

IT 융합교육 활성화를 위한 Computational Thinking 수업 모형 설계

손영수* · 이광재**

Computational Thinking Teaching Model Design for Activating IT Convergence Education

Young-Su Son* · Kwang-Jae Lee**

요 약

최근 학문 간의 경계를 허물고 새로운 지식을 창출하는 융합 교육이 대학 교육의 핵심 이슈로 부각되고 있다. IT융합 교육의 핵심은 SW 교육을 통해서 이루어지고 SW 교육의 목적은 CT(: Computational Thinking)을 향상하는데 초점이 맞추어 지고 있다. 본 논문의 목적은 IT 융합 교육을 활성화하기 위한 CT 수업모형을 설계하는 것이다. 본 연구에서는 IT 비전공 학습자의 CT 능력과 SW 적응력을 높이고 전공분야에서 활용될 수 있도록 알고리즘과 프로그래밍을 중심으로 CT 교육과정을 제시하고, 교수자와 학습자간의 CT 수업 효과와 성취도를 향상하기 위하여 CT 수업 제반요소와 수업에 대한 성취기준, 교수 학습방법을 통해 IT 비전공 학습자에 적합한 수업 모형을 설계 하였다. 이는 대학에서 IT 융합 교육을 위한 CT 수업 방법과 교육 과정을 설계하는데 활용할 수 있을 것으로 기대된다.

ABSTRACT

Breaking down the boundaries between recent study has emerged as a key issue of convergence education to create new knowledge. The core of IT Convergence Education is being made through educational SW, SW purpose of education has been focused on improving the CT(: Computational Thinking). The purpose of this paper is to design a CT teaching model for activating IT Convergence Education. In this study, we designed a CT Curriculum focusing on algorithm and program for the purpose of enhancing non-majors learner's CT ability and software adaptability and being utilized in the majors. It is expected to be utilized to design a CT teaching methods and curriculum in University.

키워드

Convergence Education, Computational Thinking, CT Teaching Model, Non-Majors Learner, Curriculum
융합 교육, 컴퓨팅 사고력, CT 수업 모형, 비전공 학습자, 교육 과정

1. 서 론

융합 교육은 기존의 단편적인 지식전달 위주의 교육을 통해서 는 실제 생활 속에서 발생하는 복잡하고

* 한려대학교 전기전자공학부(sys5733@hanmail.net)

** 교신저자 : 한려대학교 전기전자공학부

• 접수 일 : 2016. 04. 25

• 수정완료일 : 2016. 05. 13

• 게재확정일 : 2016. 05. 24

• Received : Apr. 25, 2016, Revised : May. 13, 2016, Accepted : May. 24, 2016

• Corresponding Author : Kwang-Jae Lee

Dept. of Electrical and Electronic Engineering, Hanlyo University

Email : kjlee3@naver.com

다양한 문제를 해결하기에 어려움이 따른다는 점을 지적하며 이에 대한 대안으로 떠오르고 있는 교육이다[1]. 융합적 접근을 통한 교육 방식은 학문적 탐구 뿐 만 아니라 실생활의 문제를 다차원적으로 그리고 심층적으로 다룰 수 있기에 보다 실제적이고 실용적인 교육을 도모할 수 있다[2]. 컴퓨터과학적인 측면에서 융합 교육은 학문분야에 관계없이 실생활에서 발생하는 복잡한 문제를 컴퓨터 과학의 기본기술과 컴퓨팅 원리를 활용해 문제를 해결하는 과정이다. 융합 교육에서 가장 필요로 하는 것은 융합적인 사고를 학습하는 것인데, CT는 융합적 사고력이 요구되는 기초 능력이라 할 수 있으며 컴퓨터 과학 교육의 핵심 개념이라고 할 수 있다[3]. IT분야에서 융합은 다양한 산업분야에서 활발히 진행되어 게임, 앱, 클라우드 등 다양한 어플리케이션이 이미 우리의 생활 전 영역에 적용되고, 더 나아가 가전제품, 자동차 등 다양한 사물에 SW가 내장되면서 모든 사물이 네트워크를 통해 상호작용하는 것을 의미하는 '사물인터넷(IoT)'이라는 새로운 디지털 용어가 탄생하였다. 조선, 건설, 전력, 섬유 등 대부분의 전통산업에서도 IT 융합이 확대되면서 내장형 소프트웨어의 채택이 확산되고 소프트웨어 역량이 중요한 경쟁우위로 부상하고 있다.

이러한 변화를 인지한 정부에서는 2015년 7월에 "소프트웨어 중심사회를 위한 인재양성 추진계획"을 발표하면서 타 전공지식과 SW 소양을 겸비한 인재양성을 위해, 모든 비전공 학생 등을 대상으로 대학 내 SW 기초교육을 확대하고 컴퓨터 과학의 원리를 활용하여 문제를 분석 해결하는 CT 역량을 강화할 수 있는 대학별 특화된 SW 융합 교육과정운영을 의무화 하도록 하였고 이와 관련된 CT 교육에 대한 대학의 교과과정은 대학실정에 따라 차이가 있지만 IT 비전공자를 대상으로 교양 필수로 지정 하여 수도권 대학 중심으로 운영되어 지고 있고 점차 확대되어 가고 있는 추세이다.¹⁾ 하지만 SW 융합교육을 위한 CT 교육 방법이나 수업에 관련된 콘텐츠는 미진한 실정이다. 지금까지 대학의 교양 IT 교육과정은 정보소양 중심의 이미 만들어진 소프트웨어를 활용하는 방법을 강

의하고 실습하도록 구성 되어있다[4].

CT에 대한 교육과정 개발은 IT와 관련 없는 비전공자들에게 자신의 전공분야에서 IT를 통한 혁신 아이템을 찾을 수 있는 능력, 기술에 대한 두려움을 극복하고 활용 대상으로 인식할 수 있는 능력, 혁신적 아이디어를 구현할 수 있는 능력 등을 갖추도록 도울 수 있도록 구성되어야 한다. 이러한 CT 능력은 컴퓨터 프로그래밍 교육을 통해서 개발하고 훈련할 수 있다[5-6].

본 연구에서는 CT 교육을 통해서 IT 융합교육을 활성화하고 비전공학습자의 IT 융합능력과 CT 능력을 배양하여 학습자의 전공분야에서도 학습한 내용이 응용될 수 있도록 CT 수업모형을 설계 하고자 한다. 본 논문의 구성은 2장에서는 CT의 이론적 배경과 선행연구를 기술하고 3장에서는 CT 교육과정을 제시하고 CT 수업에 필요한 제반 요소를 통하여 CT 수업모형을 설계한다. 4장과 5장에서는 CT 수업모형에 대한 성취기준과 교수 학습방법을 기술한다. 6장에서는 결론을 논하였다.

II. 이론적 배경

2.1 CT 정의

CT란 용어는 2006년 미국 카네기 멜론 대학교의 컴퓨터과학과 교수 Wing에 의해 컴퓨터학계에 처음으로 소개되었는데, Wing은 이것이 21세기를 살아가는 모든 사람에게 기본적으로 필요한 기술로서 읽기, 쓰기, 셈하기와 같은 기본소양에 CT를 추가해야한다고 주장하고 있다. 또한 CT는 복잡하고 어려운 문제를 감소시키고, 시뮬레이션 방법을 통해 우리가 해결하고자 하는 문제로 재구조화하는 것 이라고 정의 하였다. 그리고 CT의 예로 재귀적 사고, 방대하고 복잡한 문제를 해결하기위한 추상화와 분해, 프리패칭과 캐싱, 단순명료한 시스템 설계, 중복·손상 억제 및 오류정정을 통한 최악의 시나리오에서 예방하고 보호 및 복구의 관점에서 생각, 발견적 추론 등을 들었다[7-8].

2.2 CT 구성요소

Wing(2008)은 CT를 크게 추상화(Abstraction)와 자동화(Automation)로 설명하였다. 추상화는 실제세

1) <http://www.moe.go.kr/web/100026/ko/board/view.do?bbsId=294&pageSize=10¤tPage=45&enodeYn=N&boardSeq=60077&mode=view>

계의 문제를 해결 가능한 형태로 표현하기 위한 사고 과정이라고 할 수 있으며, 추상화 과정을 통해 만들어진 해결 모델을 컴퓨터가 이해할 수 있는 프로그래밍 언어로 표현하여 인간이 처리하기 어려운 많은 양의 반복된 작업이나 시뮬레이션을 실시하는 것을 자동화라고 하였다[8-9].

ISTE(International Society for Technology in Education)와 CSTA(Computer Science Teachers Association)에서는 CT의 조작적 정의를 내리면서 세부요소를 Data Collection, Data Analysis, Data Representation, Problem Decomposition, Abstraction, Algorithms & Procedures, Automation, Simulation, Parallelization 9 가지 요소를 표 1과 같이 제시하였다[10].²⁾³⁾

표 1. CT 어휘 및 정의
Table 1. CT vocabulary and definition

CT Vocabulary	Definition
Data Collection	The process of gathering appropriate information
Data Analysis	Making sense of data, finding patterns, and drawing conclusions
Data Representation	Depicting and organizing data in appropriate graphs, chart, words, or images
Problem Decomposition	Breaking down tasks into smaller, manageable parts
Abstraction	Reducing complexity to define main idea
Algorithms & Procedures	Series of ordered steps taken to solve a problem or achieve an objective
Automation	Having computers or machines do repetitive or tedious tasks
Simulation	Representation or model of a process. Simulation also involves running experiment using models
Parallelization	Organize resources to simultaneously carry out tasks to reach a common goal

또한 CT는 다음 특성들을 포함하는 문제 해결 과정이라고 하였다[11].

첫째, 문제 해결을 돕기 위해 컴퓨터 또는 다른 도구를 사용할 수 있도록 문제를 정형화하기

둘째, 자료를 논리적으로 분석하고 모델링, 시뮬레이션과 같은 추상화를 통해 자료 표현하기

셋째, 알고리즘적 사고를 통해 해법들을 자동화하고 가능한 최적 해법들의 확인, 분석, 구현하기

넷째, 이러한 문제 해결과정을 광범위한 다양한 문제들로 일반화하고 전이하는 것이라 하였다. 이러한 특성들은 IT 융합교육을 통하여 학습해야 하는 필수 요소이다.

2.3 CT 교육과 수업모형에 대한 선행연구

수업모형은 교수·학습 절차 또는 단계를 지칭하는 용어로, 교수·학습 모형, 교수모형, 교수·학습 유형 등과 같은 의미로 사용된다. 계획, 진단, 지도, 평가 등과 같이 학습 단계를 하나의 모형으로 보기도 한다. 조이스와 웨일(Joyce & Weil)은 수업 모형을 교육과정이나 교과과정을 구성하거나, 수업자료를 선정하고, 교수자의 행위를 안내하는데 이용될 수 있는 형태나 계획이라고 정의하였다[12-13]. CT 교육의 효과와 CT 교육의 수업모형에 관한 선행 연구의 내용은 첫째, CT 능력의 세부요소로 보이는 증거들을 기초, 발달, 능숙의 3단계 수준으로 평가 할 수 있는 루브릭을 고안하고 CT 세부 역량을 체계적으로 연결 시켜서 CT 역량강화를 위한 방안을 제시하였다[14]. 둘째, CT를 길러주기 위하여 디자인 기반학습을 적용하고 그 결과를 분석하여 CT 능력의 향상 등을 제시하였다[15]. 셋째, 수학교과에서 CT를 적용한 교육 프로그램을 개발하여 학생의 CT 능력 신장에 도움이 된다는 결과를 도출하였다[16]. 넷째, SW 교육의 목표인 CT의 증진을 위해 컴퓨터 비전공자를 위한 CT 교육으로서 학습자 중심의 교수 학습전략을 적용하여 그 효과성을 검증하였다[17]. 위의 선행연구의 공통점은 여러 방법의 CT 교육방법과 다양한 교육 내용을 통하여 CT 역량을 강화 시키는 것이다.

III. CT 수업모형 설계

3.1 CT 수업모형의 학습 가이드라인

본 연구에서는 CT 수업모형을 설계하기 위하여 CSTA Computer Science Standards(2011) Level 1,2,3의 교육영역(Computational Thinking, Collaboration,

2) <http://csta.acm.org/Curriculum/sub/CurrFiles/471.11CTLeadershipToolkit-SP-vF.pdf>

3) <http://csta.acm.org/Curriculum/sub/CompuThinking.html>

표 2. CSTA 레벨 2,3: CT 적용개념(CT,CP&P)

Table 2. CSTA Level 2,3: Applying Concepts(CT,CP&P)

Level	CSTA Applying Concepts(CT)	CSTA Applying Concepts(CP&P)
2	<ol style="list-style-type: none"> 1. Use the basic steps in algorithmic problemsolving to design solutions. 2. Describe the process of parallelization as it relates to problem solving. 3. Define an algorithm as a sequence of instructions that can be processed by acomputer. 4. Evaluate ways that different algorithms may be used to solve the same problem. 5. Act out searching and sorting algorithms. 6. Describe and analyze a sequence of instructions being followed (e.g., describe a character's behavior in a video game as driven by rules and algorithms). 7. Represent data in a variety of ways including text, sounds, pictures, and numbers. 8. Use visual representations of problem states, structures, and data (e.g., graphs, charts, network diagrams, flowcharts). 9. Interact with content-specific models and simulations (e.g., ecosystems, epidemics, molecular dynamics) to support learning and research. 10. Evaluate what kinds of problems can be solved using modeling and simulation. 11. Analyze the degree to which a computer model accurately represents the real world. 12. Use abstraction to decompose a problem into sub problems. 	<ol style="list-style-type: none"> 1. Select appropriate tools and technology resources to accomplish a variety of tasks and solve problems. 2. Use a variety of multimedia tools and peripherals to support personal productivity and learning throughout the curriculum. 3. Design, develop, publish, and present products (e.g., webpages, mobile applications, animations) using technology resources that demonstrate and communicate curriculum concepts. 4. Demonstrate an understanding of algorithms and their practical application. 5. Implement problem solutions using a programming language, including: looping behavior, conditional tatements, logic, expressions, variables, and functions. 6. Demonstrate good practices in personal information security, using passwords, encryption, and secure transactions. 7. Identify interdisciplinary careers that are enhanced by computer science. 8. Demonstrate dispositions amenable to openended problem solving and programming (e.g., comfort with complexity, persistence,brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge). 9. Collect and analyze data that is output from multiple runs of a computer program.
3	<ol style="list-style-type: none"> 1. Use predefined functions and parameters, classes and methods to divide a complex problem into simpler parts. 2. Describe a software development process used to solve software problems (e.g., design, coding, testing, verification). 3. Explain how sequence, selection, iteration,and recursion are building blocks of algorithms. 4. Compare techniques for analyzing massive data collections. 5. Describe the relationship between binary and hexadecimal representations. 6. Analyze the representation and trade-offs among various forms of digital information. 7. Describe how various types of data are stored in a computer system. 8. Use modeling and simulation to represent and understand natural phenomena. 9. Discuss the value of abstraction to manage problem complexity. 10. Describe the concept of parallel processing as a strategy to solve large problems. 11. Describe how computation shares features with art and music by translating human intention into an artifact. 	<ol style="list-style-type: none"> 1. Create and organize Web pages through the use of a variety of web programming design tools. 2. Use mobile devices/emulators to design, develop, and implement mobile computing applications. 3. Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing) 4. Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models). 5. Use Application Program Interfaces (APIs) and libraries to facilitate programming solutions. 6. Select appropriate file formats for various types and uses of data. 7. Describe a variety of programming languages available to solve problems and develop systems. 8. Explain the program execution process. 9. Explain the principles of security by examining encryption, cryptography, and authentication techniques. 10. Explore a variety of careers to which computing is central. 11. Describe techniques for locating and collecting small and large-scale data sets. 12. Describe how mathematical and statistical functions, sets, and logic are used in computation.

Computing Practice & Programming, Computer and Communication Device, Community Global and Ethical Impacts) 중에서 표 2와 같이 Level 2,3의 Computational Thinking과 Computing Practice & Programming을 기본으로 설정하고 CSTA와 ISTE에서 제시한 Data Collection, Data Representation, Data Analysis, Problem Decomposition, Abstraction, Algorithms & Procedures, Automation, Parallelization, Simulation의 9가지 조작적 정의를 CT학습 세부요소로 적용하였다.⁴⁾

3.2 CT 수업 제반요소

IT 비전공 학습자의 CT 수업의 효과를 높이고 교수자와 학습자간의 CT 수업의 성취도를 높일 수 있는 수업의 준비요소(Preparation), 운영요소(Operation), 전환요소(Transition)로 구분하여 CT 수업모형에 활용하기 위해 CT 수업의 제반요소를 제시하였다.

3.2.1 준비요소(Preparation)

CT의 세부적인 수업목표를 달성하기 위해서는 각 목표별로 어떻게 교육할 것인가에 대한 CT 수업 준비 사항들이 고려되어야 한다. 준비요소로서 첫 번째는 컨텐츠(Contents)이다. 컨텐츠는 CT 수업 주제를 무엇으로 할 것 인가에 대한 요소를 의미한다. 컨텐츠는 교육 시작 전에 강의계획서에 기술한다. 강의 계획서에 포함된 수업 컨텐츠는 학습자들의 수준과 수업 환경에 따라 정의 되어야 하며, 컨텐츠가 얼마나 효과적으로 구성되었는지에 따라서 CT 수업의 질을 결정하게 된다. 특히 주어진 일정에 따라 수업을 진행해야 하기 때문에 합리적인 컨텐츠의 범위 및 내용 결정이 중요하다.

두 번째는 도구(Tools)이다. CT에서 다루고자 하는 주제에 대하여 도구를 사용하여 구현이 가능하게 해야 한다. 따라서 CT 수업의 주제들을 보다 효과적이고 능률적으로 전달하기 위해서는 어떠한 도구를 어떤 주제로 사용할 것인지가 결정되어야 한다.

3.2.2 운영요소(Operation)

CT 교육의 수행과정에서 수업의 질을 향상시키기 위한 운영요소로서 첫 번째는 명확성(Clearity)이다. CT 수업에서는 다양한 용어 및 개념에 대한 명확한 이해가 선행되어야 한다. 예를 들면 융합과제를 수행하기 위한 “추상화란 무엇인가?” 라는 질문에 학습자의 명확한 대답이 가능해야한다. 기본개념에 대한 정확한 정의를 알고 있다는 것은 실제과악이 명확하게 이루어 졌다는 차원에서 매우 중요한 CT 수업향상의 결정요소이다.

두 번째는 관련성(Relativity)이다. 관련성은 개념에 대한 정확한 숙지를 기반으로 개념간의 상호 연관성을 명확히 이해해야한다. 예를 들면, 추상화와 분해의 관계가 어떤 관계인지에 대해서 숙지해야한다. 이러한 개념간의 관련성에 대한 이해가 없다면 CT 수업의 진행이 어렵고, 또한 학습자로부터 기대할 수 있는 교육성과도 달성할 수 없다.

세 번째는 점진성(Graduality)이다. CT 수업의 대상이 되는 모든 세부 주제들은 점진적이고 단계적으로 학습이 이루어 져야 하며, 이것이 왜 중요한지, 그리고 어떻게 도구를 이용하여 적용되는지를 이해하는 것이 중요하다. 점진적이고 단계적인 수업이 진행될 때 심화된 주제에 대하여 CT 학습이 가능하게 된다.

네 번째는 실증성(Reality)이다. CT 수업을 처음 접하는 학습자들은 CT 수업이 대체적으로 추상적인 개념을 다루고 있다는 점에서, CT와 관련된 지식들이 각 학문 분야의 실무적인 측면에서 어떻게 쓰이고 있는지 실증적인 예시가 필요하다. 이러한 예시를 통해서 설명하는 것은 CT 학습효과 및 이해도를 높이는 요소라 할 수 있다.

3.2.3 전환요소(Transition)

이 요소는 CT에 대한 기본개념 학습을 수행한 후 각각 다른 전공분야에서 실무에 적용하기위해 요구되는 능력으로서 첫 번째는 적용성(Applicability)이다. CT 교육은 학문의 융합 관점에서 적용할 수 있도록 수업이 이루어 져야 한다. 수업이 충실하게 이루어 졌다 할지라도 전공분야 업무에 적용할 수 없다면 CT 교육의 의미는 상실된다. 따라서 CT의 주제들이 각 학문의 실무 현장에서 잘 적용될 수 있도록 하는 것이 CT 교육의 성과와 직결된다.

4) <http://csta.acm.org/Curriculum/sub/CompuThinking.html>

표 3. CT 교육과정 설계
Table 3. CT Curriculum Design

Level	Week	Theme	Core Contents	Learning Content Themes
1	1	<ul style="list-style-type: none"> - What is Computational Thinking? - Computational thinking is, why is it important? - Examples of computational thinking 	<ul style="list-style-type: none"> - Computational thinking is required to understand why at this time. - Discover the differences between computer science and computational thinking in daily life. 	<ul style="list-style-type: none"> - Computer and Computational Thinking Why Computational Thinking? - Examples of computer science and daily life - Group Discussion Lesson Feedback & Movie Lesson - http://www.google.com/edu/resources/programs/exploring-computational-thinking/
	2	<ul style="list-style-type: none"> - The structure of computer hardware - Software influence on the human - Introduction scratch and Python 	<ul style="list-style-type: none"> - Understand the Evolution of computer architecture. - Discuss what the AI program AlphaGo Google. - The sample program runs through the successes gives the computational thinking motivation 	<ul style="list-style-type: none"> - Evolution of computer architecture - AlphaGo Program(Group Discussion Lesson Feedback) - https://studio.code.org/gallery - https://www.python.org/about/success/ - Movie Lesson
2	3	<ul style="list-style-type: none"> - Pseudo code - Flowchart - Algorithm(linear,non-linear) - Algorithm complexity 	<ul style="list-style-type: none"> - Pseudo code to learn how to use. - The learning algorithm the differences between the linear and non-linear algorithm and understanding. - It analyzes the complexity of the algorithm to understand the algorithm design process 	<ul style="list-style-type: none"> - Pseudo code, Flowchart - Stack, Queue, Deque, List - Graph(DFS/BFS) - Selection Sort, Quick Sort, Merge Sort - Sequential Search, Binary Search, Hashing
	4	<ul style="list-style-type: none"> - Algorithm Design Techniques 	<ul style="list-style-type: none"> - Learn algorithmic thinking through examples of daily life. - Dynamic programming compared to the divide-and-conquer algorithm 	<ul style="list-style-type: none"> - Divide and Conquer - Recursive Call - Dynamic Programming - Greedy technique
	5	<ul style="list-style-type: none"> - Solving Problems - Concurrent Activity 	<ul style="list-style-type: none"> - Find the core concepts of computational thinking in daily life. - To understand the difference between concurrency and parallelism 	<ul style="list-style-type: none"> - Problem Definition, Logical Reasoning - Decomposition, Abstraction - Concurrency, Parallelism, Pipeline, Deadlock - Group Discussion Lesson Feedback
	6	<ul style="list-style-type: none"> - Modeling Solution - Diagramming control flow 	<ul style="list-style-type: none"> - Learn control abstraction in activity diagram - Modeling the growth process of learners. 	<ul style="list-style-type: none"> - Symbol in activity diagram - Symbols in state diagram
	7	- Midterm Exam(Team Project)		
	8	<ul style="list-style-type: none"> - Install Python - Input, Output statement - Debugging - Variable, Operator - Draw using Turtle 	<ul style="list-style-type: none"> - Understand what the python program. - Use arithmetic operator - Use relational operator - Draw various polygons using Turtle 	<ul style="list-style-type: none"> - Introduction to Python - Programming error - Arithmetic operator - Relational operator - Turtle module
	9	<ul style="list-style-type: none"> - Conditional statement - Iterative statement - Function 	<ul style="list-style-type: none"> - The learning conditional statement iterative statements - Learn how to use the basic structure and function. 	<ul style="list-style-type: none"> - Use if statement to code several conditions - Use while statement & for statement - Define functions & Parameters to passing
3	10	<ul style="list-style-type: none"> - Objects and Classes - Abstraction - Encapsulation 	<ul style="list-style-type: none"> - To understand the concepts of encapsulation and abstraction through the abstraction of the class. 	<ul style="list-style-type: none"> - Class definitions - Configuring object - Access to members of the object
	11	<ul style="list-style-type: none"> - List 	<ul style="list-style-type: none"> - Learn how to use list. 	<ul style="list-style-type: none"> - Function list, Binary Search, Selection Sort
	12	<ul style="list-style-type: none"> - Recursion 	<ul style="list-style-type: none"> - Uses a recursive function to learn the problem solving process. 	<ul style="list-style-type: none"> - Factorial, Fibonacci, Hanoi Tower
	13	<ul style="list-style-type: none"> - Software design methods 	<ul style="list-style-type: none"> - The learning object-oriented design methods 	<ul style="list-style-type: none"> - Abstraction, Stepwise refinement, Modularization, Information hiding
	14	- Team Project		
	15	- Final exam(Team Project)		

두 번째는 자율성(Autonomicity)이다. CT 수업은 세부 주제에 대한 이론적 배경을 근간으로 하는 실습 과정이 반드시 수반되어야 하며, 실습을 수행하는 과정에서 학습자의 자율적인 참여하에 스스로 주어진 문제를 풀어 갈 수 있도록 지도해야 한다. 사전에 학습한 방법을 스스로 실습과정에 적용해 봄으로써, 학습자 개인의 기술적 능력 뿐 만 아니라 자신의 전공 분야에서 문제 해결능력을 향상 시킬 수 있다.

세 번째는 진화성(Evolvability)이다. CT에 대한 교육을 이수한 후에, 각 전공분야에서 현실적인 문제를 해결하기 위해서는 바탕이 되는 지식을 기반으로 새로운 문제를 풀기 위한 해법을 찾을 수 있어야 한다. 그러한 해법들은 학습자의 기반 지식의 진화를 통해 창의적이고 융합적인 사고방식을 배양함으로써 달성할 수 있다.

3.3 CT 교육과정 설계

대학의 기초 교양교육으로서 융합교육에 대한 중요성이 대두되고 있지만 융합 교육에 필요한 CT 수업모형은 거의 연구가 이루어 지지 않고 있으며, 이에 따른 교육과정 설계에 대한 기준 또한 없다[18-19]. 따라서 본 연구에서는 CSTA와 ISTE에서 제시한 CT의 9가지 주요 요소에 대한 내용을 본 연구의 CT 수업모형 설계의 기본요소로 활용하였다. 특히 문제 해결을 위한 규칙을 발견하고 문제에 대한 이해력과 적응력을 기를 수 있는 알고리즘과 절차에 중점을 두어 설계함으로써 IT 비전공 학습자들에게 자신의 전공 분야에서 CT를 적용할 수 있도록 하였다. 또한 실습을 위한 프로그래밍언어는 다른 프로그래밍언어에 비해 손쉽게 프로그래밍이 가능하고 CT와 연관된 요소가 많은 객체지향 프로그래밍언어인 파이썬을 선택하였고 팀 프로젝트 수행 시 대학과정상 필요한 요소들을 추가하여 표 3과 같이 CT 교육 과정을 설계하였다.

3.4 CT 수업모형

위와 같은 연구를 바탕으로 IT 융합교육을 활성화하고 IT 비전공 학습자에게 효율적인 CT교육을 수행할 수 있도록 설계한 CT 수업모형은 그림 1과 같다.

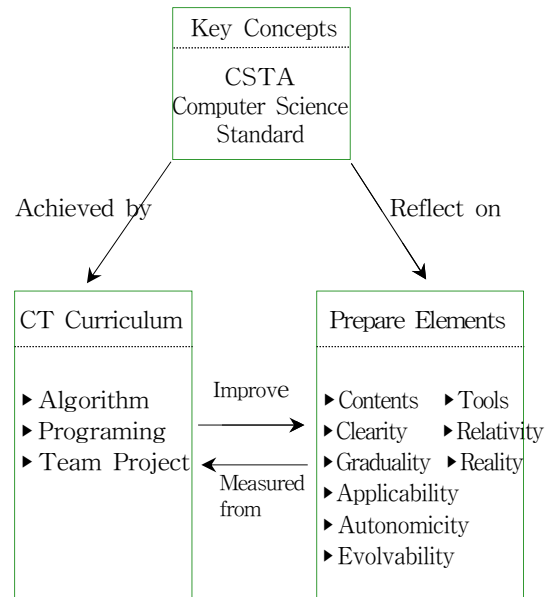


그림 1. 설계된 CT 수업모형
Fig. 1 Designed CT teaching model

CT 수업모형을 설계하기 위한 기본 방향은 첫째, 미국의 ISTE와 CSTA의 CT 교육에 대한 교육영역 및 조작적 정의를 가이드라인으로 하고 본 연구에서 제시한 CT 수업 제반요소를 반영한다. 둘째, 프로그래밍언어는 CT 융합교육에 적합하고 향후 학습자들의 전공분야에서도 적용 가능한 프로그래밍 언어를 채택한다. 셋째, 대학 교양 교육 필수과목 3학점(15주:45시간)을 기준으로 시수를 편성한다. 넷째, 교수·학습의 형태는 CT 교육의 단계별 특성에 따라 적합한 형태로 수업 한다. 학습자의 학습 동기 및 흥미를 유발하기 위하여 무료 온라인 사이트를 제시하고 융합교육의 역량강화를 위해 팀 프로젝트를 통한 자신의 작품이나 아이디어를 공유하여 상호 피드백이 가능하도록 학습 환경을 구성한다. 다섯째, 팀 프로젝트는 융합 교육을 수행하는 과정으로 학습자의 전공별 특성에 맞게 편성한다.

IV. CT 수업모형에 대한 성취기준

IT 비전공 학습자에게 CT 능력배양은 자신의 전공분야가 컴퓨터 과학적 관점에서 왜 필요한지 해결하고자 하는 문제가 무엇인지를 이해하고 새로운 문제가 발생했을 때 발생한 문제를 추상화하여 문제를 풀 수 있는 알고리즘과 프로그램을 개발할 수 있는 능력을 갖추 수 있도록 하는 것이다. 세부적인 성취기준은 표 3과 같이 1단계(1주~2주), 2단계(3주~7주), 3단계(8주~15주)로 구분 하였다.

4.1 1단계

1단계에서는 CT에 대한 개념을 인식하는 단계로서 성취기준은 다음과 같다.

첫째, CT에 대한 개념을 일상적인 예와 과학적인 측면에서 이해할 수 있다.

둘째, 인공지능 프로그램인 AlphaGo가 어떤 구조의 컴퓨터 구조에서 운영되는지 CT와는 어떤 관계가 있는지 토론을 통해서 이해할 수 있다.

셋째, CT 수업과 관련하여 교육용 프로그래밍언어인 스크래치와 객체지향 언어인 파이썬이 무엇인지 알 수 있다.

4.2 2단계

2단계에서는 CT와 연계하여 문제 해결능력을 높이고 프로그램을 효율적으로 작성하기위한 전단계로서 성취기준은 다음과 같다.

첫째, 의사코드(Pseudo Code)와 순서도(Flowchart)를 학습하여 알고리즘을 표현할 수 있으며 스택, 큐, 데큐에 대한 일상생활의 의미를 이해할 수 있고, 너비우선탐색과 깊이우선탐색의 차이점을 알 수 있다.

둘째, 알고리즘의 복잡도를 통해서 문제해결과정의 중요성을 인식할 수 있다.

셋째, 분할정복 알고리즘에서 재귀적 기법을 이해하고 응용할 수 있으며 분할정복 알고리즘이 CT에 어떻게 적용되는지 이해할 수 있다.

넷째, 동적프로그래밍 기법과 분할정복 알고리즘 기법의 차이점을 이해 할 수 있다.

다섯째, 모델링에 필요한 활동 다이어그램(Activity Diagram)과 상태 다이어그램(State Diagram)을 이해하고, 문제해결과정에 적용할 수 있다.

4.3 3단계

3단계에서는 객체지향언어인 파이썬을 사용하여 2단계에서 학습한 알고리즘 내용을 프로그램하고 팀 프로젝트 수행을 위한 준비단계로 성취 기준은 다음과 같다.

첫째, 입출력 명령과 Turtle module을 사용하여 프로그래밍 할 수 있으며 산술, 관계, 논리연산자와 조건식을 사용하여 프로그래밍 할 수 있다.

둘째, 반복구조인 while루프, for루프를 이해하고 코딩할 수 있으며 함수의 사용방법과 함수의 추상화 개념을 이해하고 프로그래밍 할 수 있다.

셋째, 객체와 클래스를 설명하고 객체를 설계하기위해 클래스를 사용할 수 있고 소프트웨어 개발의 핵심요소인 클래스의 추상화와 캡슐화 기법을 적용할 수 있다.

넷째, 선형검색과 이진검색을 코딩하고 리스트가 코딩에서 왜 유용한지 설명할 수 있으며 재귀함수의 정의와 재귀를 사용하는 이점에 대해서 설명할 수 있다. 재귀는 CT 요소와 어떠한 관계가 있는지 설명할 수 있다.

다섯째, 소프트웨어 설계 기법 인 객체지향 설계방법을 이해하고 설계원리인 추상화, 단계적 분해, 모듈화, 정보은닉의 개념이 CT 요소와 어떠한 관계가 있는지 구분하고 설명할 수 있다.

V. CT 수업모형에 대한 교수·학습 방법

CT는 일상에서 발생하는 복잡한 문제를 인지적 교육이나 SW 교육을 통해 문제해결력을 향상 시킬 수 있다. CT가 무엇이고 왜 필요한지 CT에 대한 개념을 동영상시청(동영상 수업)과 학습자의 일상생활의 예를 통해 이해하고 프로그래밍 수업을 하기 전 CT에 대한 다양한 토론(토론 수업)과 알고리즘 학습을 통하여 문제해결방법을 이해하도록 한다. 또한 교육용 프로그래밍 언어인 스크래치와 관련된 다양한 소프트웨어를 실행하여 프로그래밍에 대한 관심과 흥미를 갖게 하고 객체지향 프로그래밍언어인 파이썬을 통해 프로그램 실습수업을 실시한다. 수업에 대한 이해와 관심을 높이기 위해서 교육용 온라인 사이트를 방문하여 교육에 활용할 수 있도록 한다. 교육과정에 대한 교수·학습 방법을 세단계로 구분하였다.

5.1 1단계

1단계(1주~2주)에서는 CT가 무엇이고 왜 필요한지 학습자에게 CT에 대한 개념을 동영상 수업을 통해 인식하도록 한다. 특히 그룹 토론을 통해서 CT의 컴퓨터 과학적인 예와 일상생활에 대한 경험을 발표하게 함으로써 CT를 이해할 수 있도록 하고 파이썬 온라인 사이트를 방문하여 파이썬의 성공 스토리를 시청하고 스캐치 온라인 사이트에서 완성된 많은 예제프로그램 실행을 통해서 CT 수업에 대한 흥미와 관심을 갖도록 유도한다.

5.2 2단계

CT 교육은 사고하는 방법과 기술이 포함되어져야 하는데 2단계(3주~7주)에서는 표 3의 내용과 연계하여 문제해결력을 향상할 수 있는 알고리즘을 학습시킨다. 알고리즘을 학습하기 위해서는 알고리즘과 관련된 프로그램을 한다든지 실제 예제프로그램을 응용할 수 있어야 하지만 프로그램 교육이 이루어지지 않은 상태에서 초보자에게는 어려운 게 사실이다. 따라서 알고리즘을 표현하는 방법으로 의사코드와 순서도를 학습하여 알고리즘을 표현할 수 있도록 한다. CT와 관련된 내용은 학습하기 전 일상생활에서 발생하는 예를 반드시 생각하도록 한다. 알고리즘은 선형구조와 비선형 구조로 나누어 학습하고 선형구조 알고리즘과 비선형구조 알고리즘의 차이점을 확인한다. 알고리즘의 복잡도 학습은 동일한 일에 대해서 복잡도를 줄이면서 질을 향상시키는 작업을 탐색하는 것이다. 선택정렬과 퀵정렬, 순차검색과 이진검색을 준비된 예제프로그램을 수행하여 학습자는 수행결과를 확인하고 동일한 문제를 해결하는데 사용되는 서로 다른 알고리즘의 효율성에 대해서 이해하고 문제를 해결하는데 사고하는 방법과 절차에 따라 효율성이 달라질 수 있다는 것을 인식하도록 한다. CT는 패턴이나 추이를 발견하는 것이다. 즉 규칙이나 원칙을 통찰에 의해서 패턴이나 추이를 분석하고 추상화를 통해서 일반화하는 것이다. 분할정복알고리즘은 대부분 재귀적으로 구현되기 때문에 CT를 이해하는 중요한 기법이라고 할 수 있다. 퀵정렬과 이진검색의 의사코드를 작성하게 하여 분할정복법과 재귀적 호출을 이해하고 CT와의 연관성을 확인할 수 있도록 한다. 동적프로그래밍 기법과 분할정복 알고리즘 기법의 차이점을 분석 하도록 한다. 추상화

는 문제해결과정에서 불필요한 요소를 무시하고 간소화하는 과정이기 때문에 추상화와 분해, 병렬화의 연관성을 일상생활에서 찾아 토론을 통해 CT를 학습하도록 한다. 또한 추상화와 분해를 지원하는 모델링 방법인 활동 다이어그램과 상태 다이어그램을 학습하여 CT와 팀 프로젝트에 활용할 수 있도록 한다.

5.3 3단계

CT 능력 배양은 소프트웨어 교육을 통해서 가능하다. 3단계에서는 IT 비전공 학습자들에게 흥미를 부여 하면서 다른 프로그래밍언어에 비해 단순하면서도 직관적인 문법구조로 학습하기 쉽고 다양한 응용프로그램을 작성할 수 있는 파이썬 프로그래밍 도구를 활용하여 문법보다는 실습을 통한 문제 해결력에 초점을 둔다. 파이썬은 객체지향언어이기 때문에 CT 요소들을 포함하고 있어 2단계에서 학습한 알고리즘 내용과 문제해결 과정을 프로그래밍을 통해서 실습 한다. 프로그램은 난이도가 낮은 것, 중간 수준의 것을 프로그램하게 하고 난이도가 높은 프로그램은 문제해결과정을 단계별로 정리 기술하는 과정을 통해 학습하는 시간을 할애 하도록 한다. 학습자들이 효율적으로 팀 프로젝트를 수행하기 위한 소프트웨어 설계기법 중 객체지향 설계방법을 학습한다. 설계원리인 추상화, 단계적 분해, 모듈화, 정보은닉의 개념을 이해하도록 한다. 팀 프로젝트의 활동은 CT 수업을 통한 IT 융합교육을 달성하는 중요한 과정이다. 팀 프로젝트 안내 및 평가 계획은 홈페이지에 구체적으로 공지한다. 팀의 구성은 프로젝트 수행 시 전공에 관련된 내용을 CT에 적용하여 전공분야의 융합적 사고를 배양하기 위해 비전공학생자(인문,사회,예체능계열) 학과중심으로 4-5명으로 구성한다. 두 번의 팀 프로젝트를 통해 전공분야에 대한 자료를 수집 분석하는 능력을 키우고 다양한 프로그래밍기법을 학습하게 된다.

VI. 결론

현재와 미래사회는 ICT를 활용하여 동시적이고 복합적인 활동을 할 수 있는 디지털 정보화 사회로 많은 나라들은 CT를 기반으로 문제 해결력을 갖춘 창의적이고 융합적인 인재 양성을 위해 온 힘을 쏟고 있다.

이에 대학에서도 융합 교육의 역량을 강화하기 위해 IT 비전공자를 대상으로 CT 교육을 진행하고 있으며 더욱 확산되는 추세이다. 본 연구에서는 CT 교육을 통해서 IT 융합교육을 활성화하고 비전공학습자의 IT 융합 능력과 CT 능력을 배양하여 학습자의 전공분야에서도 학습한 내용이 응용될 수 있도록 CT 수업모형을 설계하였다. 본 연구결과의 활용방안은 다음과 같다. 첫째, IT 비전공 학습자의 CT 교육과정 활용이다. 본 연구에서 설계한 CT 교육과정은 문제에 대한 이해력과 적응력을 키울 수 있는 알고리즘 분야와 학습자의 전공분야에서 CT 능력과 SW 적응력을 높이기 위해 프로그래밍 실습, 팀 프로젝트 수업으로 설계되어 대학의 IT융합 교육을 위한 CT 교육과정에 활용될 수 있을 것이다. 둘째, 제시된 CT 수업의 제반요소로서 준비요소, 운영요소, 전환요소는 교수자와 학습자간의 CT 수업의 효과를 높이고 성취도를 높이는데 활용할 수 있을 것이다. 셋째, CT 수업모형에 대한 교수·학습방법은 CT에 기초한 구체적인 교육 과정을 통해 교수자가 학습자에게 무엇을 어떻게 교육해야 하는지 CT 수업 방향을 설정하는데 활용할 수 있다.

본 연구에서는 설계된 CT 수업모형을 일반화하기 위한 유효성은 검증하지 못했지만 CT 수업모형에 대한 프로토타입을 설계하는데 의의를 두고자 한다.

References

- [1] E. Sung, H. Oh, and Y. Kim, "An Instruction Model Design of Convergence Project for Cultivating Industrial Convergence Talent in Higher Education," *J. of Korea Educational Methodology Studies*, vol. 25, no. 3, 2013, pp. 543-580.
- [2] K. Kim and K. Jang, "A Discussion on the Concert of Convergence for STEAM-Approach in Music Education," *J. of the Korea Elementary Education*, vol. 26, no. 4, 2015, pp. 211-234.
- [3] S. Choi, "An Analysis of 'Informatics' Curriculum from the Perspective of 21st Century Skills and Computational Thinking," *J. of Korean Association of Computer Education*, vol. 14, no. 6, 2011, pp. 19-30.
- [4] H. Jung, "An Empirical Study on Information Liberal Education in University based on IT Fluency and Computational Thinking Concept," *J. of the Korea society of computer and information*, vol. 19, no. 2, 2014, pp. 263-274.
- [5] K. Kim, "A Case Study on Necessity of Computer Programming for Interdisciplinary Education," *J. of Digital Convergence*, vol. 2, no. 11, 2014, pp. 339-348.
- [6] K. Park, K. OH, N. Ryu, H. Lee, and E. Kim, "Learning System of Programming Language using Basic Algorithms," *J. of the Korea Institute of Electronic Communication Science*, vol. 5, no. 1, 2010, pp. 66-73.
- [7] J. Wing, "Computational Thinking," *Communications of the ACM*, vol. 49, no. 3, 2006, pp. 33-35.
- [8] J. Wing, "Computational Thinking and Thinking about Computing," *Philosophical Transactions of the Royal Society*, vol. 366, 2008, pp. 3717-3725.
- [9] M. Guzdial, "Education : Paving the way for computational thinking," *Communications of ACM*, vol. 51, no. 8, 2008, pp. 25-27.
- [10] B. Kim, Y. Jeon, J. Kim, and T. Kim, "Development and Application of Real Life Problem Solving Lesson Contents Based on Computational Thinking for Informatics Integrated-Gifted Elementary School Students' Creativity," *J. of Teacher Education*, vol. 31, no. 1, 2015, pp. 159-186.
- [11] I. Jeon, "Study on the Achievement Goals and Teaching-Learning Methods of Problem Solving Topic of Informatics Subject," *J. of The Korean Association of Information Education*, vol. 18, no. 2, 2014, pp. 243-254.
- [12] T. Hyun, J. Choi, and J. Lee, "Development of Teaching Model for Problem-solving methods and procedures section in the 2012's revised informatics curriculum," *J. of the Korea Society of Computer and Information*, vol. 17, no. 8, 2012, pp. 17-28.
- [13] B. Joyce and M. Weil, *Model of teaching*, New York: Prentice Hall, 1980.
- [14] H. Choi, "Developing Lessons and Rubrics to Promote Computational Thinking," *J. of The Korean Association of Information Education*, vol. 18, no. 1, 2014, pp. 57-64.
- [15] S. Kim and S. Han, "Design-Based Learning for Computational Thinking," *J. of the Korean Association of Information Education*, vol. 16, no. 3, 2012, pp. 319-326.

- [16] M Ryu and S. Han, "Development of Computational Think-based Educational Program for SW Education," *J. of The Korean Association of Information Education*, vol. 19, no. 1, 2015, pp. 11-20.
- [17] S. Kim, "Effects of Teaching and Learning Strategies of Learner-Centered Learning for Improving Computational Thinking," *J. of The Korean Association of Information Education*, vol. 19, no. 3, 2015, pp. 323-332.
- [18] C. Choi and S. Joo, "A Curriculum Design of Computer Application Department for Non-Commissioned Officers," *J. of the Korea Institute of Electronic Communication Science*, vol. 9, no. 5, 2014, pp. 583-588.
- [19] S. Choi, "A Design of the Database Curriculum for College Students," *J. of the Korea Institute of Electronic Communication Science*, vol. 10, no. 5, 2015, pp. 599-606.

저자 소개



손영수(Young-Su Son)

1988년 조선대학교 전산통계학과
졸업(이학사)

1991년 조선대학교 전산통계학과
졸업(이학석사)

1999년 조선대학교 전산통계학과 졸업(이학박사)

1996년~현재 한려대학교 전기전자공학과 교수

※ 관심분야 : 프로그래밍어, 인공지능, 컴퓨터교육



이광재(Kwang-Jae Lee)

1986년 전북대학교 전자공학과
졸업(공학사)

1990년 전북대학교 대학원 전자
공학과 졸업(공학석사)

2006년 전북대학교 대학원 전기공학과 졸업(공학
박사)

1995년~현재 한려대학교 전기전자공학과 교수

※ 관심분야 : 부호이론, 이동통신

