Journal of Digital Convergence
Vol. 18. No. 3, pp. 35-42, 2020

ISSN 2713-6434 / eISSN 2713-6442
https://doi.org/10.14400/JDC.2020.18.3.035

디지털 트랜스포메이션 시대의 언플러그드를 적용한 컴퓨팅 사고력에 대한 효과성 분석

이명숙
계명대학교 타불라라사칼리지 교수

Effectiveness Analysis of Computing Thinking with Unplugged in Digital Transformation

Myung-Suk Lee
Professor, Tabula Rasa College, Keimyung University

요 약 디지털 트랜스포메이션은 가상과 현실간의 상호작용을 혁신하는 것이다. 이 과정에서 발생하는 복잡한 문제를 해결해야 하고, 그 방법 중 하나가 컴퓨팅 사고력이다. 따라서 본 연구는 대학에서 교양교육으로 언플러그드를 이용한 소프트웨어 교육이 컴퓨팅 사고력을 향상시키는데 효과가 있는지 살펴보는 것을 목표로 한다. 그 방법으로 전체 학년을 대상으로 한 교양 소프트웨어 수업에 컴퓨팅 사고 5개 요소를 추출한 후 언플러그드를 적용하여 수업을 진행하였다. 한 학기 16차시 수업을 진행하였고 설문지를 통해 컴퓨터 사고력 향상을 측정하였다. 그 결과 수업 이후 수강생의 컴퓨팅 사고력이 전반적으로 증가하였다. 관찰조사에서는 전 학문분야에서 컴퓨팅 사고력 요소 중 추상화 요소를 개념화하는데 어려움을 느꼈으며, 예·체능은 알고리즘 요소를, 인문학은 패턴인식 요소를 개념화하는데 더 어려움을 느꼈다. 향후 각 요소를 학문영역별로 다양한 내용의 콘텐츠를 개발하여 학습자에게 이해를 돕는 것이 필요하다.

주제어 : 컴퓨팅 사고, 언플러그드, 소프트웨어교육, 비전공자, 하이브리드교수법, 디지털 트랜스포메이션

Abstract Digital transformation is about revolutionizing the interaction between virtual and reality. The complex problems that arise in this process must be solved, and one of the methods is computing thinking. Therefore, this study aims to observe whether software education that uses unplugged as liberal education is effective in enhancing computing thinking. For this, 5 elements of computing thinking were extracted and unplugged was applied to liberal software classes, and classes were conducted. During one semester, 16 sessions of classes were conducted and computing thinking enhancement was measured through surveys. As a result, the computing thinking of the students increased overall after classes. Observation surveys showed that, among computing thinking elements, students of all academic fields felt difficulties conceptualizing abstraction elements, those of arts and physical education felt more difficulties with algorithm elements, and those of the humanities felt more difficulties with pattern recognition elements. In the future, various contents for each element should be developed by academic field to aid learner understanding.

Key Words : Computational Thinking, Unplugged, Software Education, Non-Majors, Hybrid Teaching, Digital Transformation

*This research was supported by the Bisa Research Grant of Keimyung University in 2019
*Corresponding Author : Myung-Suk Lee(mslee@kmu.ac.kr)
Received January 30, 2020 Revised March 6, 2020
Accepted March 20, 2020 Published March 28, 2020

1. 서론

최근 세계 각국의 많은 산업들이 소프트웨어 중심 산업으로 빠르게 변화하고 있다. 디지털 트랜스포메이션으로 변신을 시도하고 있고 그 선두에 선 기업이 우리가 제조회사로 알려져 있는 제너럴 일렉트릭(GE)이다. GE는 더 이상 산업장비 신제품을 내세우는 것이 아니라 프리딕스(Predix)란 소프트웨어 플랫폼을 핵심 제품으로 강조하고 있다[1].

디지털 트랜스포메이션은 가상의 디지털 데이터/정보를 활용해서 현실의 제품/서비스 등에 적용되어 이들 간의 상호작용을 혁신하는 것이다. 이러한 패러다임 변화로 문제를 접근하는 우리의 사고 전환도 필요한 시기이다.

디지털 트랜스포메이션 시대의 인재는 창의력, 문제해결 능력, 논리 사고력, 그리고 융합적 사고력을 필요로 한 이유이기도 하다[2]. 이에 세계적으로 소프트웨어 교육을 국가 차원의 교육과정을 마련하여 소프트웨어 교육을 강화하는 등 소프트웨어 교육의 중요성을 인식하고 있다.

소프트웨어 교육을 하는 목적은 컴퓨터를 통해서 실제 현실에서 일어나는 복잡 다양한 문제들을 가장 효율적인 방법으로 해결할 수 있는 절차적 과정, 즉 논리적 사고능력을 키우는데 그 의의가 있다. 소프트웨어 교육이 주입식 교육 보다 사고력 중심으로 구성해야 하는 이유이기도 하다[3]. 사고력 중심의 교육으로 되기 위해서는 컴퓨팅 사고 교육이 이루어져야한다. 컴퓨팅 사고가 미래 핵심역량이기도 하지만[4], 급변하는 사회에 대응하기 위해 문제를 창의적으로 해결하는데 있어 컴퓨팅 사고력을 적용할 수 있는 인재를 확보하는 것이 곧 국가경쟁력을 좌우하게 될 것으로 예상하기 때문이다[5].

이미 미국, 핀란드, 영국, 이스라엘 등 해외에서는 컴퓨팅 능력의 중요성을 인식하고 컴퓨팅 사고를 교육과정에 도입하기 위한 다양한 노력을 기울여 왔다. 미국은 새로운 국가수준의 과학교육과정 표준을 연구할 때 과학적 개념과 원리 중심의 학습에서 개념과 원리를 활용한 모델을 구현하는 공학적 실천으로 교육의 방향을 확장하고 있고, 이러한 변화 과정에서 컴퓨팅 사고를 활용한 공학적 실천에 관련된 내용을 교육과정에 포함하고 있다[6].

한국은 2018년부터 초·중·고등학교에서 소프트웨어 교육을 정규 교육과정으로 운영하고 있고 대학에서도 소프트웨어 중점대학을 중심으로 대부분의 대학에서 교양과목으로 소프트웨어 과목을 운영하고 있다[7]. 소프트웨어 중점대학은 모든 학생들이 소프트웨어 교육을 3~9학점 이수한 후에 졸업을 하고 그렇지 않은 학교는 학교 정책에 따라 인문/사회/예·체능 계열 학생에 대한 소프트웨어 교육을 접하지 못한 채 소프트파워가 중심이 되는 사회로 나아가야 한다. 이에 많은 대학에서 비전공자를 중심으로 다양한 교수법을 적용하여 소프트웨어교육을 하고 있지만 프로그램 위주의 교육으로 큰 효과를 보지 못하고 있다.

인문·사회, 예·체능 계열 등 비전공자들이 컴퓨팅 사고 교육이 필요한 이유는 산업, 교육, 경제, 체육 등 모든 분야에서 소프트웨어를 기반으로 동작되고 소프트웨어가 적용 되지 않는 분야는 거의 찾아보기가 어렵다. 이에 비전공자들도 미래의 인재로서 다양한 역량을 길러주어야 하며, 학습자들의 컴퓨팅 사고력, 문제해결력, 창의성을 길러주는 방향으로 설정되어야 한다[8]. 그러나 컴퓨팅 사고 교육이 프로그램 교육으로 일관되어서는 안된다.

따라서 본 연구에서는 컴퓨팅 사고 교육에서 프로그램 교육이 아닌 언플러그드로 과학의 원리를 쉽게 접근할 수 할 수 있는 방법을 제안하고 컴퓨팅 사고력 효과를 측정 하고자 한다. 이를 위해 언플러그드와 하이브리드 교수법을 이용하여 교수자는 수업의 방향과 핵심내용을 제시하고, 학습자는 문제해결 과정을 자기 주도적으로 컴퓨팅 사고력을 키워가는 것은 철저하게 학습자 자신이 되도록 하였다. 이 연구를 통해 컴퓨팅 사고의 핵심 요소를 추출하고 이 요소와 학습 내용을 연결한다. 이는 역량 중심의 문제해결을 위한 컴퓨팅 사고 교육을 개발하는데 방향을 제시할 있을 것으로 기대한다.

2. 배경연구

2.1 선행연구 검토

'언플러그드'를 주제어로 RISS에서 검색한 결과 Table 1과 같이 23편의 논문이 검색되며, '컴퓨팅 사고'를 키워드로 검색한 결과 353편의 학술논문이 검색된다. '언플러그드' 논문은 많은 양의 논문이 개발되지 않고 특히 대학에서 응용 사례는 거의 찾기 어렵다. '컴퓨팅 사고' 주제의 논문은 꾸준히 증가되는 것을 볼 수 있다. 그 이유는 소프트웨어 중점대학에서 기본 교과목으로 '컴퓨팅사고력', '논리적사고력', 'SW기초교육' 등이 필수 교양교육 과목으로 지정되어 있기 때문이다[7].

이수진(2018)[9]은 비전공자 대학생들의 컴퓨터로 소통하기 위해 컴퓨터 사고를 이해하기 위한 방법을 모색하였고 1주일에 2시간은 물리적 환경의 한계가 있었으며, 컴퓨팅 사고 교육이 기초 교육으로 프로그래밍 교육

Table 1. Keyword Search Papers on RISS

subject	2019	2018	2017	2016	2015	before 2014	sum
Unplugged	5	4	3	1	1	9	23
CT	81	76	37	31	18	110	353

에 집중하는 것에 문제를 지적하였다. 정선호(2019)는 대학의 신입생을 대상으로 컴퓨팅 교양 수업에서 다양한 문제를 해결하는 과정에서 컴퓨팅 사고력 향상에 효과가 있는지 검증하였다. 5개의 요인으로 범주화하여 측정한 결과 4개의 요인인 패턴인식, 분해, 알고리즘, 프로그래밍은 높게 나타났으나 1개의 요인인 추상화는 기준치보다 낮게 나타나는 결과를 보였다. 컴퓨터 사고력 관련 교육이 점점 확대되고 있는 시점에 특정 요소만으로 구분하는 것에 한계가 있으며 설문 문항을 통하여 컴퓨팅 사고력 구성요소에 대한 측정은 할 수 있다고 하더라도 컴퓨팅 사고력 역량 그 자체를 측정하는 것은 쉽지 않기 때문에 많은 연구가 필요하다고 하였다. 구진희(2017)[10]는 비전공자들에게 소프트웨어 교육을 위한 앱 인벤터 교육과정을 설계하였고, 이 연구는 피지컬 컴퓨팅, 로봇 프로그래밍 등 다양한 형태의 컴퓨팅 사고에 초점을 둔 교육과정 개발 연구에 활용될 수 있다고 하였다. 한희섭 외(2009)[11]은 정보관련 예비교사들에게 ICT 소양교육을 위한 프로그램을 개발하여 적용하였고 그 결과 이해 및 자신감 향상에 큰 영향을 보였고, 주의할 점으로는 잘못된 개념의 형성은 컴퓨터적 상황을 제시해주는 교수·학습활동을 설계하는 것이 중요하다고 하였다.

언플러그드에 대한 연구들을 살펴보면 첫 번째는 초·중·고등학교에서 컴퓨터 프로그래밍 교육을 어떻게 할 것인가를 교육모형, 설계를 중심으로 이루어졌고 두 번째는 학업성취도를 통해 프로그램 능력이 얼마나 향상되었는가를 측정하는 학업성취도에 대한 연구, 세 번째는 컴퓨터 사고가 얼마나 향상되었는가를 측정하는 연구들로 이루어져있다. 따라서 본 연구는 대학의 소프트웨어 교양 수업에서 언플러그드를 적용하여 컴퓨팅 사고력을 향상시키는데 이용하고자 한다.

2.2 컴퓨팅 사고력의 구성 요소

컴퓨팅 사고력의 교육 목적은 복잡한 문제를 창의적으로 문제를 해결하는데 있어 컴퓨팅 사고를 기반으로 효율적인 문제 해법을 찾는 것이다. Wing(2006, 2008)[12]은 컴퓨팅 사고의 핵심은 추상화와 자동화라고 밝힌 바 있으며, Barr와 Stephenson(2011)[13]은 자료수집, 자료분석, 자료제시, 문제분해, 추상화, 제어구조, 알고리즘 및 절차, 분석과 모델 타당화, 자동화, 병렬화, 시험 및 검증, 시뮬레이션의 12가지 단계로 세분화하여 제시하였다. 미국의 과학 교사 협회(CSTA: Computer Science Teachers Association)는 컴퓨터 사고를 데이터 수집, 데이터 분석, 데이터 표현, 분해, 추상화, 패턴인식, 알고리즘, 자동화, 시뮬레이션 및 병렬화 등을 포함하는 문제 해결 과정 요소로 설명하고 있다[14]. 교육부에서는 CSTA에서 제시한 문제해결을 위한 컴퓨팅 사고 단계를 이용하여 2015 교육부 소프트웨어 교육 운영지침을 통해 자료수집, 자료분석, 구조화, 추상화, 자동화 단계로 분류하였다. 김진숙외(2015)[15]의 연구에서는 핵심 4개의 요소로 구성하여 DPAA(P)모델로 재정의 하고 있다.

본 연구에서는 이를 재정의하여 교육부지침에서 자료수집, 자료분석, 구조화 부분의 문제발견 요소로 정의하고 DPAA모델의 프로그래밍을 제거하여 5단계로 재정의 하였다. 그 이유는 비전공자는 문제발견 요소과정을 거치지 않으면 분해부터 이해하는 것을 어려워하였고 실제 코딩도 16차시의 주어진 시간 한계에 부딪혀 어려웠다. 그 대신 알고리즘 요소에서 문제해결 방법을 최대한 자연어로 상세히 표현하게 하였고 Python언어로 바꿔주는 작업에 이해도가 더 높았다. 이에 Table 2와 같이 문제발견, 분해, 패턴인식, 추상화, 알고리즘 요소로 컴퓨팅 사고 요소를 재정의 하였다.

2.3 언플러그드 컴퓨팅(Unplugged Computing)

언플러그드는 컴퓨터 없이 컴퓨팅 사고를 향상시킬 수 있는 학습 활동이다. 주로 놀이를 통하여 과학의 원리를 이해하는 학습 활동으로 초·중등 학생에게 많이 사용되나 소프트웨어에 대한 지식이 전혀 없는 비전공 학생에게도 매우 효과적이다. 주로 본격적인 프로그램 수업 전에 원리를 이해하기 위한 방법으로 상황과 문제를 제시하기, 문제 해결하기, 실생활 적용의 단계로 진행하고 있다. 언플러그드 활동은 두 곳에서 활발하게 연구하고 있다. Tim Bell 교수팀에서 개발한 CS Unplugged는 '이진수 표현, 패리티 비트, 픽셀'과 같은 컴퓨터 과학 언플러그드 활동들을 주로 연구하고 있다. Computing at school(CAS)에서는 'C.T'를 주로 연구하고 있다. 언플러그드는 수업이 놀이로 끝나지 않고 놀이를 통해 원리를 파악할 수 있도록 교육과정을 구성하는 것이 중요하다[16].

언플러그드 교육은 한꺼번에 학교 전체 학생을 대상으로 소프트웨어 수업이 진행될 경우에 모든 학생이 이용할 수 있는 컴퓨터실을 구비하기 쉽지 않다. 이런 경우

Table 2. Computing Thinking Components

	J. Wing (2008)	CSTA & ISTE (2011)	Google(2015)	Ministry of Education Guidelines(2015)		DPAA(P)Model(2015)	Proposed Model
CT Item	Abstraction	Data collection	Data collection	Data collection			Problem found
		Data analysis	Data analysis	Data analysis			
			Pattern Recognition				
		Data presentation	Data presentation	Structured			
		Problem Decomposition	Decomposition	Abstraction	Decomposition	Decomposition	Decomposition
		Abstraction	Pattern generalization		Modeling	Pattern Recognition	Pattern Recognition
			Abstraction				
	Automation	Algorithms and Procedures	Algorithm Design		Algorithm	Abstraction	Abstraction
		Automation	Automation	Automation	Coding	Algorithm	Algorithm
		Simulation	Simulation		Simulation	(Programming)	
		Parallelism	Parallelism	Generalization			

다양한 방법으로 소프트웨어 교육이 가능한데 특히 과학의 원리를 이해하기 어려운 학생들에게 매우 효과적이다. 따라서 본 연구에서는 급변하는 대학의 교육정책에 맞춰서 갖춰지지 않은 기존의 교육환경에 서도 가능한 언플러그드 방식으로 다양한 학문분야에 컴퓨팅 사고력을 향상시키는데 접목하고자 한다.

2.4 하이브리드 교수법

이명숙(2017)[17]은 온·오프라인 통합교육 모델인 하이브리드 교수법을 연구하였다. 하이브리드 교수법은 형식학습과 무형식학습으로 구분하였다. 형식학습에서는 MOOC를 예습/복습용으로 활용한다. 형식학습 후 해당 주제에 대해 학습자들은 무형식학습, 즉 자유롭게 경험/성찰학습을 한다. 이것은 학습자가 주도적으로 온라인과 오프라인에서 자유롭게 경험하고, 경험에 대한 성찰을 통해 학습하는 단계이다. 하이브리드 교수법에서 학습자 경험학습인 무형식 학습을 형식학습으로 포함시켜 심화학습으로 구체화 하였다.

하이브리드 교수법은 지식습득 단계, 사고확장단계, 경험/성찰 단계로 구분하여 학습을 진행한다. 지식습득 단계에서는 기초학습이 이루어지는 단계로 영상을 활용하거나 교수자가 직접 티칭이 가능한 단계이다. 사고확장 단계에서는 지식습득 단계로 학습자들이 중심이 되어 토론을 전개하고, 토론에 필요한 자료를 링크하여 팀원들 간에 정보를 공유한다. 이를 통해 동료 학습자들 사이에 학습이 이루어진다. 특히 프로젝트 기반학습은 학습자중심학습을 촉진하는 방법으로 효과적이다. 경험/성찰 단계는 학습자가 실생활에서 수업내용을 적용해보는 단계이다. 경험학습은 학습자들이 관련된 아이디어, 콘텐츠의 의미, 이해를 돕기 위해 과거의 경험과 연결 짓고 아이디어를 개발하는데 초점을 맞춘다. 학습자들은 학습에 지속적인 동기부여를 위해 교육과정과 실세계 사이의 명확한 연관성을 찾고, 실세계에서 경험하고 새로운 아이디어를 창조하는 역할을 한다.

하이브리드 교수법 모델은 경험/성찰 단계는 실생활과 연관성을 찾기 위해 현장에 가야하는 어려움이 있어 정규수업에서 적용하기가 쉽지 않다. 따라서 본 연구에서는 이 모델을 변형해서 적용하고자 한다.

3. 제안된 교수·학습 모델

본 연구에서 제안된 온·오프라인 통합교육 모델은 하이브리드 교수법[17]을 변형하여 Fig. 1과 같이 완성하였다. 하이브리드 교수법에서 무형식학습을 없애고 기초학습, 학습자 주도학습, 심화학습으로 구성하였다. 기초 학습에서는 지식습득단계로 기초 지식을 MOOC 영상을 통해 습득한다. 사고확장 단계에서는 학습자 주도로 습득된 기초지식을 동료학습법을 통해 팀별로 이해과정을 거친다. 전체적으로 이해되지 않는 부분은 교수자가 전체학생에게 티칭 한다. 심화학습은 경험/창의 단계로 배운 내용을 형식학습 후 해당 주제에 대해 학습자들은 무형식학습, 즉 자유롭게 경험/성찰학습을 한다. 이것은 학습자가 주도적으로 온라인과 오프라인에서 자유롭게 경험하고, 경험에 대한 성찰을 통해 학습하는 단계이다.

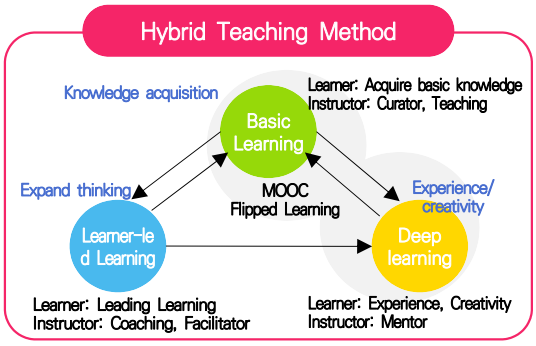


Fig. 1. Proposed teaching and learning model

Table 3. Learning and class activity steps

Learning	Class activities	Explanation
Pre-class	Learner level diagnosis	Diagnose prior learning and set learning goals. Learners who lack basic knowledge come to class with a significant learning deficit at the beginning of the lesson, so they should be aware of the learning deficit.
	Basic Learning Supplement	Compensate for the prior learning deficits found through basic education diagnosis. Provide basic Python Training through flip learning.
In-class	Specify class goals	Clearly present and recognize the learning goals of the unit. It presents the learning goals by connecting the Python learning stages through flip learning with the computing thinking stages.
	Execute class	Conduct real lessons, increase learning density and provide ample learning opportunities. Identify and feedback learners' flip-learning content.
	Formative evaluation	Evaluate understanding based on what is learned in flip learning. This assessment provides the learner with feedback about their progress and assesses the learner's level and applies it to the class.
	Deep learning	In order to improve learners' comprehension of lessons, classes are conducted by linking flip-learning lessons with computing thinking. Through the lessons, practice, experiments, etc., students go through the expansion stages of thinking.
After-class	Applied Learning	Find out what you learned in class in real life and present and apply it in the next class.
	Comprehensive Evaluation	At the end of the class, learners are assessed for their effectiveness by comprehensively assessing their academic achievement. Comprehensive assessments are given at the end of the semester and reflected in the next semester.

온·오프라인 통합교육 모델에서 학습은 Pre-class, In-class, After-class 단계로 이루어지며 Table 3과 같다. Pre-class는 주로 플립러닝 방법으로 기초 학습을 하였다. 첫 시간은 선행학습을 진단하고 수업에 대한 의지를 파악한다. 이 수업을 듣기 위한 기초학습이 불충분할 경우 영상으로 기초 지식을 학습할 수 있도록 기초 학습 보충 과정을 두었다. In-class에서는 명확한 수업목표 제시를 해야 하며, 수업전개에서는 실제 수업이 이루어지는 것으로 플립러닝 내용으로 충분한 학습 기회를 제공한다. 형성평가는 Pre-class 단계에서 이루어진 플립러닝과 관련하여 확인하는 절차를 거치고 심화학습에서 컴퓨팅 사고 요소에 맞는 5개의 요소와 연결된 실습, 실험, 활동 등의 수업이 이루어진다. After-class는 수업시간에 배운 내용을 주변 생활에서 찾아서 다음 수업시간에 수업 내용과 연계하여 질문과 토론 또는 활동으로 진행된다.

4. 연구방법

4.1 연구대상

본 연구는 대구소재 4년제 사립대학에서 컴퓨팅 사고력 향상을 목적으로 소프트웨어 교양 수업에 대하여 언플러그드를 적용한 수업이 컴퓨팅 사고력 향상에 도움을 주는지 확인하는 것을 목적으로 한다.

교양수업으로 전 학년을 대상으로 모든 전공계열과 학년 구별 없이 들을 수 있는 수업이므로 수업에 대한 기초지식의 차이가 매우 큰 편이다. 이를 해결하기 위한 방법으로 하이브리드 교수법을 적용하였고 기초지식 습득 단계에는 플립러닝을 활용하였다. 수업 참여자 68명 모든 응답자가 응답하였고 표본의 통계학적 특성은 Table 4와 같다.

Table 4. General characteristics of respondents

Major \ Gender	Responder Configuration		
	Male	Female	sum
Humanities and Social	10 (15%)	6 (9%)	16 (24%)
Nature	4 (6%)	4 (6%)	8 (12%)
Engineering	18 (26%)	20 (29%)	38 (56%)
Medical	0 (0%)	2 (3%)	2 (3%)
Art and Physical Education	2 (3%)	2 (3%)	4 (6%)
sum	34명	34명	68명

4.2 분석방법

본 논문에서는 교양수업 수강생들을 대상으로 컴퓨팅 사고력 측정 도구의 변인과 관련하여 설문을 통해 수업의 효과성을 분석하기 위한 양적 분석 방법을 사용하였다. 분석 설문지는 한영신(2018)[18]에서 개발한 신뢰도가 확보된 설문 문항을 본 수업에 맞게 변경하여 사용하였다.

자료 처리방법은 리커트 4점 척도로 구성하였으며 '매우그렇다'는 4점, '그렇다'는 3점, '그렇지않다'는 2점, '매우그렇지않다'는 1점으로 하여 분석하였다. 설문조사의 결과는 IBM SPSS Statics 프로그램을 사용하였다. 언플러그드를 이용한 플립러닝 기반의 프로그래밍 효과성 측정은 기술 통계, 컴퓨팅 사고력 향상 측정은 대응표본 t-검정(Paired t-Test)을 통해 통계적으로 검정하였다.

5. 컴퓨팅 사고력 향상에 대한 분석 결과

본 연구에서는 플립러닝 기반의 언플러그드 교육 전-후의 컴퓨팅 사고력의 향상 정도를 측정하기 위해 사전·사후 검사를 시행하였다. 설문 도구는 총 25문항으로 플립러닝 기반의 언플러그드 교육의 효과성을 측정하기 위한 15문항, 컴퓨터 사고력 향상을 측정하기 위한 10문항으로 구성되어 있다. 컴퓨팅 사고력 향상을 측정하기 위한 문항은 인지수준(Cognitive Level) 영역과 해결 능력(Ability to Solve) 영역으로 구성된다.

5.1 계열별 t-검정 결과

플립 러닝 기반의 언플러그드 학습이 컴퓨팅 사고력에 대한 변화를 살펴보기 위해 프로그래밍 교육 전후에 검사를 시행하였다. 검사는 동일한 설문지로 시행하였으며 리커트 4점 척도로 설문 결과 값에 대한 대응표본 t-검정을 수행하였다. 전공계열별 전체 영역에 대한 대응표본 t-검정의 분석내용은 Table 5와 같다.

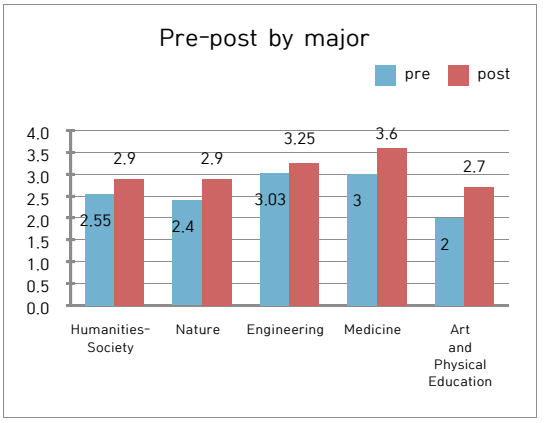
Table 5. T-test results by major

Major	Domain	Survey	*N*	*M*	*SD*	*t*	*p*
Humanities and Society	Cognitive Level & Ability to Solve	Pre	16	2.45	.245	-8.2825	.0000
		Post	16	2.90	.200		
Nature	Cognitive Level & Ability to Solve	Pre	8	2.40	.469	-4.6291	.0189
		Post	8	2.90	.683		
Engineering	Cognitive Level & Ability to Solve	Pre	38	2.95	.287	-4.9182	.0001
		Post	38	3.18	.227		
Medician	Cognitive Level & Ability to Solve	Pre	2	3.00	.141	-2.0000	.2952
		Post	2	3.40	.283		
Art and Physical Education	Cognitive Level & Ability to Solve	Pre	4	2.00	.283	-1.7500	.3305
		Post	4	2.70	.849		

* P 〈 0.05

인문·사회 계열과 공학계열은 유의수준 .0000과 .0001로 유의수준 0.05 이하 값으로 높은 유의성을 보였으나 자연, 의학, 예·체능은 통계적으로 유의미하지 않았다. 이는 표본수가 너무 낮아서 정확한 통계가 나오지 않은 결과이다.

그러나 계열별 사전-사후의 평균 차이를 보면 Fig. 2와 같다. 예·체능, 의학, 자연, 인문·사회, 공학 계열 순으로 큰 차이를 보였다. 비전공자를 위한 수업 방법으로 연구할 필요성이 있다. 표본수의 한계가 있지만 향후 계열별로 학습이 이루어질 때 추가 연구가 필요하다.



5.2 설문에 대한 사전-사후 t-검정

컴퓨팅 사고력 향상에 대한 대응표본으로 t-검정한 결과는 Table 6에서 보여준다. 컴퓨팅 사고력 향상에 대한 문항은 인지 수준 4문항과 해결 능력 6문항으로 구성된다. 컴퓨터 사고력에 대한 전체 문항의 평균은 리커트 4점 척도로 구성하여 사전 2.67, 사후는 3.07이다. 유의수준에서도 모든 문항에서 0.05이하의 값으로 유의미한 값을 보여준다. 인지수준에서는 사전 2.54, 사후 3.04, 해결능력에서는 사전 2.77, 사후 3.10이다. 이와 같은 결과를 통해 언플러그드 기반의 소프트웨어 교육이 컴퓨팅 사고력 향상에 도움이 되었으며, 관심 또한 높아졌음을 확인 할 수 있었다.

6. 결론

본 논문은 교양 수업에서 소프트웨어 수업을 하기 위한 방법으로 언플러그드를 적용하였다. 언플러그드를 적용하기 위하여 하이브리드 학습법을 개선하여 새로운 온·오프라인 통합 모형을 만들고 컴퓨팅 사고의 핵심 요소 5개를 추출하여 수업을 진행하였다. 수업을 통하여 다

Table 6. Pre-post t-test results for the survey

Question			Survey	*N*	*M*	*SD*	*t*	*p*
Cognitive Level	1	I feel the difficulty in computational thinking.	Pre	68	2.06	.694	-4.464	.0001
			Post	68	2.53	.861		
	2	I am interested in computational thinking.	Pre	68	2.29	.579	-8.899	.0000
			Post	68	3.00	.778		
	3	I think computational thinking is necessary.	Pre	68	2.74	.567	-4.662	.0000
			Post	68	3.26	.567		
	4	I think PBL-based programming education helps improve computational thinking.	Pre	68	3.06	.649	-2.963	.0056
			Post	68	3.35	.646		
Ability to solve	5	I can logically design the process to solve the problem.	Pre	68	2.62	.739	-2.244	.0031
			Post	68	2.79	.729		
	6	I can gather the information to solve the problem.	Pre	68	2.91	.668	-4.519	.0001
			Post	68	3.29	.579		
	7	I can sort and analyze the collected data to solve the problem.	Pre	68	2.82	.626	-3.447	.0016
			Post	68	3.17	.626		
	8	I can solve a complex problem by dividing it into several problems.	Pre	68	2.62	.779	-4.737	.0000
			Post	68	3.12	.640		
	9	I can solve the problem through computational thinking.	Pre	68	2.74	.567	-3.186	.0031
			Post	68	2.97	.627		
	10	Computational thinking helps solve problems in real life.	Pre	68	2.91	.712	-3.973	.0004
			Post	68	3.24	.554		

* P 〈 0.05

양한 문제들을 해결하는 과정은 다양한 전공이 섞여있는 상태에서 문제 해결을 누구나 쉽게 접근할 수 있는 방법을 모색하였고 컴퓨팅 사고력 향상을 측정하였다.

또한 수업 전과 수업 후에 학생들의 컴퓨팅 사고력 관점의 능력 향상에 효과가 나타는지 확인하기 위해 구성된 설문을 수강생들에게 실시하여 유의미한 차이를 알아보았다.

전공 계열별 측정 결과에서는 인문·사회와 공학 계열에서 유의미한 결과를 보였고 나머지 계열은 의미가 없었다. 유의미한 결과를 보이지 않은 이유로는 표본수가 너무 낮아서 정확한 통계가 나오지 않은 결과이다.

그러나 계열별 사전-사후의 평균 차이를 보면 예·체능, 의학, 자연, 인문·사회, 공학 계열 순으로 큰 차이를 보였다. 비전공자를 위한 수업 방법으로 연구할 필요성을 보였으며, 표본수의 한계가 있지만 향후 계열별로 학습이 이루어질 때 추가 연구가 필요하다.

컴퓨팅 사고력 향상에 대한 대응표본으로 t-검정한 전체 결과는 사전 2.67, 사후는 3.07이다. 유의수준에서도 모든 문항에서 0.05이하의 값으로 유의미한 값을 보여준다. 인지수준에서는 사전 2.54, 사후 3.04, 해결능력에서는 사전 2.77, 사후 3.10이다. 이와 같은 결과를 통해 언플러그드 기반의 소프트웨어 교육이 컴퓨팅 사고력 향상에 도움이 되었으며, 관심 또한 높아졌음을 확인 할 수 있었다.

또한 관찰자 조사에서 컴퓨팅 사고의 요소를 조사한 결과 모든 계열에서 추상화 개념을 가장 어려워했다. 인문·사회와 자연, 의학 계열에서는 패턴인식을, 예·체능 계열에서는 알고리즘 개념을 익히는데 가장 어려워했다. 이를 비전공자를 위한 교육과정을 개발할 때 반영할 필요가 있다. 컴퓨팅 사고력 관련 교육이 점점 확대되는 시점에서 교육과정을 통하여 관련 역량을 향상시킬 수 있는지 즉 교육과정이 효과가 있는 지에 대한 평가는 매우 중요하다. 그러나 컴퓨팅 사고력 구성요소에 대한 측정은 할 수 있다고 하더라도 컴퓨팅 사고력 역량 그 자체를 측정하는 것은 쉽지 않다. 다만 언플러그드 교육과정을 적용한 것은 컴퓨터 없이 누구나 컴퓨터 교육을 할 수 있는 방법을 제시하였다는 점과 미미하지만 컴퓨팅 사고의 각 구성요소별 효과를 알 수 있었다는 점에서 차후 컴퓨팅 사고력의 역량을 측정하는 연구에 도움이 될 것이다. 향후 다양한 언플러그드 콘텐츠를 개발하여 컴퓨팅 사고 요소별 사전-사후 효과를 검정할 것이다.

REFERENCES

[1] IT TREND, https://blog.his21.co.kr/302.

[2] Resnick, M. (2002). Rethinking Learning in the Digital Age. In The Global Information Technology Report: Readiness for the Networked World, edited by G. Kirkman. Oxford University Press.

[3] EduWang, https://eduwang2013.blog.me/221774114289

[4] Education Department. (2018). 2015 Revised National Curriculum.

[5] J. H. Chun & J. Y. Yoo. (2014). Trials and effects of a learner-centered creative training technique on undergraduate education of medical record information management. *Journal of Digital Convergence, 12(3),* 277-288.

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

[7] M. S. Lee. (2017). A Study on Creative and Convergent SW Education Programs for improving Computational Thinking. *Journal of The Korea Society of Computer and Information*, *22(7)*, 93-100.

[8] Y. J. Lee. etc. (2014). Research for Introducing Computational Thinking into Primary and Secondary Education. KOFAC Report.

[9] S. J. Lee. (2018). A Study on Designing a Class of Convergence Thinking based on Computational Thinking. *The Journal of Korean Society of Science & Art, 36*, 255-263.
DOI : 10.17548/ksaf.2018.12.30. 255.

[10] J. H. Ku. (2017). Designing an App Inventor Curriculum for Computational Thinking based Non-majors Software Education. *Journal of Convergence for Information Technology*, *7(1)*, 61-66.
DOI: 10.22156/CS4SMB.2017.7.1.061

[11] H. S. Han, S. K. Han. (2009). A Case Study on Information Education for Pre-Service Teacher using Unplugged Computing, *Journal of The Korean Association of information Education, 13(1)*, 23-30.

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

[13] J. M. Wing. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A, 366*, 3717-3725.

[14] K. S. Oh. (2016). A study on the contents of computational thinking for programming education.Ph.D. dissertation. Sungkyunkwan University, Seoul.

[15] J. S. Kim. (2015). SW Education Teaching and Learning Model Development. KERIS Report, CR2015-35.

[16] Software Education Blog, https://m.blog.naver.com/PostView.nhn?blogId=gi_sik_in&logNo=220238605443&proxyReferer=https%3A%2F%2F.

[17] M. S. Lee. (2016), *Hybrid teaching method using MOOC in liberal arts education. Korea National Institute for General Education Report,* RR-2016-14- 633.

[18] Y. S. Han. (2018), Effectiveness of problem-based learning based programming education : Focus on Computational Thinking, *Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities and Sociology, 8(7)*, 433-445,
DOI: 10.21742/AJMA HS.2018.07.74

이 명 숙(Myung-Suk Lee) [정회원]

· 2001년 2월 : 계명대학교 컴퓨터공학과(공학사)
· 2003년 2월 : 계명대학교 컴퓨터공학과(공학석사)
· 2009년 8월 : 계명대학교 컴퓨터공학과(공학박사)
· 2013년 3월 ~ 현재 : 계명대학교 타불라라사칼리지 교수
· 관심분야 : 컴퓨터 네트워크, 컴퓨터 구조, 컴퓨터 교육, 고등교육, OER, 학습 분석, 인공지능, 블록체인
· E-Mail : mslee@kmu.ac.kr