대학생들의 컴퓨팅 사고력 향상을 위한 UDDPAAP 역량 교수·학습 모델 설계

전미연 · 김의정 · 강신천 · 김창석 · 정종인

공주대학교 컴퓨터교육과

A Design of an UDDPAAP Competence Teaching-Learning Model to Improve Computational Thinking in College Students

Mi-Yeon Jeon · Eui-Jeong Kim · Shin-Cheon Kang · Chang-Suk Kim · Jong-In Chung

Dept of Computer Education, Kongju National University

E-mail : jmy0826@kongju.ac.kr, ejkim@kongju.ac.kr

요 약

본 연구는 대학생들의 SW 교육 핵심역량 중 컴퓨팅 사고력을 신장시키기 위해 역량 교수·학습 모델을 설계하였다. 사전에 소프트웨어 코딩 경험이 없는 학습자의 역량을 분석하고 기존의 SW 중심 교수·학습 5가지 모델 중 시연 중심 모델(DMM)과 개발 중심 모델(DDD) 그리고 CT 요소 중심 모델(DPAA) 등을 재구성하고 실생활 문제를 해결 및 컴퓨팅 사고력을 키우기 위한 Unplugged 활동과 Bebras Challenge 컴퓨팅 사고력 평가 도구 등을 면밀히 분석하여 역량 교수·학습 모델인 UDDPAAP (Unplugged-Demonstration-Decomposition-Pattern Recognition-Abstraction-Algorithm-Progrmming)을 설계하였다. Unplugged 활동 중 일부분을 대학생들 수업에 적합하게 변형하고, Bebras Challenge 컴퓨팅 사고력 평가 도구에서 제시하는 문제를 선별한 후 기존의 교수·학습 모델에 적용하였다. 연구의 효과를 검증하기 위해 코딩 경험이 없는 대학교 1학년 학생들에게 SW 교육 및 컴퓨터 정보 소양 교육 경험에 따른 컴퓨팅 사고력과 자신감 등의 사전 검사를 하고 UDDPAAP 교수·학습 모델을 적용하여 수업을 진행한 후 사후 검사를 하였다. 연구 결과 UDDPAAP 교수·학습 모델을 통해 SW 교육을 경험한 학생들의 컴퓨팅 사고력 관련 역량이 향상됨을 알 수 있었다.

ABSTRACT

The purpose of this study was to design a competence teaching-learning model that could help college students improve their computational thinking among core competences in SW education. A competence teaching-learning model, UDDPAAP (Unplugged-Demonstration-Decomposition-Pattern Recognition-Abstraction-Algorithm-Programming), was designed by analyzing competences of learners with no experience in software coding, by reconstructing DMM, DDD, and DPAA among the five existing SW-based teaching-learning models, and by analyzing unplugged activity and the Bebras challenge computational thinking scale carefully. The unplugged activity partially adapted to instruction for college students and some items chosen from the Bebras challenge computational thinking scale were applied to the existing teaching-learning model. To determine the effects of the study, pretest was conducted in freshmen for computational thinking and self-confidence on the basis of the experience in SW and computer information literacy education, and posttest following instruction applying the UDDPAAP teaching-learning model. The students provided with SW education based on the UDDPAAP teaching-learning model saw their computational thinking competence improved.

키워드

컴퓨팅 사고력, 수업 모델, 교수 학습 방법, 역량, 학습 모델

I. 서 론

OECD의 DeSeCo 프로젝트는 청소년과 성인이 21세기에 필요한 핵심역량과 능력 요인을 분석하고 제안하였다. 또한 지속적인 연구로 2030년 학생들이 미래 사회에서 요구되는 역량을 기

르는 방법에 대한 교육 틀을 마련하기 위한 Education 2030 프로젝트가 진행 중이다. Education 2030 개념 틀은 역량(competencies)과 행동(action)을 포함하고 있고 역량은 특정 상황에서 지식, 스킬, 태도 및 가치를 활용하는 능력을 말하고, 행동은 이러한 역량을 발휘하는 것을 말한다.[1]

2018년까지 미래 사회에서 요구되는 역량 프레임워크 개발을 목표로 1단계 연구와 2019년 이후의 21세기 역량을 기르기 위한 교수-평가 방법 등에 대한 2단계 연구로 구분해서 진행되고 있다. 이러한 역량 중심 교육은 단순 지식에 대한 교육이 아닌 새로운 문제에 적용할 수 있는 힘을 기르는 교육으로 한국도 핵심역량에 관한 지속적인 연구를 통해 2015년 개정 교육 과정을 역량 중심으로 개정하여 창의 융합형 인재 양성을 목표로 한다.[2] 2015 정보과 교육 과정에 제시된 핵심역량은 컴퓨팅 사고력, 정보문화 소양, 협력적 문제해결력이다. 이러한 필요성에 따라 대학에서도 21세기에 필요한 핵심역량인 컴퓨팅 사고력를 강조하고 학생들의 역량을 키우도록 교수·학습 설계를 하고 있다.[3]

본연구의 목적은 대학생들의 소프트웨어 교육 및 컴퓨터 정보 소양 교육 경험에 따른 컴퓨팅 사고력 역량을 분석한 후 여러 가지 다양한 문제를 해결을 위해 필요한 핵심역량인 컴퓨팅 사고력이 향상될 수 있도록 교수·학습 모델을 설계하여 효과성을 확인하고자 한다.

II. 이론적 배경

Jeannette M. Wing은 컴퓨팅 사고는 컴퓨터과학자뿐만 아니라 누구나 배워서 활용할 수 있는 보편적인 사고이자 기술로써 읽기, 쓰기, 셈하기를 배우듯이 아이들이 기본적으로 갖춰야 할 역량으로 정의하고 있다. 컴퓨터과학은 다양한 분야에서 융합되어 활용되고 있다. 생물학 분야에서 데이터 구조와 알고리즘과 같은 추상화를 통해 단백질을 구조화하고 기업의 통계 관련 부서에서는 컴퓨터 공학자를 채용하고 화학에서는 나노 컴퓨팅, 물리학에서는 양자 컴퓨팅이 사용된다. 이와 같은 사회의 변화에 대학 신입생들도 '컴퓨터 과학자처럼 생각하기' 과목을 필수로 이수해야 하며 예비 대학생들도 컴퓨팅 방법론과 모델을 가르쳐야 한다고 주장하고 있다.[4] 이에 발맞춰 교육 현장에서도 컴퓨팅 사고력에 관한 연구가 활발히 진행되고 있다.[5][6][7]

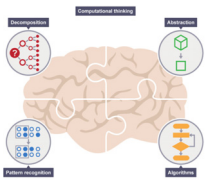
교육부에서 제시하고 있는 CT 구성요소는 다음과 같다.

Table 1. Computational Thinking[8][9]

SW 역량	컴퓨팅 사고력		
	문제해결요소		내용
분석 능력	자료수집		문제해결에 필요한 내용이나 사항을 모으는 것
	자료분석		자료의 이해 및 패턴 찾기 및 결론 도출하기
	자료표현(구조화)		문제를 그래프, 차트, 그림으로 시각화하기
모델링 능력	문제분해		큰 문제를 해결 가능한 작은 문제로 나누기
	추상화		문제해결에 반드시 필요한 요소만 남기고 단순화시키는 것
	패턴인식		문제해결과정에 반복적으로 나타나는 패턴을 인식하는 것
	알고리즘		문제를 해결하기 위한 일련의 과정
구현 능력	자동화	코딩	프로그래밍 언어를 이용하여 주어진 문제해결
		시뮬레이션	프로그램 실행
일반화 능력	적용과 일반화		문제해결과정을 다른 문제에 적용하기

컴퓨팅 사고력 구성요소가 너무 세분화 되었을 경우 교수·학습이나 교과목 목표를 달성하는 것이 어려울 수 있고 여러 학문이 융복합될 때 세분화시키기 어렵다고 한다.[9] 따라서 Keris에서는 컴퓨팅 사고력 요소를 다음과 같이 재정의하였다.

Table 2. Redefining Computing Thinking[9].

KS3 용어		Google 용어
Decomposition		Decomposition
Pattern recognition		Pattern recognition
Abstraction		Abstraction
Algorithms		Algorithms
Programming		

컴퓨팅 사고력의 필요성에 따라 교수·학습 모델인 DMM((Demonstration-Modeling-Making) 모델, UMC(Using-Modify-reCreate) 모델, DDD (Discovery-Design-Development) 모델, NDIS(Need-Design-Implementation-Share)모델, DPAA (Decomposition-Pattern Recognition-Abstraction-Algorithm)등이 개발되었다[9]. 그리고 컴퓨팅 사고력을 평가할 수 있는 도구들도 계속 연구중이다.[10] 본 연구에서

는 평가 도구의 하나로 Dr.Scratch를 이용해 컴퓨팅 사고력을 평가하였다. 국내외 연구에서 컴퓨터 비전공 대학생들을 대상으로 스크래치 프로젝트를 분석한 결과 Dr.Scratch의 항목을 이용해 컴퓨팅 사고력 역량 평가에 활용될 수 있음을 보고 되었다.[11][12]

III. 연구 방법 및 결과

코딩 경험이 없는 대학생들의 컴퓨팅 사고력 향상을 위한 역량 교수·학습 모델 UDDPAAP를 설계하고 효과성을 입증하기 위해 다음과 같은 연구 절차를 진행하였다.

첫째, 국내외 선행 연구를 통해 21세기에 필요한 역량 교수·학습 모델과 컴퓨팅 사고력 평가 도구 등을 분석하였다.

둘째, 선행 연구 분석을 통해 교수·학습 모델 UDDPAAP(Unplugged-Demonstration-Decomposition-Pattern Recognition-Abstraction-Algorithm-Progrmming)을 설계하였다.

셋째, 코딩 경험이 없는 대학교 1학년 학생들의 소프트웨어 교육 및 컴퓨터 정보 소양 교육 경험에 따른 컴퓨팅 사고력과 자신감 등을 사전-사후 검사를 시행하였다.

넷째, UDDPAAP 역량 교수·학습 모델을 적용하여 수업을 진행한 후 프로젝트 결과를 컴퓨팅 사고력 평가 도구 Dr.Scratch로 분석하였다.

3.1 UDDPAAP 역량 교수·학습 모델 설계

코딩 경험이 없는 대학교 1학년 학생들의 역량을 분석하고 기존의 교수·학습 모델을 재구성하였다. 컴퓨터과학 원리와 개념을 이해시키기 위한 Unplugged 활동과 Bebras Challenge 컴퓨팅 사고력 평가 도구 등을 분석하여 역량 교수·학습 모델인 UDDPAAP을 설계하였다. Unplugged 활동 중 일부분을 대학생들 수업에 적합하게 변형하고, Bebras Challenge 컴퓨팅 사고력 평가 도구에서 제시하는 문제를 선별한 후 기존의 교수·학습 모델에 적용하였다.

3.1.1 UDDPAAP 모듈 구성

SW 교수·학습 모델을 참조하여 대학생들을 위한 지식 모듈과 활동 모듈로 나누어 구성하였다.

지식 모듈Knowledge Module)은 문제 핵심을 파악하고 필요한 컴퓨터 과학의 개념과 원리를 인식하는 단계로 실생활 예제와 사례를 학습하는 모듈이다.

활동 모듈(Action Module)은 지식 모듈에서 제시한 문제를 세부 모듈 4개로 나누어 필요에 따라 분해해서 사용할 수 있도록 설계하였다.

3.1.2 수업 설계

수업은 15주로 구성하여 주당 3시간씩 SW코딩 수업을 진행하였다. 먼저 매주 Unplugged 개념을 적용한 Bebraschallenge 문제 한 문제를 선별하여 Quiz 형식으로 제시하였다. 1주~3주차까지 Scratch를 이용해 컴퓨터 과학의 원리와 기본 사용법을 익힌 후 4~6주차는 컴퓨팅 사고력 요소인 분해, 패턴인식, 추상화, 알고리즘, 프로그래밍(자동화)을 적용할 수 있는 주제를 선정하여 모듈1부터 모듈4를 제시하였다. 단, 수업 환경과 학생의 수준을 고려하여 적절하게 주제를 선정하여야 한다. 7주~10주차는 매주 한 개씩 큰 프로젝트인 '미로 탈출하기' '그림판 만들기', '청소로봇 만들기'와 같은 실생활 관련 문제 프로젝트를 작성하였다.

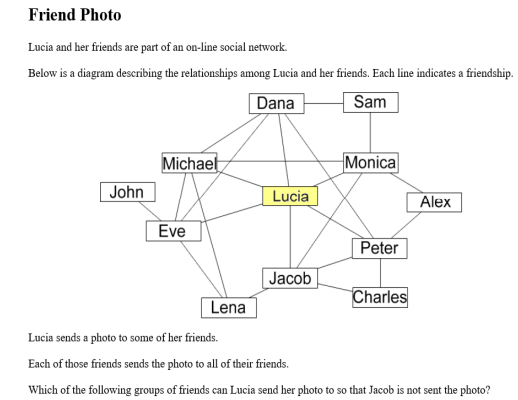


Fig 1. Bebraschallenge Sample Questions[11].

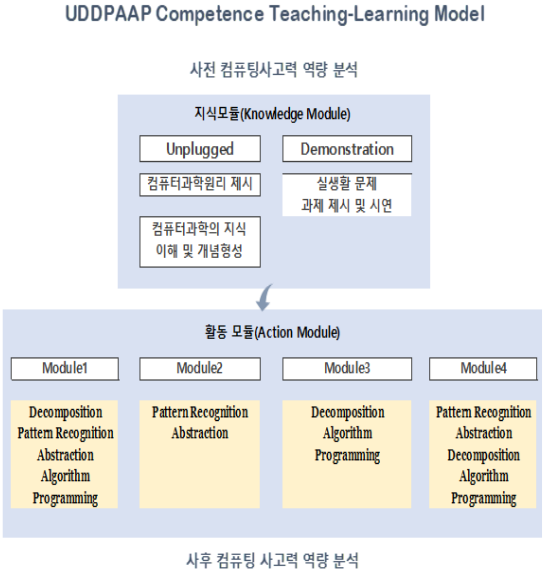


Fig 2. UDDPAAP 역량 교수·학습 모델.

3.2 컴퓨팅 사고력 역량 사전·사후 검사

대학생들의 컴퓨팅 사고력 역량을 파악하기 위해 실증분석 방법을 사용하였다. 본 연구의 실증분석은 모두 유의수준 p<.05에서 검증하였으며, 통계처리는 SPSSWIN 21.0 프로그램을 사용하여 분석하였다. 범주형 자료에서 비모수 기법일 경우 교차분석(chisqure test) χ^2(p)을 실시하여 집단 간의 차이를 살펴보았다.

3.2.1 코딩교육 경험 및 일반적 사항

조사대상자의 일반적인 특성을 알아보기 위해 빈도분석(Frequency Analysis)을 했다. 남자 85.7%, 여자 14.3%이고 코딩교육 경험의 경우 80%가 없는 것으로 나타났다. 도시 규모의 경우 대도시 68.6%, 중소도시 25.7%, 읍면지역 5.7%로 나타났다.

Table 3. 코딩 교육 경험 및 일반적인 사항.

		빈도	%
성별	남자	30	(85.7)
	여자	5	(14.3)
교육경험	없음	28	(80.0)
	1년미만	5	(14.3)
	1년이상 2년 미만	2	(5.7)
도시규모	읍면지역	2	(5.7)
	중소도시	9	(25.7)
	대도시	24	(68.6)
	합계	35	(100.0)

3.2.2 신뢰도 분석

컴퓨팅 사고력 자신감 역량에 대한 신뢰도 검사를 수행하여 문항 간의 신뢰도를 측정하여 예측 가능성, 정확성 등을 살펴보았다. 수집된 설문지의 각 항목에 대한 안정성, 일관성 및 예측 가능성을 알아보기 위하여 본 연구에서는 크론바하알파(Cronbach's α) 계수를 신뢰도 계수로 사용하였다.

Table 4. 신뢰도 분석.

	Cronbach의 알파
문제핵심파악	.940
자료수집	.925
자료분석	.927
자료표현	-
문제분해	-
추상화	-
패턴인식	-
알고리즘과절차	.928
자동화	.875
시뮬레이션	-
컴퓨팅 사고력 자신감 역량(전체)	.984

사회과학에서 일반적으로 0.6 이상을 측정지표의 신뢰성에 커다란 문제가 없다고 인정하므로, 본 연구에서도 0.6 이상을 기준으로 신뢰성을 평가하기로 하는 것으로 한다. 본 연구에서는 이와 같은 크론바하알파 계수를 이용하여 내적 일관성에 의한 측정 도구의 신뢰도를 검증하였으며, 컴퓨팅 사고력 자신감 역량에서 2문항 이상일 경우 신뢰도를 측정하였으며, .8 이상으로 신뢰도가 높게 나타났다. 전체적으로 컴퓨팅 사고력 자신감 역량의 경우 .984로 신뢰수준을 만족한다고 할 수 있다.

3.2.3 문제해결요소에 따른 컴퓨팅 사고력 자신감 역량

일반적인 특성에 따라 컴퓨팅 사고력 자신감 역량에 대한 평균의 차이를 살펴보기 위하여 평균 차이 검증인 T-test를 실시하였다. 컴퓨팅 사고력 자신감 역량 분석결과 전체적으로 사전보다 사후에 더 높게 나타났으며, 유의한 차이를 보였다(p<.01). 특히 문제 핵심을 파악하는 역량과 자료수집, 자료표현, 추상화 순으로 역량 상승이 두드러지며 이러한 결과는 코딩 경험이 없는 학생들이 설계된 수업을 통해 컴퓨팅 사고력 자신감 역량의 향상에 도움이 되었다는 것을 알 수 있다.

Table 5. 컴퓨팅 사고력 자신감 역량.

	시기	N	평균	표준편차	t	p
문제핵심파악	사전	35	2.20	.78	3.387***	.001
	사후	35	2.81	.74		
자료수집	사전	35	2.37	.89	2.084*	.041
	사후	35	2.77	.70		
자료분석	사전	35	2.21	.80	3.068**	.003
	사후	35	2.79	.76		
자료표현	사전	35	2.20	.93	2.951**	.004
	사후	35	2.80	.76		
문제분해	사전	35	2.31	.93	2.174*	.033
	사후	35	2.74	.70		
추상화	사전	35	2.26	.85	2.544*	.013
	사후	35	2.74	.74		
패턴인식	사전	35	2.26	.85	2.223*	.030
	사후	35	2.69	.76		
알고리즘과절차	사전	35	2.23	.85	2.038*	.045
	사후	35	2.63	.79		
자동화	사전	35	2.21	.87	2.231*	.029
	사후	35	2.64	.73		
시뮬레이션	사전	35	2.29	.86	2.489*	.015
	사후	35	2.77	.77		
전체	사전	35	2.25	.79	2.738**	.008
	사후	35	2.74	.68		

p<.05, **p<.01, ***p<.001

3.3 Dr.Scratch 프로젝트 평가

UDDPAAP 교수·학습 모델을 적용하여 수업을 진행한 후 Dr.Scratch를 이용해 프로젝트를

평가한 결과는 다음과 같다. 제시한 프로젝트는 동일한 점수를 적용하여 평가하였다. 21점 만점에 평균 16.89, 표준편차 1.409, 최대점수 20점, 최소점수 11점으로 결과를 보였다.

Table 7. Dr.Scratch Project Scores (n = 35).

Dr.Scratch	Min	Max	M	SD
Score	11	20	16.89	1.409

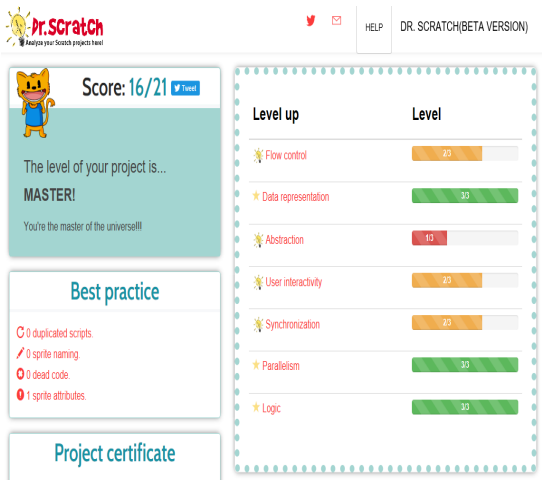


fig 4. Dr.Scratch Project Evaluation.

IV. 결 론

본 연구는 코딩 경험이 없는 대학생들의 컴퓨팅 사고력을 신장시키기 위한 UDDPAAP 역량 교수·학습 모델을 설계하고 효과를 분석하였다.

UDDPAAP 모델은 학습자의 능력을 분석한 후 교수·학습 전략을 세울 수 있다는 장점이 있다. 또한, 컴퓨터 과학의 원리 및 개념을 실생활 사례 및 예제를 통해 이해하므로 학생들의 이해도가 전반적으로 높았다. 단지 코딩을 처음 하는 학생들은 프로그램 도구에 익숙하지 않아 기능을 익히는데 시간이 조금 걸렸다.

컴퓨팅 사고력 자신감역량은 UDDPAAP 교수·학습 모델을 적용하여 코딩 수업을 진행한 후에 더 높게 나타났다. 문제 핵심을 파악하는 역량이 가장 높게 나타났으며 자료수집, 자료표현, 추상화 순으로 컴퓨팅 사고력 역량이 상승한 것을 볼 수 있었다. 이러한 결과는 매주 Unplugged Bebraschallenge 문제를 한 문제를 선별하여 Quiz 형식으로 풀어 본 것과 코딩 프로젝트를 매주 한 개씩 진행함으로써 컴퓨팅 사고력이 신장 된 것으로 보인다. 전체적인 컴퓨팅 사고력이 향상됨에도 불구하고 분해, 추상화, 패턴인식, 알고리즘과 절차, 프로그래밍의 문제해결요소에서는 코딩 경험이 없는 대학생들은 조금 어려움을 느꼈다. 학생들의 학습 수준에 맞춰 난이도를 조절해 문제 설계를 할 필요성이 보인다.

연구 결과 UDDPAAP 역량 교수·학습 모델은 코딩 경험이 없는 대학생들의 컴퓨팅 사고력 향상에 영향을 미치는 것으로 나타났다.

실제 수업을 하는 교수자들은 정형화된 교수·학습 모델보다는 수업 아이디어와 교수·학습 전략을 통해 실제 적용 가능한 교육 방법을 선호한다. UDDPAAP 교수·학습 모델이 정형화된 모델이 아닌 교수 전략으로 활용되고 전공을 하지 않은 코딩 경험이 없는 학생들의 컴퓨팅 사고력 역량 향상에 도움을 주길 기대한다.

참고문헌

[1] 한국교육과정평가원, 21세기 역량 기반 교육과정 개발 방향연구 OECD Education 2030

[2] 한국교육개발원·한국교육학술정보원(2015), SW교육 교수·학습 모형 개발 연구

[3] Young-Su Son·Kwang-Jae Lee, Computational Thinking Teaching Model Design for Activating IT Convergence Education, JKIECS, vol. 11, no. 5, 511-522, 2016

[4] Jeannette M. Wing, COMMUNICATIONS OF THE ACM March 2006/Vol. 49, No. 3

[5] 박성빈, 안성진, 컴퓨팅 사고력의 역량 탐색 연구, 한국컴퓨터교육학회 논문지, 제19권 제5호(2016.9)

[6] Soojin Jun, Journal of The Korean Association of Information Education Vol.21, No.6, December 2017, pp.619-627

[7] Yong-Ju Jeon·Tae-Young Kim(2016), 한국컴퓨터정보학회, 제53차 동계학술대회논문집, 24권 1호

[8] Jennette M. Wing(2011), Computational Thinking. CSTA Workshop. 4 March

[9] 한국교육개발원, 한국교육학술정보원, SW 교육 교수학습 모형 개발 연구 수탁연구 CR 2015-35

[10] 정웅열·이영준, 한국컴퓨터교육학회 논문지 제20권 제5호(2017.9)

[11] Soohwan Kim (2015), Analysis of Scratch Assessment about Computational Thinking Capability. The Korean Association of Computer Education Research Journal, 18(5), 25-34.

[12] Jesus Moreno-Leon·Gregorio Robles (2015), Dr.Scratch : a Web Tool to Automatically Evaluate Scratch Projects. WIPSCE '15 November 09-11, 2015, Londeon, United Kingdom.