Journal of The Korean Association
of Information Education Vol. 19, No. 1,
March 2015, pp. 21-30

http://dx.doi.org/10.14352/jkaie.2015.19.1.21

컴퓨터 프로그램 교육에서 자기조절 학습 모델 개발

김갑수
서울교육대학교 컴퓨터교육과

요 약

21세기 지식 정보 사회에 컴퓨터 교육이 매우 중요하다. 컴퓨터 교육에서 컴퓨터 프로그래밍 교육이 매우 중요하다. 컴퓨터 프로그래밍 교육에는 교수 학습 모델이 거의 없다. 본 연구에서는 학생들이 자기조절 학습을 할 수 있는 자기 조절 학습 모형을 개발한다. 본 연구에서는 자기 조절 학습 요소, 자기 조절 학습 단계와 자기 조절 학습 모형을 제안한다. 자기조절 학습 요소는 과제 수준, 일반화, 효율화이다. 자기조절 학습 단계는 문제이해, 설계, 코딩, 시험, 유지보수이다. 자기조절 학습 모델은 복사하기, 변형하기, 창조하기, 도전하기이다. 본 연구의 결과는 다음과 같다. 학습 요소들과 성취도간의 상관관계 분석은 효율화와 일반화가 과제 수준보다 더 높았다. 학습 단계에는 문제 이해와 설계 단계가 다른 단계보다 더 높았다. 학습 모형에서는 변형하기, 창조하기, 도전하기가 구현하기보다 상관관계가 더 높았다.

키워드 : 자기 조절 능력, 초등학생, 프로그래밍 교육, 컴퓨터 언어

A Self-regulated Learning Model Development in Computer Programming Education

Kapsu Kim
Dept. of Computer Education, Seoul National University of Education

ABSTRACT

Information and knowledge society in the 21st century computer education is very important. Computer programming education in computer education is very important. There are very few teaching and learning model of computer programming education. In this paper, we develop a self-regulated learning model for students to be self-regulated learning. In this study, we propose self-regulated learning elements, a self-regulated learning steps and self-regulated learning modele. Self-regulated learning elements are task level, generalized level, and efficiency level. Self-regulated learning phases are problem understanding, design, and coding, testing, and maintenance. Self-regulated learning models are to copy, to modify, create, and to challenge. The results of this study are as follows. At Correlations between learning elements and achievement, generalized level, and efficiency level are higher than the task level. At Correlations between learning and achievement, Understanding and design stages are higher than the other stages. At Correlations between learning model and achievement, to transform, to create, and to challenge are higher than to copy.

Keywords : Self-regulated, Elementary School, Programming Education, Computer Language

이 논문은 2014년도 서울교육대학교 교내연구비에 의하여 연구되었음.
논문투고 : 2014-12-28
논문심사 : 2015-01-13
심사완료 : 2015-03-10

1. 서론

21세기 지식 정보 사회에서 초등학생들부터 지식 정보를 스스로 구성할 수 있는 능력이 매우 중요하다. 따라서 초등학교에서 컴퓨터 프로그램 교육이 매우 중요함이 부각되어 세계의 각 국가들은 초등학교부터 컴퓨터 프로그램 교육에 많은 관심을 가지고 있다. 특히, 미국과 영국 등은 컴퓨터 프로그램 교육의 중요성을 인식하고 초등학교 때부터 기존의 컴퓨터 활용교육에서 컴퓨터 프로그램 교육을 실시하는 것으로 전환하고 있다. 물론 북유럽의 작은 국가들도 컴퓨터 프로그램 교육의 중요성을 인식하여 초등학교 단계부터 컴퓨터 프로그램 교육을 실시하고 있다.

이제 막 컴퓨터 프로그램 교육은 실시 초기 단계이기 때문에 컴퓨터 프로그램 교육의 다양한 교육학적 접근이 부족하다. 김갑수[9]는 Miltiadou와 Yu[11]가 개발한 것을 수정하여 개발한 컴퓨터 프로그램 교육에서 자기 효능감 측도에서 학생들이 컴퓨터 프로그램 교육을 많이 받을수록 자기 효능감이 높게 나타났다. 이처럼 컴퓨터 프로그램 교육의 타당성을 입증하기 위해서 컴퓨터 프로그램 교육에서 다양한 교육학적 접근이 필요하다. 또한, 컴퓨터 프로그램 교육을 하기 위하여 교육 모델이 많이 필요하다. 다른 교과 교육들은 교육학의 기본 모델을 기반으로 교과 교육 모델들을 많이 만들어서 현장에 적용하고 있지만 컴퓨터 교육은 현장 적용 정도가 미미하다.

컴퓨터 교육 프로그램은 학생들이 논리적인 전개과정이 있어야 하고 창의적인 아이디어를 구현할 수 있는 능력을 기르는 것이 필요하다. 컴퓨터 프로그램 교육은 논리적인 교육과 창의적인 교육을 동시에 할 수 있는 좋은 교과목이다. 일반적으로 창의성을 중시하면 논리성이 부족할 수 있고, 논리성을 강조하면 창의성이 부족할 수 있다. 그러나 컴퓨터 프로그램 교육은 창의적인 아이디어를 발현하여 이를 논리적으로 표현하여 구현할 수 있는 도구 교육과목이다. 따라서 초등학교에서 컴퓨터 프로그램 교육이 매우 필요하다.

따라서 본 연구에서는 초등학생들이 컴퓨터 프로그램 교육을 위해서 자기조절 학습 모델을 만들어서 초등학생들에게 효과적인 프로그램 교육 방법을 제안한다.

컴퓨터 프로그래밍 교육은 학생들과 교사들이 상호작용하면서 프로그램 도구들을 이용하기 때문에 학생들의 학습과정에서 교사 학생의 일대일관계에서 교사, 학생, 프로그래밍 도구의 3개 객체간의 관계가 형성되기 때문에 학생들이 학습을 할 때에 자기 조절 능력이 매우 중요하다. 따라서 본 연구에서는 학생들이 컴퓨터 프로그램 언어 교육을 하면서 학생들이 자기조절 능력을 기르게 하는 모델을 제안하는 것이 매우 중요하다.

자기 조절 능력(SRI-Self-Regulated Learning)은 스스로 학습을 관리하는 능력으로서 학습 계획, 목표 설정, 전략 실행 및 요약 점검 과정이다[17][2][14]. Zimmerman와 Martinez-Pons[6]는 자기조절학습과 학업성취도는 상관관계가 있다는 것을 입증하였고, 자기조절 학습 능력이 높은 학생들이 그렇지 않는 학생들보다 과제 수행 결과가 좋다는 것을 검증하였기 때문에 자기조절 학습이 매우 중요하다. 따라서 학생과 교사의 일대일 수업보다 컴퓨터 프로그램 도구를 이용하기 때문에 자기조절 학습 능력이 컴퓨터 프로그램 교육에서 매우 중요하다.

자기 조절 학습 능력은 컴퓨터 교육과 관련 있는 분야에 적용한 것은 디지털 게임에 학생들의 자기 조절 학습 능력을 적용한 사례는 많았지만[5][6][7] 컴퓨터 프로그램 교육에 대한 자기 조절 학습 능력 모델이 없다. 따라서 최종 초등학생들에게 컴퓨터 프로그램 교육을 해야 하는 상황에서 자기조절학습 모형을 만드는 것이 필요하다.

기존의 프로그램 교육 방법론은 일반적으로 소프트웨어 개발 방법론을 적용한 프로그램 방법론을 이용하고 있고, 소프트웨어 개발 방법론에 요구사항 분석, 설계, 코딩, 시험 및 유지보수 단계를 거쳐서 소프트웨어 개발을 하는 방법에 적합한 체크 포인터가 자기조절 능력의 항목이 될 수 있다. 그렇지만 초등학생들은 즉각적인 결과를 보는 관점이 중요하기 때문에 이 방법론을 바로 적용할 수 없다. 따라서 본 연구에서 초등학생들이 컴퓨터 프로그래밍을 하면서 자기 조절 능력을 기를 수 있는 모델이 더욱 필요하다.

지금까지의 자기조절 학습에 대한 국내 연구들은 다음과 같다. 구영수와 양연숙[19]은 초등 영재아와 일반아의 학습 양식, 자기조절학습 전략 및 학습 모입에 대한 연구에서 자기조절 학습 전략들이 학생들의 몰입에 많은 영향을 미친다는 것을 연구하였다. 물론 영재아들이 일반 아동보다 자기조절 능력이 있다는 것을 알 수

있고, 이는 컴퓨터 프로그램을 교육을 할 때에 자기조절학습 모형을 만들어서 적용하면 컴퓨터 프로그램 학습이 좋다는 것을 간접적으로 시사한다는 것을 알 수 있었다.

허재은과 김홍찬[8]은 자기조절학습능력이 높은 학생들은 자기조절학습능력이 낮은 학생들에 비해 학생 본인의 학습 과정과 보조학습이 학생 자신에게 미치는 영향에 대해 구체적으로 판단하고 합리적 근거로 보조학습을 선택, 유지, 변경하는 경향이 높았고, 또한 수업 참여에 적극성이 높았고, 내재적인 동기에 의한 적극적인 행동이 많았고 또한 흥미와 자신감이 향상되었다. 이에 본 연구에서 컴퓨터 프로그래밍 교육을 하면서 자기조절 능력이 있는 학생들은 프로그래밍의 구성요소들을 잘 선택하여 주어진 문제를 해결할 수 있고 프로그래밍에 대한 흥미와 관심을 높일 수 있다.

본 연구의 제2장에서는 본 연구와 관련 있는 연구를 자세히 설명하고, 제3장에서는 본 연구에서 제안한 컴퓨터 프로그램 언어 학습을 위한 자기조절 학습 모델에 대해서 설명하고, 제4장에서 본 연구에서 제안한 컴퓨터 프로그래밍 언어 교육을 위한 자기조절학습모형을 성취도와 검증하고, 제5장은 본 연구의 결론이다.

2. 관련 연구

2.1 자기 조절 학습

자기조절 학습(SRL)은 신념, 동기, 전략, 반성의 과정 즉 학습자 스스로 자신의 학습과정을 주도적으로 지시하는 것으로 일반적으로 설명할 수 있다[3][15]. 이 자기조절학습은 많은 인지 전략에 대한 지식, 환경 조성 전략, 적절한 자원 탐색을 필요로 한다. 자기 조절 학습 모형은 학습 동기와 밀접한 연관 관계가 있다[13].

초인지(메타인지)는 문제해결력을 위한 전략에 대한 지식과 전략의 사용과 인지에 대한 지식의 구성 체계를 정의하는 것으로 인지 조절을 돕게 하고, 인지에 대한 지식은 개개인의 선언적, 과정적, 조건적 지식이고, 인지조절은 인지, 기억에 대한 통제나 학습 또는 문제해결의 과정에서 조절행동(계획, 점검, 평가)에 대한 학습이나 구체화이다. 자기 조절 학습이 잘 이루어지기 위해서는 동기 요인이 중요하다[17][18]. 근본적인 자기 자신에 대한 신념과 성취도가 높은 학생들은 초인지적 지식을 더 많이 갖고 있으며, 자기 조절 행동을 더 많이 보인다. 그러므로 자기조절학습은 전략의 다각적인 연속, 메타인지, 동기적 변수로 나타난다.

가장 일반적인 자기조절 학습 모델은 다중 단계 자기조절 학습 모델이다. 이 모델은 자기 조절 학습 모델은 가장 높은 단계인 성향 수준이고, 다음은 영역 수준이고, 가장 작은 단위가 과제 수준이로 다음 세 가지 측면에서 구성된다.

먼저 과제 수준을 살펴본다. 과제 수준은 선행의 측면, 규칙의 측면, 자기 반성적 측면에서 고려되어야 한다. 선행의 측면(anticipatory)에서는 과제 정의, 이용 가능한 자원과 방해 요소에 대한 평가, 과제 목표 결정, 과제에 접근하는 신념(지식, 목표지향, 흥미, 과제에 대한 가치판단, 자기효능감 등) 등이다.

규칙의 측면에서는 선행측면의 여러 가지 변수들의 상호작용, 메타인지과정(수행을 감독하는, 이해와 전략의 효율성과 적절성 판단), 학습 환경과 자원의 도움에 대한 탐색, 작업기억(효율성과 정보의 질을 판단하는 중요한 요소로 작용함) 등이다.

자기 반성적 측면은 목표, 신념, 전략, 흥미, 지식의 통합을 수정하는 기회이다.

다음은 영역 수준이다. 이 수준은 개개인별로 전문성, 메타인지 기술, 자기 효능감의 수준, 영역별로 채택된 목표에 있어서 큰 차이를 보인다. (전문성 수준의 차이에 의해서) 작업기억의 효율성, 문제 해결의 자동화, 지식 기반 사고체계의 세련됨의 차이는 학습자의 자기조절 효율성(특히 어려운 문제에서)에 영향을 준다.

세 번째는 성향 수준이다. 이 수준에는 장기 목표, 신념, 관점(세계관)에 의한 개개인의 차이가 더욱 드러난다. 또한 점검의 정확성, 융통성, 정교성 등의 영역 일반적인 요소에 의한 차이가 드러난다. 이런 측면서 자기조절 능력이 매우 중요하다.

2.2 컴퓨터 프로그래밍 교육의 중요성

본 연구에서 컴퓨터 프로그래밍 언어 학습할 때 학생들이 교수학습 효과를 최대로 하기 위한 자기조절 학습 모형을 개발하는 것이다. 본 모형을 개발하기 위한 현

장 도메인으로 학생들의 인지 구조측면에서 컴퓨터 프로그래밍 언어의 중요성을 살펴보면 다음과 같다.

컴퓨터 프로그래밍 언어 교육을 하면 창의성과 논리성을 개발하기 때문에 다른 어떤 분야보다 학생들의 인지 능력을 향상시킬 수 있다. Salomon G와 Perkins[16]는 컴퓨터 프로그래밍 교육이 학생들의 인지 능력의 전이가 빨리 일어난다. 컴퓨터 프로그램 교육은 학생이 생각하는 것을 바로 구현할 수 있기 때문에 학생들의 감성과 행동을 바로 표현하고 확인할 수 있기 때문에 다른 어떤 교과목보다 학생들의 인지 기술이 전이된다고 할 수 있다고 하였다. Salomon G와 Perkins[16]는 구체적으로 컴퓨터 프로그래밍 교육으로 수학적 기하학적 개념과 원리 향상할 수 있다고 하였다. 그 예로 생활 속에서 수학적 기하학적 개념을 컴퓨터 프로그래밍 언어를 이용하여 만들어 봄으로 학생들의 개념 형성을 쉽게 할 수 있다. 문제해결 능력 향상은 학생들이 주어진 문제들을 실제 컴퓨터 프로그램을 통해서 실 데이터를 이용하여 모의 실험해 볼 수 있기 때문이다. 문제 찾는 능력과 문제 관리 능력 향상은 학생들이 컴퓨터 프로그램에서 다양한 경우의 수를 검정하는 프로그램을 만들어서 확인할 수 있기 때문이다. 논리적인 추론과 표현 능력 향상은 학생들이 논리적인 추론을 프로그래밍 언어로 구현하여 자신의 논리를 실제 적용해 볼 수 있는 능력을 알 수 있다. 지식, 생각, 학습의 모델링 능력 향상은 학생의 데이터를 구현할 때에 다양한 정보를 이용하여 정보를 조직화하여 프로그램을 작성할 수 있는 능력 등을 알 수 있다.

Dona M. Kagan[4]은 교사가 학생들을 교수할 때에 지식정보의 구조 부분과 학생들의 인지과정에서 교수설계가 매우 중요하다고 하였다. 이는 컴퓨터 프로그램 교육에서 학생들의 실제 데이터를 기반으로 지식정보를 구조화하여 데이터 구조화를 할 수 있고, 학생들이 실제 데이터를 관리하는 절차나 방법을 모르는 상태에서 컴퓨터 프로그램 언어 교육을 할 수 없다. 학생들이 주어진 문제를 기반으로 지식정보를 구조화하여 프로그래밍 언어를 이용하여 문제 해결할 수 있는 능력을 기를 수 있다. 따라서 학생들이 실생활에서 문제 해결할 수 있는 좋은 도구가 컴퓨터 프로그래밍 언어 교육이라고 할 수 있다. 학생들이 실제 운영되고 있는 데이터를 수집하여 처리할 수 있는 큰 장점이 있다.

Nong Ye과 Gavriel Salvendy[12]는 컴퓨터 프로그래밍을 수행할 때에는 목적(objective), 개념(conceptual), 기능(functional), 논리(logical), 및 물리(physical)적인 추상화의 단계로 나누어서 프로그래밍을 수행한다고 하였다. 학생들이 컴퓨터 프로그램 교육을 할 때에 학생들이 컴퓨터 프로그램 언어의 문법을 학습하고 이 문법을 이용하여 간단한 예제 등을 만들어 학습하면 지루할 수 있다. 이때 프로그램 예제를 만들 때에 목적, 개념, 기능, 논리 및 물리적인 추상화에 관련된 예제를 만들어서 프로그램 교육을 할 수 있게 한다.

3. 자기 조절 학습 모델 개발

3.1 개요

본 연구에서 컴퓨터 프로그래밍 언어를 교육할 때에 학생들의 자기조절학습 능력을 향상시킬 수 있는 교수학습 모델을 개발한다. 본 연구에서 개발할 자기조절학습 모델은 자기 조절 학습 요소와 자기조절 학습 단계로 구성한다. 이를 기반으로 자기 조절 학습 모델을 만든다. 먼저, 다음 각 절에서 자기조절 능력을 향상시킬 수 있는 학습 요소들을 정의하고 각 요소들에 대해서 간단히 설명하고 구체적인 예를 든다. 다음은 자기조절 학습 단계를 정의하고 각 학습 단계에서 해야 할 일을 정한다. 이를 기반으로 자기조절 학습 모형을 만든다. 본 연구에서 개발한 자기 조절 학습 모형은 자기 조절 학습 요소와 소프트웨어 개발 단계에 4C(copying, changing, creating, challenging) 모델을 만든다. 4C는 학생들이 어떤 학습을 할 때에 복사하기(copying), 변형하기(changing), 창작하기(creating), 도전하기(challenging)로 자기 조절 학습 능력을 기를 수 있다.

3.2 자기조절 학습 요소

컴퓨터 프로그램 언어 학습을 위한 자기조절 학습에 영향의 미치는 요소들을 다음과 같이 정의한다.

요소1은 과제 수준 정도이다. 과제 수준은 다음과 같은 세단계로 나눈다. 단순한 개념이해 과제, 실생활의 과제, 도전적인 과제로 나눈다. 단순한 개념이해 과제는

학생들이 개념을 이해하기 위해서 간단한 과제로서 10개의 숫자들을 순서대로 배열하는 하는 과제가 구체적인 예제이다. 실생활의 과제는 실제 사회 및 자연 환경에서 발생하는 데이터를 이용하는 과제이다. 실제 예제는 우리 반 학생 30명의 학생이름을 순서대로 배열하는 예제이다. 또한 우리나라 지진 데이터를 분석하여 가장 큰 지진이 일어난 날짜를 찾아보는 것이다.

요소2는 일반화 정도이다. 일반화의 정도는 단순한 예제 단계에서 실생활에 적용하는 데이터를 사용하는 것이다. 예를 들어 숫자들을 정렬하는 프로그램을 문자들을 정렬하는 프로그램으로 적용할 수 있게 하는 것 등이다. 또한 10개의 데이터에 만족하는 것을 100개의 데이터에 만족하게 하거나 1000개의 데이터 더 나아가 모든 데이터에 만족하게 하는 것이다.

요소3은 효율화 정도이다. 자기가 개발한 프로그램이 다른 사람이 개발한 프로그램보다 좀 더 빨리 수행하는 정도이다. 같은 일을 수행하는 데 더 좋은 알고리즘을 사용하면 더 빠른 결과가 나올 수 있다. 이런 측면에서 프로그램 언어 학습을 휘해서 자기조절 학습 능력에 효율화 정도는 매우 중요한 개념이다. 데이터는 10개 중에 특정 데이터 한 개를 찾을 때에 순차적으로 찾는 방법이 효율적이지만 데이터 1만 개가 되면 순차적으로 원하는 데이터를 찾는 것은 매우 많은 시간이 소모되는 것이다. 따라서 좋은 알고리즘을 알고자 하는 자기조절 능력이 매우 프로그램 언어 교육에서 중요한 요소이다.

3.3 프로그램 학습의 자기 조절 학습 단계

본 연구에서 제안한 자기조절학습 모형에서 자기조절 학습 단계는 다음과 같은 5개의 단계로 구성한다.

첫 번 단계는 문제이해 단계이다. 문제이해 단계를 학생들 학습 요소들을 정확히 이해서 주어진 문제를 이해하는 단계이다. 문제 이해의 요소는 학생들이 반드시 입력 데이터와 출력 데이터를 이해해서 입력 데이터에서 출력 데이터로 어떻게 변환하는 것을 이해해야 한다. 문제이해를 위해서 기존의 응용 프로그램을 이용할 수 있어야 하고 실제 관련 있는 프로그램을 사용해 본 경험이 있어야 한다. 이 단계에서 과제 수준, 일반화 수준, 효율화 수준을 정의할 수 있어야 한다.

두 번째 단계는 설계 단계이다. 학생들이 프로그램 언어를 이용하여 주어진 문제를 해결하기 위해서 어떤 설계를 할 것인가이다. 설계단계는 과제 수준에 따라 일반화 정도에 따라 효율화 정도에 따라 설계에서 자기조절 능력이 다르게 된다. 설계 도구는 학생들이 기존의 설계도를 그리는 방법과 실제 설계 도구를 이용하는 방법 등이 있다. 이 단계에서도 과제 수준, 일반화 수준 및 효율화 수준에 따라 설계도 등이 매우 다르게 생성된다.

세 번째 단계는 구현단계이다. 학생들이 주어진 문제를 해결하기 위해서 문제를 이해하는 단계와 설계하는 단계를 거쳐서 실제 구현하는 단계이다. 구현 단계에서 프로그램 언어를 어떤 것을 선택하는지 등에 대해서 설명하는 것이 필요하다. 구현 단계는 설계도 형태대로 구현하는 것이다. 구현 전략은 다음과 같다.

네 번째는 시험 단계이다. 이 단계에는 시험 데이터 집합을 만들어서 시험 단계에서 원하는 결과가 나오는지 시험하는 것이다. 이때, 자기 조절 학습 전략은 다음과 같이 3가지가 있다. 전통적인 데이터를 이용하여 시험하는 전략, 입력 데이터의 주변 값을 시험 데이터로 이용하는 전략, 입력 데이터 이외의 데이터를 시험 데이터로 이용하는 전략이다. 예를 들어 1부터 어떤 수까지의 합을 계산하는 프로그램을 만들 때에 시험 데이터는 전통적인 데이터는 60을 입력하여 1부터 60까지의 합을 계산하는 방법과 -40을 입력하여 결과가 어떤 결과가 나오는지 확인하는 방법 등이 있을 수 있다.

다섯 번째 유지 보수 단계이다. 이 단계에서는 기존에 개발한 프로그램을 새로운 환경에 적용하는 것, 새로운 기능을 향상하게 하는 것, 에러를 수정하는 것, 기능을 좀 더 세련되게 하는 4개의 유형이 있다. 이 4개의 유형이 자기조절 학습 단계의 유지 보수 단계에서 자기조절 학습 전략이다.

3.4 자기 조절 학습 모델

본 연구에서 제안한 자기조절 학습 모형은 자기 조절 학습 요소와 자기조절 학습 단계를 기반으로 4C(copying, changing, creating, challenging) 모델을 제안한다. 4C의 각 단계에 대한 설명은 다음과 같다.

제1단계 모형은 모방하기(copying) 모델이다. 모방하기 모형은 문제 이해 모방하기, 설계 모방하기, 구현 모

방하기, 시험 모방하기, 유지보수 모방하기가 있다.

문제이해 모방하기는 주어진 문제에서 입력값 찾기와 출력값 찾기 모방하기를 하여 입력값과 출력값의 관계를 찾아서 변환하는 것을 그대로 모방하는 것이다. 설계 모방하기는 설계도를 그대로 따라하는 것이다. 구현 모방하기는 프로그램 소스 코드를 그대로 모방하여 실행해 보는 것이다. 시험 모방하기는 테스트 데이터는 교사가 수행하는 것을 그대로 따라하는 것이다. 유지보수 모방하기는 에러를 찾는 방법 등을 그대로 따라 하는 것이다.

구현 모방하기를 좀 더 자세히 살펴보면 학생이 컴퓨터 프로그램 코드를 그대로 따라하는 것이다. 주어진 코딩을 그대로 학생들이 모방하여 그 결과를 보면서 흥미를 느끼게 하는 것이다. C언어 코드를 그대로 따라하는 것이다. 모방하기를 하면 과제의 정도, 일반화 정도, 효율화 정도의 요소를 학습하게 하고 또한 문제이해, 설계, 코딩, 시험, 유지보수 등의 단계에서는 코딩, 시험, 유지보수 단계만 수행할 수 있다. 기본적인 예제는 다음 <Table 1>과 같다. <Table 1>과 같이 학생들이 그대로 따라하는 것이다.

<Table 1> Copying Step

```
#include <stdio.h>
void main()
{
int i,n, sum;
sum=0;
scanf("%d",&n);
for(i=1;i<=n;i++)
      sum=sum+i;
printf("1부터 %d까지의 합은 %d이다",%n,%sum);
}
```

제2단계 모형은 변형하기(changing) 모형이다. 이 모형은 주어진 단계에서 어떤 조건을 변경하여 새롭게 만드는 것이다. 이해하기 변형하기 모형, 설계 변형하기 모형, 구현 변형하기 모형, 시험 변형하기 모형, 유지보수 변경하기 모형이 있다.

이해하기 변형하기 모형은 이해하기 단계에서 입력, 처리, 출력값을 과제 수준에 따라 변형할 수 있고, 일반화의 정도에 따라 변형할 수 있다. 설계 변형하기 모형은 설계 구조도를 변경할 수 있다. 여러 개의 함수로 구현하게 할 수도 있고, 다른 입출력 문을 사용하여 변형할 수도 있다. 구현 변형하기 모형은 프로그램 코드 작성을 할 때에 언어의 구문을 다르게 사용할 수도 있고, 기존 코드에 기능을 단순히 변경할 수도 있다. 시험 변형하기 모형은 시험 데이터를 변경할 수 있는 것이다. 유지보수 변경하기 모형은 유지 보수할 때에 기본 틀을 변경할 수도 있다. 만약 구현 변경하기 모형은 1부터 주어진 수까지의 합을 변경하여 주어진 두 수의 합을 계산하는 프로그램으로 변형하는 것이다. 이것이 구현 변형하기 모형이다. 학생들에게 <Table 1>에 보여주고 따라하기를 한 후에 <Table 1>의 프로그램 소스를 입력한 두수까지의 합을 계산하는 프로그램으로 변경하게 하면 <Table 2>와 같은 결과가 나오게 한다.

<Table 2> Changing Step

```
#include <stdio.h>
void main()
{
int i,a,b, sum;
sum=0;
scanf("%d",&a);
scanf("%d",&b);
for(i=a;i<=b;i++)
      sum=sum+i;
printf("%d부터 %d까지의 합은 %d이다",a,b,sum);
}
```

제3단계 모형은 창작하기(creating)모형이다. 이 모형의 주어진 문제의 조건에 따라 모방하기 단계, 변형하기 단계를 거친 후에 직접 자신의 생각으로 만들어 보는 모형이다. 창작하기 모형은 주어진 문제를 이해하기를 직접 수행하여 설명할 수 있어야 하고, 설계하기 단계에서는 자신이 어떤 방법으로 설계한 것인지를 설명해야 하고, 구현하기 단계에서는 자신이 어떤 프로그램 코드로 직접 만들었는지를 설명할 수 있어야 하고, 시험하기 단계에서는 자신이 창의적인 시험 데이터를 만들어 보는 것이고, 유지보수하기를 직접 수행하는 것이다. 구현단계에서 창작하기 모형은 프로그램 코드를 주지 않고 직접 만드는 것이다.

제4단계 모형은 도전하기(challenging) 모형이다. 이

모형은 주어진 문제에 새로운 것을 도전하여 기존의 것과 다르게 만드는 것이다. 기존에 다른 사람들과 차이점을 분석할 수 있고, 직접 만든 것이 어떤 점에서 우수한지를 설명할 수 있어야 하고 직접 비교할 수 있어야 한다. 문제 이해하기 단계에서는 매우 어려운 문제들을 쉽게 이해할 수 있게 도전해 보는 것이다. 영화 이미테이션에서 암호문을 이해해 보는 것이 한 예이다. 이 부분은 과제 수준과 밀접한 연관 관계가 있다. 두 번째 단계는 설계 단계이다. 설계 단계에서는 과제 수준이나 학습 요소 등에 따라 창의적인 설계를 도전해 보는 것이다. 이런 관점은 효율성 문제와 일반화 문제 등이 모두 관련 되어 있다. 코딩 단계에서는 새로운 코드로 도전해 보는 것이다. 시험 단계 도전은 시험의 경우의 수가 매우 많은데 이를 줄이는 방법 등이 도전하기 모형이다. 유지보수는 에러를 수정하거나 새로운 환경에 프로그램을 수정하는 것 등이 도전하는 것이다.

4. 실험 적용 분석

4.1 실험 모형 개요

본 연구에서 실험 모델은 다음과 같이 만들었다. 먼저, 자기 조절 학습 요소는 본 연구에서 제안한 학습 요소들인 과제 수준, 일반화, 효율화 3가지 측면에서 측정하였다. 측정방법은 과제 수준은 초등학교 학년을 기반으로 현재의 교과 내용을 기반으로 5단계 척도이다. 3이 현재의 학년이고 4는 1학년 위의 교육과정이고, 5는 2학년 이상의 교육과정이고, 2는 1학년 아래의 교육과정이고, 1은 2학년 이상 아래의 교육과정이다. 일반화 정도는 같은 문제를 내어 1부터 10까지의 합을 계산할 수 있다가 가정하면 3이고 이를 1000까지의 합을 계산할 수 있으면 4이고, 어떤 경우라고 할 수 있으면 5의 척도를 둔다. 효율화의 정도는 현재 만들 것을 3이라고 하면 알고리즘적으로 개선하면 4이고, 최신 개념을 찾아내면 5라는 척도를 둔다. 두 번째는 자기 조절 학습 단계는 단순히 문제 이해, 설계, 코딩, 시험, 유지보수의 5단계를 두어 평가한다. 세 번째는 학생들이 학습을 하면 어느 단계까지 자기 조절 학습 모형에 완성하는지를 4C(copying, changing, creating, challenging)로 결정하게 한다.

4.2 상관관계 분석

일반적으로 자기조절 학습 모형의 타당도를 검증할 때에 종속변수를 성취도를 많이 이용한다[10]. 본 연구에서도 자기 조절 학습에 필요한 각 요소들과 성취도 간의 상관관계를 분석한다.

본 연구에서 참여한 초등학생들에 대해서 학습 주제를 8개 주제를 분석하여 학생들의 14명의 데이터를 기반으로 하였다. 본 연구에서 제안한 모델 데이터와 학생들의 성취 점수 간의 상관관계 표는 다음 <Table 3>과 같다.

<Table 3> Estimated Correlation Between Self-regulated Learning Model and Academic Performance

Subarea	Components	Correlation
learning elements	task level	0.32**
	general level	0.48**
	efficiency level	0.52**
learning step	understanding	0.38**
	design	0.41**
	coding	0.24**
	testing	0.21*
	maintain	0.32**
learning model	copying	0.03
	changing	0.21*
	creating	0.44**
	challenging	0.32**

(**p<0.01, *p<0.05)

<Table 3>에서는 연구에서 제안한 자기 조절 학습 모형의 3개의 하위 요소들과 학업 성취도와 상관관계들은 복사하기(copying) 항목들을 제외하고 유의 수준 1% 이내로 상관관계는 매우 높다는 것을 알 수 있다. 세부적으로 분석하면 다음과 같다.

첫 번째 학습 요소들 간의 분석은 과제 수준, 일반화 정도, 효율화 정도가 성취도와 매우 상관관계가 높다는 것을 알 수 있다. 과제 수준에서 일반화와 효율화로 세분화할 수 있지만 이들 간의 상관관계도 매우 높다는 것을 알 수 있다. 효율화 정도가 0.52이고, 일반화 정도가 0.48이고, 과제 수준이 0.32이다. 이 세 요소 중에서 과제수준보다는 컴퓨터 프로그래밍 교육에서는 효율화 정도가 가장 상관관계가 높고 다음은 일반화 정도라고 볼 수 있다. 이것은 컴퓨터 프로그래밍 교육에서 과제

수준보다도 일반화와 효율화가 더 중요하다는 것을 알 수 있다.

학습 단계는 학습 요소보다 상관관계가 높지 않다는 것을 알 수 있지만 설계 단계와 문제 이해 단계가 성취도와 상관관계가 매우 높다는 것을 알 수 있었다. 구체적으로 문제 이해 단계에는 0.38이고, 설계 단계에서는 0.41이고, 구현 단계에서는 0.24이고, 시험 단계에서는 0.21이고, 유지보수 단계에서는 0.32이다. 단계에서 상관관계가 높은 것은 설계 단계, 문제 이해 단계가 매우 높고 구현 단계는 상대적으로 낮다는 것을 알 수 있다. 이는 프로그래밍 교육에서 단순하게 구현하는 데 초점을 맞추는 것이 아니라 설계와 문제 이해 단계에 초점을 맞추어야 한다. 다만 유지 보수 단계에서도 구현단계보다 상관관계가 높다는 것을 알 수 있다.

학습 모델에서도 복사하기 단계를 제외하고 상관관계가 높다는 것을 알 수 있었다. 복사하기는 0.03이고, 변형하기가 0.21이고, 창조하기가 0.44이고 도전하기가 0.32이다. 이 부분에서 학생들이 자기조절 학습을 위해서 창조하기와 도전하기 모델에서 많은 성취를 보인다는 것을 알 수 있다.

5. 결론

지금 소프트웨어 교육을 강조하고 있는 상황에서 초등학생들이 컴퓨터 프로그래밍 언어를 학습할 때에 자기조절학습 요소와 단계 및 모델을 만들어서 초등학교 컴퓨터 프로그램 수업에 활용하는 것은 매우 의미 있다. 본 연구에서 제안한 자기조절 학습 모형에서 학습 요소, 학습 단계 및 학습 모형의 3차원 구조로 프로그래밍 언어를 학습하게 하는 것은 매우 의미 있는 것이고 또한 3차원 구성의 각각의 구성요소들은 복사하기 모형을 제외하고 모든 것이 의미 있다. 특히, 학습 요소로서 일반화와 효율화가 과제 수준보다는 성취도와 상관관계가 높다는 것을 알 수 있고, 학습 단계에서 문제이해 단계와 설계 단계가 구현 단계보다 중요하다는 것을 알 수 있고, 학생들이 단순하게 복사하기보다는 변형하고, 창조하고, 도전하는 것이 더 중요하다는 것을 알 수 있다.

참고문헌

[1] B. J. Zimmerman (1990). Self-regulated learning and academic achievement: An overview. *Educational Psychologist, 25*(1), 3-17.

[2] Boekaerts, M. (1995). Self-regulated learning: Bridging the gap between metacognitive and meta-motivation theories. *Educational Psychologist, 30*, 195-200.

[3] Boekarts, M., Pintrich, P. R., & Zeidner, M. (Eds.), (2000). Handbook of self-regulation. San Diego, CA: Academic.

[4] Dona M. Kagan (1989). Research on Computer Programming as a Cognitive Activity: implications for the study of classroom teaching. *Journal of Education for Teaching, 15*(3), 177-189.

[5] Garris, R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. *Simulation & Gaming, 33*, 441-467.

[6] Gee, J. P. (2003). What video games have to teach us about learning and literacy. New York: Palgrave Macmillan.

[7] Gee, J. P. (2005). Learning by design: Good video games as learning machines. *E-Learning, 2*, 5-16.

[8] Hur, Jae Eun & Kim, Hong Chan (2014). The effect of the Self-Regulated Learning Ability on the Process of Selecting Assisted-Learning in the Middle School Mathematics Education. *Journal of the Korean School Mathematics Society, 17*(1), 1-21.

[9] Kapsu Kim (2014). Measuring and Applying the Self-efficacy in Computer Programming Education. *Journal Of The Korean Association Of Information Education, 18*(1), 111-120.

[10] Lan, W. (2005). Self-Monitoring and its relationship with educational level and task performance. *Educational Psychology, 25*(1), 109-127.

[11] Miltiadou, M., & Yu, C. H. (2000). Validation of the Online Technologies Self-Efficacy Scale (OTSES)(Publication no. ED445672). Retrieved January 15, 2004.

[12] Nong Ye Gavriel Salvendy (1996). Expert-novice knowledge of computer programming at different levels of abstraction. *Ergonomics, 39*(3), 461-481.
[13] Pintrich, P. R., & De Groot, E. V. (1990). Motivational and self-regulated learning components of classroom academic performance. *Journal of Educational Psychology, 82*, 33-40.
[14] Pressley, M., & Ghatala, E. S. (1990). Self-regulated learning: Monitoring learning from text. *Educational Psychologist, 25*, 19-34.
[15] Pressley, M., & Ghatala, E. S. (1990). Self-regulated learning: Monitoring learning from text. *Educational Psychologist, 25*, 19-34.
[16] Salomon G & Perkins D. N. (1987). Transfer of cognitive skills from programming: When and how?. *Journal of Educational computing Research, 3*, 149-169.
[17] Schraw, G. (2007). The use of computer-based environments for understanding and improving selfregulation. *Metacognition and Learning, 2*, 169-176.
[18] Schraw, G., Crippen, K. J., & Hartley, K. (2006). Promoting self-regulation in science education: Metacognition as part of a broader perspective on learning. *Research in Science Education, 36*, 111-139.
[19] Youngsu Goo & Yeonsuk Yang (2013). A Comparison Study of Learning Style, Self-regulated Learning and Learning Flow between Gifted and Normal Student. *Journal of Gifted/Talented Education, 23*(2), 177-191.

저자소개

김 갑 수

1985.2 서울대학교계산통계학과 (학사)
1987.2 서울대학교 계산통계학과 전산학전공(석사)
1996.2 서울대학교 계산통계학과 전산학전공(박사)
1987～1992 삼성전자 사원-과장
1995～1998 서경대학교 전임강사-조교수
1998～현재 서울교육대학교 컴퓨터교육과 교수
관심분야: 컴퓨터 교육, 소프트웨어 공학, 정보 영재, 기능성 게임
e-mail: kskim@snue.ac.kr