문제해결의 관점에서 컴퓨팅 사고력 증진을 위한 교수학습에 대한 연구

최숙영†

요 약

컴퓨팅 사고력은 컴퓨터를 기반으로 하는 문제해결에 관한 하나의 사고 과정으로 정의될 수 있기 때문에 일반적인 문제해결의 과정에서 관련된 컴퓨팅 사고력 개념들을 살펴보는 것은 컴퓨팅 사고력을 이해하는데 도움이 될 수 있다. 이를 위해 본 연구에서는 관련 연구를 통해 컴퓨팅 사고력의 핵심 요소들을 뽑아내고 그러한 요소들이 문제 해결과정에서 어떻게 사용되는지를 기술한다. 또한 문제해결과정에서의 컴퓨팅 사고력 요소들을 인지적인 측면에서 살펴본 후, 각 단계에서의 학습활동과 학습평가 요소들을 기술한다. 이를 기초로 문제 해결의 관점에서 컴퓨팅 사고력을 위한 기본 수업 프레임워크를 제안한다.

주제어: 컴퓨팅 사고력, 문제 해결, 수업 프레임워크

A Study on Teaching-learning for Enhancing Computational Thinking Skill in terms of Problem Solving

Sook Young Choi†

ABSTRACT

This study aims to suggest an instructional design to improve CT(Computational Thinking) skills in terms of problem solving. CT can be defined as a thought processes for computer-based problem solving. Examining the related CT concepts in the general problem solving process can be helpful for learners to understand CT. For this, this study selects the key elements of CT through literature review, describes how the elements are related to each phrase of the problem solving process, and explores cognitive aspects of the CT elements. In addition, this study describes learning activities and learning assessments of the CT elements according to each phrase of problem solving process and suggests a basic instructional design framework for CT in view of problem solving.

Keyword: Computational thinking, Problem solving, Instructional design framework

† 정 회 원: 우석대학교 정보보안학과 교수
논문접수: 2015년 12월 8일, 심사완료: 2016년 1월 17일, 게재확정: 2016년 1월 27일

1. 서 론

컴퓨팅 사고력(CT:Computational Thinking)을 3R(읽기, 쓰기, 셈하기)과 더불어 모든 학습자가 갖추어야할 기본적인 학습 능력으로 보아야 한다는 의견이 Wing 교수에 의해 제시된 이후[1], 전 세계적으로 컴퓨팅 교육에 대한 많은 관심과 논의가 있어왔다. 이는 미래의 모든 과학과 공학에 관련된 분야가 컴퓨팅의 영향력 아래에 놓여 있기 때문에 컴퓨팅 사고력을 갖춘 창의적인 인재 양성이 필요하다는 인식에서 기인된다. 이에 따라 모든 초중고 학생들이 컴퓨터 과학의 경험을 통해 CT를 배워야 한다는 의견에 합의가 이루어지고 있다 [2].

이러한 추세와 함께 세계적으로 컴퓨터과학을 국가 교육과정으로 지정하여 학생들의 CT 역량을 높이기 위한 교육을 강화시키려는 노력이 최근 몇 년 사이에 많이 일어나고 있다. 국내에서도 2014년 9월 교육부가 2015년 문·이과 통합형 교육과정 총론을 발표하면서 소프트웨어 교육을 강화하는 내용을 포함시켰다. 하지만 CT 역량을 높이기 위해서 무엇을 어떻게 교육시킬 것인가에 대해서는 각 나라와 각 기관에 따라 다양한 의견들이 제시되고 있다. 그것의 기저에는 CT에 대한 정의가 아직 확립되지 않았다는 점이 존재한다.

미국의 경우 CT 역량을 높이기 위한 교육으로 Scratch나 Alice와 같은 언어를 이용한 코딩교육을 통해서 CT 개념을 교육시키기 위한 연구들이 진행되어왔다 [3][4]. 영국에서는 2014년 9월부터 ‘Computing’을 국가 교육과정으로 지정하여 초·중·고에서 이를 필수과목으로 이수하였다. 영국의 경우 새로운 컴퓨팅 교육이 ‘코딩’ 교육에만 너무 강조되는 것에 대한 우려를 나타내는 목소리들이 나왔다. 최근의 발표된 연구에 의하면 교사들이 폭넓은 관점에서 컴퓨팅 교육을 접근하기보다 프로그래밍이라는 협소한 측면으로만 접근하는 것에 대한 문제점들을 제시하고 있다[5]. 즉, CT를 프로그래밍을 통한 결과물을 산출하는 것에 초점을 맞추기보다 문제 해결을 위한 사고 과정(thought process)으로 보고, 학습자들이 문제 해결과 상황을 이해하기 위해 CT라는 특별한 전략을 사용하여 생각하도록 해야 한다는 관점이다.

모든 사람들이 프로그래밍에 능할 필요는 없다. 단지 사회전반의 모든 시스템과 서비스들이 컴퓨터기반으로 수행되고 있기 때문에 주어진 문제상황에서 컴퓨터를 이용하여 효과적으로 문제를 해결할 수 있는 능력을 갖추는 것이 필요할 것이다.

CT는 컴퓨터를 기반으로 한 문제해결에 관한 하나의 사고 과정으로 정의될 수 있기 때문에 일반적인 문제해결의 과정에서 CT를 살펴보는 것은 의미가 있다. 즉, CT를 이해하는 것뿐만 아니라 컴퓨터를 기반으로 하는 문제 해결과정을 이해하고 이를 바탕으로 문제해결력을 높일 수 있기 때문이다.

국내에서도 2018년부터 소프트웨어 교육을 강화시키는 내용이 발표되었다. 이를 효과적으로 수행하기 위해서는 다양한 측면에서의 많은 고민과 연구들이 이루어져야 할 것이다. 먼저, 소프트웨어 교육이 코딩 교육위주가 아닌 학생들의 CT 역량을 높이도록 하는 교육과정이 마련이 되어야 할 것이다. 또한 CT 역량을 높이기 위한 교육을 할 수 있도록 교사들을 준비시키고 적절히 일선 학교에 배치되도록 국가적인 노력이 우선적으로 필요하다.

본 연구는 교사들이 CT를 이해하고 이를 수업에 어떻게 적용할 것인지에 대해 도움을 주는 것을 목표로 한다. 이를 위해 본 연구에서는 문헌연구를 통해 CT의 핵심 요소들을 도출해내고 그러한 요소들이 문제 해결과정에서 어떻게 사용되는지를 분석하였다. 또한 그 요소들을 인지적인 측면에서 살펴본 후 관련되는 학습활동과 학습평가 요소들을 제안하였다. 이를 기초로 문제해결의 관점에서 CT 역량을 증진시키기 위한 기본 수업 프레임워크를 개발하였다.

2. 관련연구

2.1 CT 정의

CT에 대한 논의를 본격적으로 시작한 Wing은 CT 능력을 문제 해결을 위해 다양한 사고를 진행하고 그 과정에서 컴퓨팅 능력이나 컴퓨터과학 분야의 개념들을 활용하여 그 해결책을 컴퓨팅적으로 구현하는 전반적인 과정으로 보았다[1].

Wing이 CT에 대한 논의를 시작한 이후 여러 학자들이 CT에 대한 정의와 그 속성을 규명하기 위한 다양한 연구들을 진행하였다. CT의 의미와 속성에 관한 다양한 의견을 나누기 위한 워크숍이 National Research Council (NRC) 주관으로 2009년과 2010년에 개최되었다. 그 워크숍에서는 CT를 문제해결을 위한 정신적인 도구 및 문제해결과정을 나타내는 새로운 언어, 추상적인 모델을 자동화하는 것, 인지적인 도구 등의 다양한 범주로 정의하였다[6]. Wing은 2011년에 CT를 “문제를 컴퓨팅 시스템에서 효과적으로 수행되도록 문제와 그 해결책들을 표현하는 것에 관련된 사고 과정”고 폭넓게 정의하고 있다[7].

미국의 Computer Science Teachers Association (CSTA)와 International Society for Technology in Education(ISTE)는 K-12를 위한 CT의 조작적(Operational) 정의를 개발하기 위한 프로젝트를 수행하였다. 이 프로젝트의 위원회에서 CT를 “컴퓨터를 이용하여 수행할 수 있는 방법으로 문제를 해결하기 위한 하나의 기법”으로 보았다[8]. Aho (2012)는 CT를 문제를 해결가능한 형태로 표현하는데 관련되는 사고 과정으로 해결책은 컴퓨팅 단계와 알고리즘으로 표현될 수 있다고 하였다[9].

국내의 연구로 이영준외 5인의 연구[6]에서도 CT를 “컴퓨팅 사고력의 역량을 활용하여 해결하고자 하는 문제를 효과적이고 효율적으로 해결할 수 있는 절차적 능력”이라고 정의하고 있다.

2.2 CT 구성요소 및 특징

Wing은 CT의 핵심요소로 추상화와 자동화를 제안하였다[1]. 추상화는 실세계의 문제를 해결 가능한 형태로 표현하기 위한 사고과정이라고 할 수 있으며, 자동화는 추상화 과정에서 만들어진 해결 모델을 컴퓨터가 이해할 수 있는 프로그램으로 표현하여 수행 가능하도록 하는 것으로 보고 있다. 미국의 CSTA와 ISTE는 CT의 세부 구성 요소를 9가지로 구분하고 각 구성요소에 따라 각 학년별 수준에 맞는 예제를 제시하였다[10]. 9가지 구성 요소는 자료 수집, 자료 분석, 자료 표현, 문제 분해, 추상화, 알고리즘과 절차, 자동화, 시뮬레이션, 병렬화이다. 또한 CT를 <표 1>에서 기술된 ‘CT 특징’들을 포함하는 하나의 문제해결 과정으로 정의하고 있다[8].

Selby와 Wollard는 Wing의 CT에 대한 정의를 기반으로 하여 5개의 세부 개념을 도출하였다[5]. 그 세부 개념들은 알고리즘적 사고, 분해, 추상화, 평가, 일반화이다. 영국의 CAS(Computing At School)도 최근 이 세부 개념들을 가지고 교사들을 위한 CT 가이드를 제시하였다 [19].

한편, 미국의 Collage Board와 National Science Foundation은 고등학교의 Computer Science Principles 코스를 CT의 실제(practice)에 기초하여 개발하였다. CT 실제는 컴퓨팅으로 연결, 컴퓨팅 산출물을 개발, 추상화, 문제와 산출물을 분석, 의사소통, 협동의 6개로 구성된다[11]. <표 1>은 각 연구에서 제시된 구성요소들과 특징을 정리한 내용이다.

<표 1> CT 세부 구성요소 및 특징

CSTA & ISTE		Selby & Wollard	CB & NSF
CT 세부 구성요소	CT 특징	CT 세부 개념	CT 실제
• 자료 수집 • 자료 분석 • 자료 표현 • 문제 분해 • 추상화 • 알고리즘과 절차 • 자동화 • 시뮬레이션 • 병렬화	• 컴퓨터를 이용한 문제해결을 위해 문제 구성 • 논리적으로 자료 조직 및 분석 • 모델링이나 시뮬레이션을 이용한 추상화 • 가능한 해결책의 식별, 분석, 적용 • 알고리즘을 통한 해결책의 자동화 • 문제해결과정을 다양한 문제로 일반화 및 전환	• 알고리즘적 사고 • 분해 • 일반화(패턴) • 추상화 • 평가	• 컴퓨팅으로 연결 • 컴퓨팅 산출물 개발 • 추상화 • 문제와 산출물 분석 • 의사소통 • 협동

2.3 CT 평가 척도

CT 역량 교육이 효과적으로 이루어지기 위해서는 CT 역량을 측정하기 위한 평가 도구들이 요구된다. 평가 도구에 대한 연구로 Seiter와 Foreman는 PECT(Progression of Early Computational Thinking) 모형을 제안하였다[12]. 이 모형에서는 세 가지 측면(Evidence Variables, Design Pattern Variables, Computational Thinking Concepts)에서 세 수준 (Basic, Developing, Proficient)으로 구분하여 평가하였다. Evidence Variables은 스크래치로 작성된 코드의 세부요소들을, Design Pattern Variables는 스크래치의 일반적 코딩 전략에 기초한 맥락적 능숙도를, Design Pattern Variables는 CT의 전반적 수준을 체크하기 위한 것이다. Brennan과 Resnick의 연구에서는 CT 평가 프레임워크를 개발하였는데 Computational Concepts, Computational Practices, Computational Perspective의 3가지 차원으로 구분하고 있다[13]. Computational Concepts은 프로그램을 작성하면서 도입하는 스크래치의 프로그램 요소적 측면을, Computational Practices는 CT 개념의 도입과 프로그램 구성 및 개발의 과정을 체크하기 위한 측면을, Computational Perspective는 프로그램을 개발하면서 자신, 타인과의 관계, 그리고 주위의 기술적 세계에 대한 이해의 변화를 체크하기 위한 것이다. Werner와 그의 동료들은 Alice기반의 프로그래밍 환경에서 학습자들에게 기 작성된 코드를 제공하고 특정한 부분을 완성하도록 함으로써 CT 요소인 알고리즘적 사고, 추상화와 모델링을 평가하고 있다[14].

국내 연구로 김수환과 한선관[15]은 초등학생을 대상으로 CT 향상을 위하여 스크래치를 이용한 디자인기반 학습을 수행한 후 Computational Literacy 능력을 평가하였다. 이 연구에서는 Computational Literacy 능력을 정보적 해결력, self-프로그래밍 능력, self-프로그래밍 흥미도 등의 관점에서 평가하였다. 최형신[16]의 연구에서는 스크래치 프로그래밍 환경에서 CT 역량을 계발하기 위한 평가 루브릭을 개발하였다. 평가루브릭은 CSTA[10]에서 제시한 CT 요소들을 기초로 6가지 CT 역량 요소를 도출하였다. 6가지 요소는 절차 및 알고리즘, 병행화 및 동기화, 자료표현, 추상화, 문제분해, 시뮬레이션으로 구성된다. 이러한 세부 요소를 3단계(기초, 발달, 능숙)로 구분하여 평가척도를 제안하고 있다.

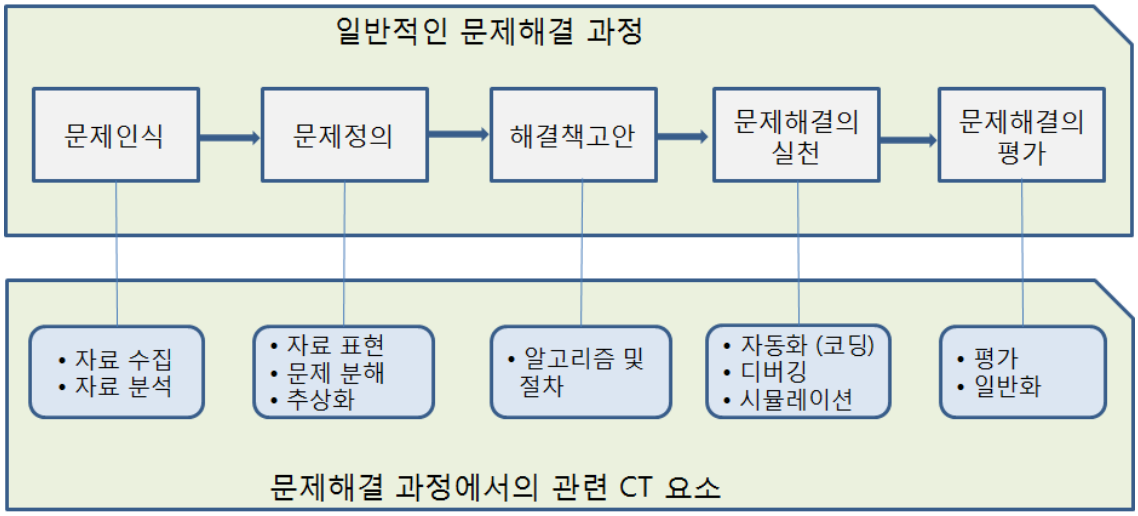
3. 일반적인 문제해결과정과 CT

CT는 “컴퓨팅을 활용한 문제해결을 위한 사고과정 혹은 기법”으로 정의될 수 있다. 본장에서는 일반적인 문제해결과정에서 CT의 세부 요소들이 어떻게 연결될 수 있는지를 살펴봄으로써 CT에 대한 이해를 돕고 CT를 교육시키기 위한 수업설계에 도움을 주도록 한다. 또한 CT가 다양한 분야의 문제들을 효과적으로 해결하는데 어떻게 사용될 수 있는 지에 대한 이해를 높이도록 한다.

3.1 문제해결과정에서의 관련 CT 요소

일반적인 문제해결과정은 <그림 1>에서 볼 수 있는 바와 같이 ‘문제인식’, ‘문제분석’, ‘해결책고안’, ‘문제해결의 실천’, ‘문제해결의 평가’로 구성될 수 있다. 첫 번째 ‘문제인식’ 단계는 해결하려고 하는 문제가 무엇인지 파악하는 단계이다. 두 번째 ‘문제분석’ 단계는 문제에서 중요한 핵심요소들을 뽑아내고 커다란 문제를 해결 가능한 작은 단계로 구성하는 단계이다. 세 번째 ‘해결책고안’ 단계는 분석된 내용을 토대로 해결방법들을 고안하고 가장 적절한 방법을 선택하는 단계이다. 네 번째 ‘문제해결의 실천’ 단계는 고안된 해결책을 직접 실현해보는 단계이다. 실현단계에서 발생되는 문제들을 검토하고 해결책을 수정 보완하도록 한다. 마지막으로 ‘문제해결의 평가’ 단계는 문제해결 단계에서 어려웠던 점, 후에 이와 유사한 문제를 수행하게 될 때 적용할 수 있는 기법, 개선해야 될 점 등에 대해 살펴본다.

이와 같은 일반적인 문제해결단계에서 컴퓨팅을 이용한 문제해결을 하려할 때 각 단계에서 고려될 수 있는 CT 구성요소를 살펴보면 먼저 ‘문제 인식’단계에서는 해결해야 되는 문제와 관련된 자료를 수집하는 과정이며 수집된 자료를 이해하고 그 자료 안에서 패턴이나 특징을 찾아 문제를



<그림 1> 일반적인 문제해결과정의 관련 CT 요소

이해하는데 도움을 얻도록 하는 단계이다. 두 번째 '문제분석'단계에서는 찾은 자료를 이해하기 쉬운 형태로 표현하거나, 정의된 문제를 해결가능한 수준의 작은 단위로 문제를 분해하도록 한다. 또한 문제해결을 위해 필요한 핵심요소를 뽑아내고 복잡한 것을 단순화시키는 추상화를 하는 단계이다. 세 번째 '해결책고안' 단계는 문제를 해결하거나 어떤 목표를 달성하기 위해 수행되는 일련의 각 과정들을 단계별로 표현하는 것으로 알고리즘을 작성하는 단계이다. 또한 이 단계에서는 문제해결을 위해 기존의 개발된 알고리즘들을 적용하거나 컴퓨팅 분야의 개념들을 적용하는 단계이다. 네 번째 '문제해결의 실천' 단계는 컴퓨팅 시스템이 수행할 수 있는 형태로 프로그래밍 언어를 선택하여 코딩하는 단계이다. 이 과정에서는 작성된 프로그램이 잘 동작이 되는지 디버깅을 하거나 오류들을 수정하도록 한다. 마지막 '문제해결의 평가' 단계는 문제해결과정에 대한 전반적인 평가단계로 작성된 프로그램의 효율성 및 개선방향에 대해 논의하도록 한다. 문제해결과정에서 사용된 기법이나 알고리즘을 다른 문제 상황에서 적용할 수 있도록 하는 일반화 과정도 포함한다.

3.2 CT의 인지적 측면

본 절에서는 문제해결과정에서의 CT 요소들을 Bloom의 교육목표 분류중 인지적 영역의 분류에 따라 구분한다. Bloom의 분류는 교수과정에서 도달해야 되는 학습목표와 함께 요구되는 인지기술의 체계를 보여주기 때문에 이를 통해 각 CT의 요소를 교수학습 측면에서 고려할 수 있다. Bloom은 인지적 학습영역을 지식, 이해, 적용, 분석, 종합, 평가로 분류하였다[17]. Anderson &Krathwohl(2001)은 Bloom의 기존의 분류 방식을 수정하여 기억하기, 이해하기, 적용하기, 분석하기, 평가하기, 생성하기로 수정하여 발표하였다[18]. 그런데 컴퓨터를 기반으로 하는 문제해결에서는 알고리즘 설계와 프로그램 작성을 한 후 그 작성된 프로그램에 대한 평가가 이루어지는 것이 일반적이다. 이와 같은 관점에서 볼 때 기존의 Bloom의 분류체계가 문제해결과정의 CT 요소들을 관련시키는데 보다 적절한 것으로 고려되어 본 연구에서는 이를 사용하였다. <표 2>는 문제해결과정에서 CT 요소들의 특징에 따라 Bloom의 인지적 영역의 분류 순서와 관련된 학습활동의 행동동사를 제시한 것을 보여준다. 이를 통해 각 단계에 따른 교수학습 과정에서 이루어질 수 있는 학습 활동들을 고려할 수 있다. 또한 각 과정에서 공통적으로 요구되는 학습활동으로 협동학습을 기술하고 있다.

첫 번째 '문제인식' 단계에서 '자료 수집', '자료 분석' 에 해당되는 인지적 분류는 '지식'과 '이해'이다. 이 단계는 문제를 인식하기 위해 기존에 알고 있던 지식을 기억하거나 검색된 자료를 통해 문제를 이해하는 과정에 해당된다. 두 번째 '문제

<표 2 > 문제 해결과정에서의 CT 요소 및 관련 학습목표의 인지적 분류

문제해결과정	CT 요소	Bloom의 분류	학습활동 행동 동사	공통 학습활동
문제 인식	• 자료수집 • 자료분석	지식	정의한다, 확인한다, 목록을 만들다, 기술한다	협동학습 탐구학습
		이해	식별하다, 토의한다, 설명한다, 예를 들다	
문제 분석	• 자료표현 • 문제분해 • 추상화	이해	그림이나 도표로 그린다. 바꾸어 말한다.	
		분석	분류한다, 관련시키다, 분해한다, 발견한다	
해결점 고안	• 알고리즘 및 절차	적용	사용하다, 작성하다, 시범을 보이다,	
		종합	통합하다, 구성하다, 결합하다, 조정하다	
문제해결의 실천	• 자동화 • 디버깅 • 시뮬레이션	종합	코딩한다. 개발한다. 일반화한다, 실행한다	
		평가	시험해본다, 결과를 확인한다, 수정한다.	
문제해결의 평가	• 평가 • 일반화	평가	평가한다, 비교한다, 개선점을 모색한다. 적용 분야를 논의한다.	

분석' 단계에 해당되는 문제분해, 추상화, 자료표현에 대한 인지적 분류는 '이해'와 '분석'으로 분류하였다. 이 단계에서는 커다란 문제를 나누고 그 모듈간의 관계를 분석하며 문제에서 중요한 부분을 뽑아내거나 문제의 특징에 해당되는 부분들을 자료로 표현하는 등의 일을 한다. 세 번째 '해결점 고안' 단계의 알고리즘 및 절차에 관한 인지적 분류는 '적용'과 '종합'이 해당된다. 이 단계는 두 번째 단계에서 분석된 내용을 토대로 실제 알고리즘을 설계하는 단계이기 때문에 문제를 해결하는데 관련 알고리즘을 적용하거나, 모듈들을 문제를 해결하기 위해 종합적으로 모으고 처리를 위해 절차적으로 구성한다. 네 번째 '문제해결의 실천' 단계의 인지적 분류는 '종합'과 '평가'가 해당된다. 직접 프로그램을 작성하여 실행하는 단계로 특정 프로그래밍 언어를 사용하여 코딩하며, 작성된 프로그램이 잘 수행이 되는지 디버깅하거나 테스트를 수행하는 단계가 된다. 마지막 '문제해결의 평가' 단계의 인지적 분류는 '평가'이다. 이 단계는 프로그램의 평가를 하는 것으로 작성된 프로그램의 효율성을 평가하거나 개선할 부분들에 대해 논의하는 단계이다. 사용된 알고리즘이나 개발된 모듈들을 적용할 수 있는 응용분야에 대해 논의하는 단계이다.

4. CT 역량 증진을 위한 수업 프레임워크

4.1 CT 요소를 위한 학습 활동

본 절에서는 각 CT 요소 및 정의, CT 요소에 따른 학습 활동 및 학습평가 요소를 <표 3>과 같이 기술한다. 문제해결 단계에 따른 각 CT 요소와 학습활동을 기술함으로써 CT 역량을 증진시키기 위한 수업설계를 용이하도록 한다.

4.2 예제를 이용한 CT 수업 프레임워크

<표 4>는 학급도서를 관리하기 위한 문제해결의 과정을 CT 요소들과 관련하여 설명한 것이다. 이 예제를 통해 문제해결 과정에서 CT 요소들이 어떻게 관련되는지를 살펴봄으로써 CT에 대한 이해를 돕고자 한다. 이 프레임워크를 기초로 실제 수업을 위한 설계를 할 때 2~3차시 기준으로 수업계획서를 작성하여 수업을 진행할 수 있다. 자동화 단계에서 프로그램을 작성하기 위해서는 적절한 프로그래밍 언어를 선택하여 그 언어의 문법에 맞게 작성해야 되므로 학습자는 해당 언어의 문법과 프로그래밍 환경에 숙달되어 있어야 한다. 본 프레임워크는 학습자가 해당 프로그래밍 언어와 프로그래밍 환경에 익숙하다고 가정하고작성된 것이다.

< 표 3 > 문제해결과정에서 CT 정의 및 학습활동

CT 요소	CT 정의	학습활동	학습평가요소
• 자료수집	해결해야 할 문제와 관련된 자료 수집	• 해결하려는 문제가 어떤 자료를 필요로 하는지 논의한다. • 문제에서 제시하고 있는 정보는 무엇인지 확인한다. • 자료 검색 방법을 결정하고 관련된 자료를 검색한다.	• 문제와 관련된 적절한 자료를 검색하는가? • 적절한 자료 검색방법을 사용하는가?
• 자료분석	자료를 이해하고 그 특징을 찾음	• 검색된 자료가 유용한지 확인한다. • 검색된 자료에서 특징을 찾아 기술한다. • 특징에 따라 자료를 구분한다.	• 검색된 자료가 적절한가? • 검색된 자료에 발견되는 특징을 잘 분류하였는가?
• 자료표현	자료를 다양한 형태로 정리	• 문제에 관련된 자료를 이해하기 쉽게 그림, 표, 식 등으로 표현한다. • 문제를 자신이 이해할 수 있는 방법으로 표현한다.	• 자료를 이해하기 쉽게 표현하고 있는가? • 문제를 정확히 이해하여 표현하고 있는가?
• 문제분해	문제를 해결가능한 수준의 작은 단위로 나눔	• 문제를 해결하는데 필요한 조건을 찾는다. • 문제를 해결가능한 작은 단위로 분할한다. • 문제에서 반복되는 부분이 있는지 찾는다.	• 문제에 있는 조건을 정확하게 기술하였는가? • 문제의 분할 단위는 논리적으로 적절하게 분할되었는가?
• 추상화	문제해결을 위해 필요한 핵심요소를 파악하고 복잡함을 단순화	• 문제에서 불필요한 부분을 제거하고 핵심부분을 표현한다. • 자료구조를 이용하여 자료를 표현한다. • 추상화 단위들의 관련성을 찾아 그 관계를 표현한다.	• 문제해결을 위한 핵심요소를 나열하였는가? • 자료구조는 적절하게 표현되었는가?
• 알고리즘 및 절차 (알고리즘 선택)	문제를 해결하거나 어떤 목표를 달성하기 위해 수행되는 순서를 기술	• 원하는 결과를 논리적인 순서로 작성한다. • 산술, 관계, 논리 연산자를 포함하도록 작성한다. • 결과를 얻기 위해 자료를 저장하고 움직이고, 작동하는 명령어를 포함하도록 작성한다. • 어떤 부분을 반복하는 것을 포함하도록 작성한다. • 모듈화를 포함하도록 작성한다. • 이미 존재하는 알고리즘을 적절하게 선택한다.	• 알고리즘의 기술이 논리적으로 올바른가? • 문제를 해결하는데 적절한 알고리즘을 선택하였는가? • 알고리즘의 기술시 적절한 제어구조를 사용하고 있는가? • 알고리즘의 기술시 적절한 연산자를 사용하고 있는가?
• 자동화	컴퓨팅 시스템이 수행할 수 있는 형태로 코딩	• 프로그래밍 언어를 선택한다. • 프로그래밍 언어의 문법에 맞게 프로그램을 작성한다. • 작성된 프로그램을 실행한다.	• 적절한 프로그래밍 언어를 선택하였는가? • 프로그래밍 구문에 맞게 프로그램을 작성하였는가?
• 디버깅 • 시뮬레이션	프로그램의 오류 검사와 실행결과 확인 및 수정	• 프로그램의 오류를 체크 및 수정한다. • 프로그램을 수행시켜 원하는 결과가 나오는지 확인한다.	• 프로그램의 오류를 잘 수정하였는가? • 프로그램이 논리적인 오류 없이 잘 수행되는가?
• 평가	개발된 프로그램의 성능 평가 및 개선 부분에 대한 논의	• 프로그램의 효율성을 체크한다. • 프로그램을 향후 개선해야 될 부분에 대해 논의한다. • 프로그램의 개발과정에 대해 평가한다.	• 프로그램의 효율성에 대한 체크가 올바른가? • 향후 프로그램의 개선될 부분에 대한 논의가 적절한가?
• 일반화	개발된 알고리즘의 적용 분야에 대해 논의	• 사용된 알고리즘이나 개발된 알고리즘의 응용분야에 대해 논의한다.	• 개발된 프로그램의 응용분야를 적절하게 논의하고 있는가?

< 표 4> 예제를 이용한 CT 수업 프레임워크

학습주제		학급도서를 관리하는 프로그램 개발	
학습목표		• 도서를 관리하기 위해 필요한 기능이 무엇인지 이해한다. • 도서 관리에 필요한 기능을 이해하고 이를 모듈화하여 알고리즘으로 작성한다. • 작성된 알고리즘을 프로그램으로 코딩하여 실행한다.	
학습의도		도서관리 문제를 해결하기 위해 문제해결단계의 CT 요소와 그에 따른 학습활동을 알아보고 이에 따라 수행함으로써 CT를 이용한 문제 해결과정을 이해할 수 있도록 한다.	
CT 요소	자료수집	도서를 관리하기 위해 요구되는 기능들을 알아보기 위해 자료 조사를 한다.	탐구 학습
	자료분석	조사된 내용을 토대로 도서관리에 필요한 기능을 분석한다. 도서정보에 필요한 요소들이 무엇인지 논의한다.	
	자료표현	논의된 도서정보의 요소들을 알아보기 쉽게 표를 이용하여 기술한다. 도서관리에 필요한 기능들을 그림이나 표를 이용하여 기술한다.	
	문제분해	도서관리에 필요한 기능들을 구분하여 모듈로 분해한다. (도서입력, 도서검색, 도서 삭제, 도서출력 등)	
	추상화	각 모듈을 정의하기 위해 필요한 요소들을 정리하여 기술한다. 도서정보를 저장할 자료구조를 결정한다. 일의 처리 순서에 따라 플로차트를 작성한다.	
	알고리즘 및 절차	추상화단계에서 정의된 플로차트의 수행 순서에 따라 알고리즘을 작성한다. 각 모듈에 대한 구체적인 수행 단계를 알고리즘으로 작성한다.	
	자동화	코딩할 프로그래밍 언어를 선택한다. 작성된 알고리즘을 프로그래밍 언어의 문법에 맞게 프로그램을 작성한다.	협동 학습
	디버깅	작성된 프로그램을 실행하여 오류를 체크하고 수정한다. 문법오류와 논리적 오류를 체크한다.	
	평가	개발된 프로그램이 효율적인지 체크하고 앞으로 개선해야 할 사항들에 대해 논의한다. (책의 양이 많아질 때 프로그램의 효율을 높이기 위해 프로그램을 어떻게 수 정할지 논의)	
	일반화	개발된 프로그램의 모듈들을 사용할 수 있는 응용 분야에 대해 논의한다. (주소록 관리, 자재 관리 등)	

5. CT 역량 교육을 위한 고려사항

CT 역량을 증진시키기 위한 교육이 효과적으로 이루어지기 위해서는 다양한 측면에서의 지원이 요구된다. 그중 CT 역량을 평가할 수 있는 평가 척도와 교수법에 대한 연구는 매우 중요한 부분이라 볼 수 있다. 2.3절에서 살펴본 CT 평가 척도에 관한 연구들은 대부분이 스크래치 프로그래밍에 기반을 두고 있으며 [17]의 연구만 Alice 기반의 프로그래밍 환경에 기반하고 있다. 즉, 특정 프로그래밍 환경에서 학생들의 코딩 산출물을 근거로 하여 평가를 수행하고 있다. 특정 프로그래밍 언어와는 독립적으로 CT 역량을 측정할 수 있는 평가 척도가 포함되는 것이 필요하다. 즉, CT 역량을 프로그래밍 스킬로 한정하기 보다는 하나의 사고과정으로 보고 인지적인 측면에서의 보다 체계적인 평가 척도에 대한 연구가 필요하다고 볼 수 있다. 한편, Brennan과 Resnick의 연구[16]의 경우 Computational Perspective는 학습자들이 스크래치를 이용한 설계기반의 학습을 통해 발전된 관점의 변화를 평가하기 위한 것이다. 이러한 관점은 스크래치를 이용한 프로그래밍을 통해 정보의 소비자에서 자신의 아이디어를 프로그래밍을 통해 산출물로 생성할 수 있고 다른 사람들과의 소통과 협동이 더 나은 아이디어와 풍성한 결과물들을 만들어낼 수 있다는 점이다. 또

한 더 나은 세상을 위해 우리가 당연하게 생각했던 것들에 대해 의문을 갖고 발전 방향을 찾고자 하는 관점이다. 이러한 Computational Perspective는 보다 폭넓은 관점에서의 CT 역량에 대한 평가로 21세기를 살아가는 학습자들이 갖추어야 할 사고의 관점이라 볼 수 있다. 따라서, 이러한 관점들을 고려한 교육과 이에 대한 평가척도를 개발하는 것도 의미있는 작업이라 생각된다.

위와 같은 평가척도에 대한 연구뿐만 아니라 CT 역량 교육을 위한 다양한 교수법 및 교수전략에 대한 연구가 요구된다. CT는 컴퓨팅을 이용한 문제해결과 관련이 있기 때문에 학습자들의 창의적인 문제해결력을 높이기 위한 교수법이 필요하다. 이를 위해서는 학습자 중심의 탐구학습과 협동학습이 이루어지도록 하는 것이 바람직하다. 미국의 Collage Board와 National Science Foundation에서 제안한 CT를 위한 교육과정 프레임워크로서 CT Practice(실제)에는 '의사소통'과 '협동'을 포함하고 있다. '의사소통'과 '협동' 능력은 문제해결과정에서 중요한 사회적 요소이기 때문에 이러한 능력들을 신장시키기 위한 교수 설계가 필요하다. 최근에 학습자들의 문제해결력과 학습자들의 능동적 학습참여를 위해 플립드 러닝과 같은 교수법이 여러 교과에서 적용되고 있다. 정보교과에서도 플립드 러닝을 통해 이미 준비된 학습자료나 동영상 자료를 통해 학습자들이 기본 학습개념이나 프로그래밍 언어의 문법을 학습한 후 실제 수업시간에는 응용문제를 코딩하도록 하는 것도 학습자의 문제해결력을 높일 수 있다. 또한 교실 수업에서 학생들이 팀을 이루어 서로 토론하며 과제를 해결하도록 함으로써 학생들의 의사소통과 협동 능력을 키울 수 있을 것으로 기대된다.

6. 결 론

본 연구에서는 문제해결력의 관점에서 CT 요소들이 어떻게 관련되는지를 살펴보고 CT 역량을 증진시키기 위한 수업 프레임워크를 제안하였다. 이를 위해 CT의 핵심요소를 뽑아내고 이 요소들의 인지적 측면을 살펴보았다. 이 연구결과를 통하여 정보교과를 가르치는 교사들은 문제해결의 관점에서 CT 요소들을 이해하고 어떻게 수업을 설계할 것인지에 대한 정보를 얻을 수 있을 것으로 기대한다.

CT 역량 교육이 효과적으로 이루어지기 위해서는 여러 가지 측면에서의 지원이 이루어져야 할 것이다. 먼저, CT 역량을 증진시키기 위한 교육이 코딩교육에 치우기보다 학습자들의 창의력, 문제해결력, 협동능력을 높이기 방향으로 접근하는 것이 필요하며 이를 위해 적절한 교육과정의 마련과 운영 방안이 요구된다. 또한 CT의 중요성을 강조하며 CT 요소들이 문제해결과정에 어떻게 사용되는지를 살펴보는 것도 중요하지만 이와 함께 컴퓨팅이 우리 삶에 미치는 영향들에 대해서 깊이 있는 이해가 이루어질 수 있도록 해야 할 것이다. 마지막으로 CT 역량 교육을 위한 다양한 수업 모델 및 평가 척도 등이 개발되어 교육현장에서 쉽게 이용할 수 있도록 하고 경험있는 선도 교사들의 수업 성공 사례들에 대한 정보 공유가 활발하게 이루어질 수 있도록 지원과 격려가 필요하다.

참 고 문 헌

[1] Wing, J. (2006). Computational Thinking. Commun. ACM, 49, 3, 33-35.

[2] Grover, S. & Pea, R.D. (2013). Computational Thinking in K-12: A Review of the State of the Field. Educational Researcher. 42(1), 38-43.

[3] Brennan, K., & Resnick, M. 2012. New frameworks for studying and assessing the development of computational thinking. Paper presented at AERA 2012, Vancouver, Canada.

[4] Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012, February). The Fairy Performance Assessment: Measuring computational thinking in middle school. In Proceedings of the 43rd ACM technical symposium on Computer Science Education (pp. 215-220). ACM.

[5] Selby, Cynthia, Dorling, Mark and Woollard, John (2014) Evidence of

assessing computational thinking. Brookes eJournal of Learning and Teaching, 1-12.
[6] 이영준 외 7인 (2014). 초중등 단계 Computational Thinking 도입을 위한 기초연구, 한국과학창의재단 연구 보고서
[7] Wing, J. (2011). Research Notebook: Computational Thinking-What and Why? The Link. Pittsburgh, PA: Carneige Mellon.
[8] International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA), 2011 Operational Definition of Computational Thinking for K-12 Education, http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf
[9] Aho, A. A. (2012). Computation and computational thinking. The Computer Journal, 55(7), 832-835.
[10] Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? ACM Inroads, 2(1), 48- 54.
[11] The College Board (2012). Computational thinking practices and big ideas, key concepts, and supporting concepts. Retrieved from http://www.csprinciples.org /home/ about-the-project.
[12] Seiter, L., & Foreman, B. (2013, August). Modeling the learning progressions of computational thinking of primary grade students. In Proceedings of the ninth annual international ACM conference on International computing education research (pp. 59-66). ACM.
[13] Brennan, K., & Resnick, M. (2012). Using artifact-based interviews to study development of computational thinking in interactive media design. Paper presented at annual American Educational Research Association meeting, Vancouver, BC, Canada.
[14] Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012, February). The Fairy Performance Assessment: Measuring computational thinking in middle school. In Proceedings of the 43rd ACM technical symposium on Computer Science Education (pp. 215-220). ACM.
[15] 김수환, 한선관(2012). Computational Thinking 향상을 위한 디자인 기반 학습, 한국 정보교육학회 논문지, 16, 3, 319-326
[16] 최형신(2014). Computational thinking 역량 계발을 위한 수업 설계 및 평가 루브릭 개발, 한국 정보교육논문지, 18,1, pp.57-64
[17] Bloom, B. S.; Engelhart, M. D.; Furst, E. J.; Hill, W. H.; Krathwohl, D. R. (1956). Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain. New York: David McKay Company.
[18] Anderson, Lorin W.; Krathwohl, David R., eds. (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Allyn and Bacon. ISBN 978-0-8013-1903-7.
[19] Computing At School (2015), Computational Thinking : A Guide for teachers, Retrieved from http://community.computingatschool.org.uk/resources/2324

최 숙 영

1998 전북대학교 전산학과 (이학사)
1991 전북대학교 전산학과 (이학석사)
1996 충남대학교 전산학과(이학박사)
2008 Nova Southeastern University 교육공학 및 원격교육(교육학박사)
1996~현재 우석대학교 컴퓨터교육과 교수
2012.3~ 2013.2 North Carolina State University 연구교수
관심분야: 이러닝시스템, 컴퓨터과학교육, 사이버불링
E-mail : sychoi@ws.ac.kr