 형성해가는 절차'에 대한 내용이 반영되어야 함을 명시하고 있다[2].

3. 연구 목적 및 연구 방법

소프트웨어 교육과 관련하여 2015개정교육과정이 중학교는 금년부터 적용되고, 초등학교는 2019년부터 적용된다는 점을 고려하여 실제 학교 현장에서 활용하기 위한 구체적인 교수학습방법에 대한 논의와 추후 교육과정을 보완하고 개선하기 위한 방법과 내용을 도출하기 위한 노력을 기울이고 있다. 미국에서는 기존의 CSTA와 ISTE에서 2011년 제시한 Computational Thinking에 대한 개념[4]을 토대로 문제해결력 중심의 Computational Thinking을 향상시킬 수 있는 교육과정의 편성을 완료하여 미국 전역으로 적용하기 위한 노력을 기울이고 있다는 점을 고려하였을 때 우리나라의 소프트웨어 교육에 대한 시사점을 도출할 수 있을 것이다. 특히 Computational Thinking에 대한 지속적인 논의와 개념정립 및 교육과정 편성에 대한 연구가 이루어지고 있음을 고려하여 Computational Thinking을 향상시키기 위한 세부적인 활동 및 학년별 성취 기준 등에 대한 내용에 대한 분석을 통해 우리나라에서 적용 가능한 시사점을 살펴볼 수 있을 것이다.

본 연구에서는 미국의 사실상 국가수준의 소프트웨어 교육과정이라고 할 수 있는 컴퓨터교육 연합체(K-12 Computer Science)에서 개발한 소프트웨어 교육과정의 체계와 세부 내용[7]을 살펴보고 Computational Thinking의 개념정립과 관련 활동에 대한 내용들의 분석을 통해 우리나라 교육과정에서 활용가능한 시사점을 도출하고 학교 급별 학년 및 학년 군에 따른 구체적인 교육과정 편성 안을 제시함으로서 발전적인 소프트웨어 교육의 제도적인 개선이 이루어질 수 있는 방안을 모색하고자 한다.

4. 미국의 컴퓨터교육 프레임워크 분석

기존의 미국에서 진행된 컴퓨터교육에 대한 교육과정 관련 연구는 CSTA와 ISTE가 주도하여 교수학습모형개발 및 활동과 단계에 대한 내용으로 이루어졌으나[4], 교육청 및 학교별 소프트웨어 교육의 학년별 체계와 성취기준 등에 대한 구체적인 교수학습내용과 편성에 대한 요구가 있어왔다. 이에 따라 CSTA, ACM, Cyber Innovation Center, National MATH+SCIENCE Initiative가 연합체를 구성하여 유치원에서 고등학교에 이르는 전체 단계에서 Computational Thinking을 향상시킬 수 있는 구체적인 활동과 성취기준을 완성하여 제시하였다[7]. 아래의 <Table 5>는 5개의 핵심 영역(Core Concepts)과 7개의 주요 활동(Core Practices)로 구성된 미국의 컴퓨터교육 프레임워크이다[7].

<Table 5> The K-12 Computer Science Framework in U.S.

Cores	Factors
Core Concepts	Computing Systems
	Networks and the Internet
	Data and Analysis
	Algorithms and Programming
	Impacts of Computing
Core Practices	Fostering an Inclusive Computing Culture
	Collaborating Around Computing
	Recognizing and Defining Computational Problems
	Developing and Using Abstractions
	Creating Computational Artifacts
	Testing and Refining Computational Artifacts
	Communicating About Computing

4.1 Computational Thinking의 개념과 주요 활동

K-12 Computer Science(K12CS)에서는 컴퓨터교육과정을 편성하면서 Computational Thinking을 향상시킬 수 있는 활동과 교육과정 편성을 구성하기 위하여 개념을 먼저

정립하고 있다. 아래의 내용(K12CS(2015), p.68)은 K12CS에서 인용하여 제시하고 있는 Computational Thinking의 개념으로서 미국의 컴퓨터 교육과정 편성에 필요한 요소들을 이끌어내기 위해 안내하고 있다[7].

Computational thinking refers to the thought processes involved in expressing solutions as computational steps or algorithms that can be carried out by a computer (Cuny, Snyder, & Wing, 2010; Aho, 2011; Lee, 2016).

컴퓨팅 사고력은 컴퓨터에 의해 수행될 수 있는 알고리즘 혹은 계산적 절차로서 표현될 수 있는 문제해결방법과 관련된 사고 절차를 의미한다 (Cuny, Snyder, & Wing, 2010; Aho, 2011; Lee, 2016).

K12CS(2015)에서는 Cuny, Snyder & Wing(2010)의 인용을 통해 Computational Thinking의 정의는 정보처리도구로서 수행될 수 있는 형태의 문제와 해결방법으로 구성되는 사고과정이라고 제시하였고, Aho(2011)와 Lee(2016)의 인용을 통해 컴퓨터로 수행될 수 있는 계산적인 과정(Computational Steps)과 알고리즘의 구체적인 형태로 문제해결과정이 제시되어야 한다는 것을 소개하고 있다[7]. 이와 같이 Computational Thinking은 모든 문제를 해결하기 위한 사고과정이라기 보다는 컴퓨터를 활용하여 해결될 수 있는 계산적 문제 상황(Computational Problem Situation)에서 활용될 수 있는 컴퓨터로 수행 가능한 문제해결과정이라는 것을 의미한다.

또한 Wing(2008, p. 3718)의 개념을 소개하면서 기본적으로 인간이 문제를 해결하는 과정이기 때문에 언플러그드 활동이 가능하다는 점을 제시하였고[7][23], Bundy(2007)의 인용을 통해 다른 교과에서도 융합할 수 있음을 안내하고 있다[3][7].

K12CS(2015, p.68)의 컴퓨터교육 교육과정에서 제시된 프레임워크에는 Computational Thinking의 핵심 개념을 기르기 위한 활동으로 다음 [Fig. 2]와 같이 세부내용을 제시하고 있다[7].

(Fig. 2) Core practices for computational thinking

컴퓨터교육의 교육과정에 편성된 성취기준으로 [Fig. 2]의 핵심 활동을 중심으로 다음과 같은 7가지의 내용을 제시하고 있으며 12학년을 마치는 초-중-고의 교육과정 전체에서 달성해야할 목표로 수립되어 있다[7].

첫째, 포괄적인 컴퓨팅 문화조성이다. 이를 위해 학생들은 computational products를 설계하고 개발하는 과정에서 자신과 다른 사람의 관점을 반영할 수 있도록 한다.

둘째, 컴퓨팅의 협업이다. 다양한 개인과의 관계를 형성하고 함께 문제를 해결하는 과정에서 상호 피드백을 제공하고 팀의 효율성을 높이기 위해 공평한 과업의 분배 및 규범 제정 등의 능력을 갖도록 한다.

셋째, Computational problems의 인식과 정의이다. 계산적으로(computationally) 해결할 수 있는 문제를 발견하도록 하며, 실생활의 복잡한 문제를 해결할 수 있는 형태로 분해하여 계산적으로 해결하는 것이 적절하고 실현가능 한지를 판단할 수 있도록 한다.

넷째, 추상화 능력의 개별과 활용이다. 문제 상황에서 공통적으로 발견되는 일반적인 특징을 추출하여 문제해결을 위한 아이디어 설계과정에 기존의 공학적 기능들을 활용할 수 있는지를 판단하고 다양한 문제 상황에 적용하여 복잡성을 줄일 수 있는 문제해결의 모듈을 만들 수 있도록 한다.

다섯째, 계산적 산출물(Computational artifacts) 만들기이다. 반복적인 절차를 활용하여 문제해결의 산출물

을 제작하기 위한 계획의 단계로서 실용성을 고려하고 개인 및 사회적 문제를 해결하기 위한 산출물을 만들거나 기존의 산출물을 수정하거나 향상시키기 위한 과정을 수행한다.

여섯째, 계산적 산출물(computational artifacts)의 테스트와 개선하기이다. 문제해결과정의 모든 상황을 고려하여 산출물을 체계적으로 테스트하여 오류를 보완하여 신뢰성과 유용성 등을 향상한다.

일곱째, 컴퓨팅에 관해 의사소통하기 이다. 컴퓨팅을 통해 계산적 절차와 해결방법에 대해 적절한 용어를 활용하여 설명하고 저작권을 고려하여 문제해결의 적절한 아이디어를 제시할 수 있도록 한다.

4.2 Computing과 Computational Thinking

미국의 컴퓨터교육 교육과정에서는 컴퓨터 교육을 Computing Education이라는 용어로 제시하고 있다. 기존의 Computing Education은 컴퓨터 활용 능력, 컴퓨터 활용 교수법, 디지털 시민성, 정보공학, 컴퓨터 과학을 기를 수 있도록 편성되어 있으나 컴퓨터를 활용하는 측면에 초점을 두고 있기 때문에 컴퓨터과학과는 구별되는 차이점임을 제시하고 있다[7]. 즉, 원리를 이해(Knowing why)하고 컴퓨터의 작동 원리를 아는 것(How computers work)이 궁극적으로 학생들의 문제해결과정에서 의사결정의 기준수립에 도움을 준다고 언급하고 있으며 아래(K12CS(2015), p.14)와 같이 제시하고 있다[7].

Computer science is the foundation for computing. The framework envisions a future in which being computer literate means knowing computer science.

컴퓨터과학(Computer Science)은 컴퓨팅(Computing)의 기초가 되며, 미래에는 컴퓨터 소양(Computer literate)을 갖춘다는 의미에 대해 컴퓨터과학을 이해한다는 것을 나타낼 것이다.

4.3 컴퓨터교육 중심의 융합교육 프레임워크

K12CS(2015, p.71)는 컴퓨터교육의 영역에 대하여 아래의 Wing(2006)의 내용을 인용하여 모든 교과에 확대해야함을 제시하고 있다[7].

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability (Wing, 2006, p. 33).

Computational thinking은 모든 사람들에게 필요한 기초적인 역량으로서, 읽고 쓰고 셈하는 과정에 computational thinking을 추가하여 모든 학생들이 분석적인 능력을 기를 수 있도록 해야 한다.

이를 위해 <Table 6>와 같이 컴퓨터과학과 수학 및 과학 교과의의 융합방안을 제시하고 있다[7].

<Table 6> Integration method for computer science

Relationship	Integration
CS +Math	· Develop and use abstraction · Use tools when collaborating · Communicate precisely
CS +Math +Sci/Eng	· Model · Define problems · Use computational thinking · Communicate rationale
CS +Sci/Eng	· Communicate with data · Create artifacts

4.4 학교 급별 컴퓨터교육 교육과정 편성 체계

미국의 컴퓨터교육 교육과정에서는 5가지의 핵심 영역을 도입하여 학년별로 이수해야하는 기준을 제시하고 있다. 5가지의 핵심 영역(Core concepts of the framework)로는 컴퓨팅 시스템(Computing Systems), 네트워크와 인터넷(Networks and the Internet), 데이터와 분석(Data and Analysis), 알고리즘과 프로그래밍(Algorithms and Programming), 컴퓨팅의 효과성(Impacts of Computing)으로 제시되어 있다[7]. 각각의 영역은 융합의 개념도 함께 제시하여 다른 영역 혹은 다른 교과와 함께 가르치게 될 경우 포함할 수 있는 5가지의 영역을 안내하고 있다.

융합을 위한 5가지 영역으로는 추상화(Abstraction), 체제의 관련성(System Relationships), 인간과 컴퓨터의 관계(Human-Computer Interaction), 개인정보와 보안(Privacy and Security), 소통과 협력(Communication and Coordination)으로 구성되어 있다[7].

학년별로 이수해야 하는 5가지 핵심 영역에 대한 세부 내용은 다음의 <Table 7>과 같이 안내되어 있다[7].

<Table 7> Core Concepts of the Framework

Concepts	Factors
Computing System	Devices
	Hardware and Software
	Troubleshooting
Networks and the Internet	Network Communication and Organization
	Cybersecurity
Data and Analysis	Collection
	Storage
	Visualization and Transformation
	Inference and Models
Algorithms and Programming	Algorithms
	Variables
	Control
	Modularity
	Program Development
Impacts of Computing	Culture
	Social Interactions
	Safety, Law, and Ethics

5. 우리나라 소프트웨어 교육과정 편성의 시사점

5.1 Computational Thinking의 개념정립

우리나라의 소프트웨어 교육과정과 관련하여 Computational Thinking은 컴퓨팅 사고력으로 소개되어 있으며, 2015개정교육과정의 정보교과해설서(p.96)에서 다음과 같은 정의로 제시되어 있다[11].

컴퓨팅 사고력은 컴퓨터과학의 기본 개념과 원리 및 컴퓨팅 시스템을 활용하여 실생활과 다양한 학문 분야의 문제를 이해하고 창의적으로 해법을 구현하여 적용할 수 있는 능력을 말한다.

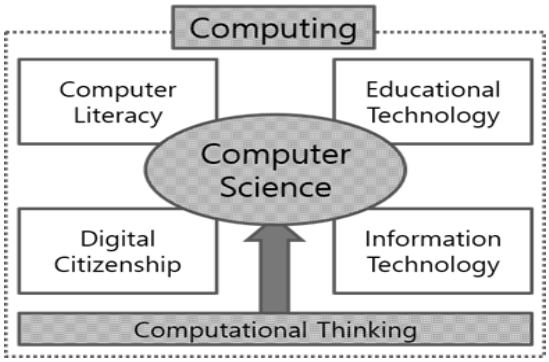
Computational Thinking의 개념에 대하여 살펴보기 위해 현재 우리나라에서 사용하고 있는 컴퓨팅 사고력이라는 용어를 Computing Thinking으로써 고려한다면, 기존의 Computational Thinking의 용어와는 의미하는 바가 다를 수 있음을 살펴볼 필요가 있다.

Computational은 계산적이라는 의미를 갖는 단어로서 계산적 절차(Computational Process)를 통해 컴퓨터에 의해 수행될 수 있는 과정을 의미한다[7]. 따라서 Computational Thinking은 계산적 사고라는 의미를 갖는다고 할 수 있으며, 컴퓨터(Computer)가 자동화 계산 도구라는 점에서 전산(電算)의 의미를 갖는다는 것을 고려한다면 컴퓨터로 수행할 수 있는 사고 전략 및 과정이라고 할 수 있다.

이와 함께 컴퓨팅(Computing)은 컴퓨터 활용 능력, 컴퓨터 활용 교수법, 디지털 시민성, 정보공학, 컴퓨터 과학의 내용을 포함하는 개념으로 컴퓨터의 활용에 보다 초점을 두고 있는 개념이다[7].

우리나라의 소프트웨어 교육과정에서 창의적 문제해결력을 신장시키기 위한 사고과정에 초점을 두는 것[2]은 컴퓨터 과학의 개념에 가깝다는 것으로 이해할 수 있고, 원리를 이해하는 과정(Knowing why)과 컴퓨터의 작동원리에 대한 이해(How computers work)를 통해 소프트웨어 및 컴퓨터의 원리를 토대로 미래사회의 인재를 길러낸다는 측면[7]에서 컴퓨터 과학의 개념을 의미한다.

따라서 Computational Thinking은 컴퓨터 과학에서 원리를 이해하고 문제해결을 위한 사고과정이라는 측면이라는 점에서 Computing에 포함되는 요소 중 하나라고 볼 수 있다. 반면, 우리나라의 소프트웨어 교육과정에서 제시하는 '컴퓨팅 사고력'이라는 용어는 Computational Thinking의 의미에서 확대된 개념이라고 판단될 수 있으며, [Fig. 3]에서 제시한 개념관계와 같이 컴퓨터과학의 기저 사고과정이 아니라 범용적인 용어로 여겨질 수 있다. 이는 Computational Thinking은 Computing의 근간을 이루는 문제해결, 시스템 설계, 인간 행동의 이해를 위한 접근법이라는 관점(Wing, 2008; 2006)과 일맥상통한다[22][23].



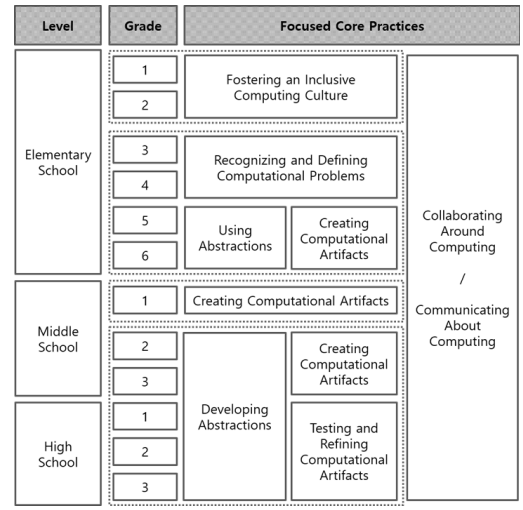
(Fig. 3) Relationship between computing and computational thinking

해외의 교육과정 사례를 살펴보면 영국과 미국의 소프트웨어 교육에서는 Computational Thinking은 사고전략으로서의 개념을 의미하고 있으며, 컴퓨팅(Computing)은 컴퓨터를 활용한 학생들의 교수학습 활동에서 용어를 사용하는 것을 살펴볼 수 있다[5][17]. 중국에서는 Computational Thinking을 계산사유(計算思維)라고 사용하고 있으며[21], 일본에서는 계산론적사고(計算論的思考)라는 용어를 사용하는 것을 고려한다면[24] 우리나라에서 사용하는 컴퓨팅 사고력에 대한 용어에 대한 명칭은 추후 검토하거나 용어의 정의를 Computational Thinking에서 확대하여 새롭게 접근해야할 필요성이 있다고 할 수 있다.

5.2 위계를 고려한 Computational Thinking 중심의 활동을 반영한 컴퓨터 교육과정 편성

미국의 컴퓨터 교육 프레임워크 개발을 위해 구성된 K12CS는 Computational Thinking을 향상시키기 위한 방법으로 학생들이 초-중-고의 과정에서 학습해야할 핵심 개념(Core Concepts)과 핵심 역량(Core Practices)를 제시하고 있다[7]. 특히 핵심 역량에 대하여 초등학교 2학년, 5학년에서 반드시 익혀야할 내용을 제시하고 있고, 중학교와 고등학교 과정의 각각에서 성취해야할 기준을 제시하고 있다[7]. 우리나라의 정보교육과정과 미국의 K12CS의 교육과정 프레임워크를 비교해본다면, 미국의 경우에는 학교급과 학년 및 학년군을 고려하여 성취 기준을 핵심 역량과 개념을 중심으로 위계를 구성하고 있는 반면 우리나라의 경우에는 핵심 개념에 따른 성취 기준은 제시하고 있으나 학교급과 학년에 따른 핵심 역량의 위계는 제시하고 있지 않다.

신승기, 배영권(2015)의 학교 급별 위계를 고려한 프로그래밍 언어 선정 기준인 [Fig. 1][17]을 토대로 미국의 K12CS에서 제시한 학년별 성취기준[7]을 우리나라의 정보교육과정의 관련성을 고려하여 학습자의 인지발달단계를 기반으로 구성한 학년(군)에 따라 제시하면 [Fig. 3]과 같이 나타낼 수 있다.



(Fig. 4) Focused Core Practices by Grade

6. 결론 및 제언

본 연구에서는 우리나라의 소프트웨어 교육과정 편성과 Computational Thinking에 대한 개념정립을 위하여 미국의 K12CS에서 제시한 컴퓨터교육 프레임워크를 분석함으로써 시사점을 도출하고자 하였다. 우리나라의 소프트웨어 교육은 2018년부터 중학교 과정에 34차시 분량으로 필수독립교과로 반영이 되었고, 2019년부터 초등학교 5~6학년에 17차시의 내용으로 실과교과에 내용이 편성되어 운영이 된다. 소프트웨어 교육 운영을 위해 국가수준에서 제시된 교육과정을 살펴보면 다른 나라와는 달리 학년별 위계에 대한 내용이 반영되어 있지 않으며, 프로그래밍 언어 선정의 기준 및 Computational Thinking에 대한 개념 정립이 다소 부족한 편이다.

핀란드, 에스토니아, 영국, 호주, 인도 등 소프트웨어 교육을 필수교육과정으로 운영하는 나라의 경우에는 학년별로 위계를 갖추고 학습자의 인지발달 단계에 따라 수준을 달리하여 세부적으로 교육과정이 편성되어 있다[17]. 미국의 경우 기존의 CSTA&ISTE(2011)에서 제시한 Computational Thinking의 교수학습모형을 구체화하여 학년별 성취기준 및 핵심 역량을 제시함으로써 체계를 갖추어 완성된 교육과정을 K12CS(2015)를 통해 제시하고 있다[4][7].

Computational Thinking의 개념이 미국에서 처음 시작되어 전 세계로 확산되었다는 것을 고려할 때 미국의 교육과정을 분석함으로써 우리나라의 소프트웨어 교육과정을 보완할 수 있는 방안을 모색하고 제언하고자 하였다.

첫 번째로 우리나라에서 Computational Thinking에 대한 용어를 컴퓨팅 사고력으로 사용하는 부분에 대해서 미국의 교육과정에 제시된 개념과 비교하여 살펴보았다. 컴퓨팅(Computing)은 문제 해결의 사고과정 보다는 컴퓨터를 활용하는 것에 초점을 두고 있는 광범위한 개념으로서 하위 요소로서 컴퓨터 과학(Computer Science)를 두고 있는 의미를 갖고 있음을 알 수 있고, 컴퓨터 과학의 기저 사고 전략이 Computational Thinking이라는 것을 고려할 때 컴퓨팅 사고력과 Computational Thinking은 문제 해결의 초점과 범위가 다름을 살펴볼 수 있었다. 또한 인접 국가 중에서 비슷한 언어적인 문화권을 갖고 있는 중국과 일본에서는 계산사유 혹은 계산론적사고라는 용어를 사용한다는 점에서 현재 우리나라의 소프트웨어 교육과정에서 사용 중인 Computational Thinking의 개념은 본래의 의미와는 다르게 해석될 여지가 있다. 아울러, Computational Thinking은 사고과정이라는 점을 고려할 때. 컴퓨팅 사고“력”은 사고과정이 아니라 사고능력을 의미한다는 점에서 Computational Thinking을 컴퓨팅 사고력으로 대치하기에는 의미의 범위에 대한 차이가 있다고 할 수 있다. 따라서 추후 교육과정 개편 및 교과교육론 연구에서 용어의 사용에 대한 고민이 필요하며 범위와 개념에 대한 정립이 필요하다는 것을 살펴볼 수 있었다.

두 번째로, 우리나라의 소프트웨어 교육과정은 학교급 및 학년에 따른 위계가 고려되지 않는다는 점을 고려하여 미국의 K12CS에서 제시하고 있는 핵심 역량을 토대로 재구성하여 교육과정 체계를 제시하였다. 독립교과로서의 정보교과는 현재 중학교에만 편성이 되어 있고, 초등학교의 경우 실과교과의 한 개 영역으로 17차시로만 반영됨에 따라 교과간의 연계 및 위계를 고려한 교육과정 편성에서 제한이 나타난다. 이는 초등학교에서의 소프트웨어 교육에 대한 독립교과로서의 편성이 요구됨을 살펴볼 수 있을 뿐만 아니라 체계적인 소프트웨어 교육의 커리큘럼을 구성할 수 있는 기준이 된다는 점을 고려하게 한다.

다가오는 미래사회에서는 창의적 문제해결력을 가진 인재를 필요로 하고 있다. 이는 소프트웨어 교육에서 추구하는 컴퓨팅 사고력 기반의 문제해결력을 지닌 창의적 인재를 양성한다는 목표와 일치한다. 소프트웨어 교육은 현재 전 세계적으로 개념과 교육과정 편성에 대한 논의가 지속적으로 이루어지고 있으나, 공통점은 Computational Thinking기반의 위계를 고려한 학년단위의 교육과정이 편성되어 있다는 점이다. 정답이 있다고 할 수는 없으나, 학습자의 발달단계와 학년성을 고려한 교육과정이 구체적으로 제시된다면 창의적인 문제해결력을 지닌 인재양성에 기여할 수 있을 것이며, 향후 교육과정 관련 연구 및 개정 과정에 반영되어야 할 것이다.

참고문헌

[1] Australian Curriculum(2015). Curriculum of Digital Technologies. Retrieved from http://www.australiancurriculum.edu.au/technologies/digital-technologies/curriculum/f-10?layout=1

[2] Bae. Y., Shin, S. (2017, August). The Study on Analysis of the Implications for the Software Education through Case Studies of Applications in the World of Computational Thinking. *KAIE Research Journal. Vol 8*, No. 2. pp.143-156.

[3] Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing, 1*(2), 67-69.

[4] Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE) (2011). Computational Thinking

Teacher Resources. Second Edition.

[5] Department for Education in U.K (2014, June). Interim KS5 minimum standards. Retrieved from https://www.gov.uk/government/publications/interim-ks5-minimum-standards

[6] Innovation Centre in HITSA. Programming at Schools and Hobby Clubs. Retrieved from http://www.innovatsioonikeskus.ee/en/programming-schools-and-hobby-clubs

[7] K-12 Computer Science Framework. (2016). Retrieved from http://www.k12cs.org.

[8] Khan Academy(2015). Computer programming. Retrieved from https://www.khanacademy.org/computing/computer-programming

[9] Koodi2016(2014). Koodi2016-ensiapua ohjelmoinnin opettamiseen peruskoulussa. Retrieved from https://s3-eu-west-1.amazonaws.com/koodi2016/Koodi2016_LR.pdf

[10] Ministry of Education, Korea (2015). Elementary School Curriculum. #2015-74 (Annex 2).

[11] Ministry of Education, Korea (2015). Informatics Curriculum. #2015-74 (Annex 10).

[12] Ministry of Eduction, Korea (2015). Software Education Instructional Guidance.

[13] Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc.

[14] Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning, 1*(1), 95-123.

[15] Sadler, D. (2015, September). Tech sector cheers on new PM Malcolm Turnbull. startup smart. Retrieved from http://www.startupsmart.com.au/planning/tech-sector-cheers-on-new-pm-malcolm-turnbull/2015091515517.html

[16] Shin, S., Bae, Y. (2014, December). Analysis and Implication about Elementary Computer Education in India. *Journal of The Korean Association of Information Education. Vol 18*, No. 4. pp. 585-594. doi: http://dx.doi.org/10.14352/jkaie.2014.18.4.585

[17] Shin, S., Bae, Y. (2015, December). A Study on the Hierarchical Instructional System Design of Software Education by School System. *Journal of The Korean Association of Information Education. Vol 19*, No. 4. pp. 533-544. doi: http://dx.doi.org/10.14352/jkaie.2015.19.4.533

[18] Shin, S., Bae, Y. (2015, March). Review of Software Education based on the Coding in Finland. *Journal of The Korean Association of Information Education. Vol 19*, No. 1. pp. 127-138. doi: http://dx.doi.org/10.14352/jkaie.2015.19.1.127

[19] Shin, S., Bae, Y. (2015, September). Study on the Implications about Curriculum Design through the Analysis of Software Education Policy in Estonia. *Journal of The Korean Association of Information Education. Vol 19*, No. 3. pp. 361-372. doi: http://dx.doi.org/10.14352/jkaie.2015.19.3.361

[20] Sridhar Iyer, Farida Khan, Sahana Murthy(2013). CMC: A Model Computer Science Curriculum for K-12 Schools. 3rd Edition. Indian Institute of Technology Bombay, Mumbai. Retrieved from http://www.cse.iitb.ac.in/internal/techreports/reports/TR-CSE-2013-52.pdf

[21] Wang, R. L. (2014). Computational thinking education(Chinese Edition). Shanghai Science and Technology Education Press.

[22] Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

[23] Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences, 366(1881), 3717-3725.

[24] Wing, J. M. (2015). Computational Thinking 計算論的思考. 情報處理, 56(6), 584-587. Retrieved from https://ipsj.ixsq.nii.ac.jp/ej/?action=repository_action_common_download&item_id=141823&item_no=1&attribute_id=1&file_no=3

저자소개

신 승 기

2007 한국교원대학교 컴퓨터교육과(교육학사)

2009 아주대학교 정보통신대학원(공학석사)

2012 대구교육대학교 컴퓨터교육과(교육학석사)

2017년 University of Georgia, Ph.D.

관심분야: 소프트웨어교육, Computational Thinking

e-mail: shin@uga.edu

배 영 권

2006 한국교원대학교 컴퓨터교육과(교육학박사)

2006~2007 Indiana University VisitingScholar

2007~2009 목원대학교 컴퓨터교육과 교수

2013~2014 University of Georgia, VisitingScholar

2009~현재 대구교육대학교 컴퓨터교육과 교수

관심분야: 소프트웨어교육, STEM교육, 정보영재교육

e-mail: bae@dnue.ac.kr