Journal of the Korea Academia-Industrial cooperation Society
Vol. 20, No. 2 pp. 587-592, 2019

https://doi.org/10.5762/KAIS.2019.20.2.587
ISSN 1975-4701 / eISSN 2288-4688

소프트웨어 기초 교육의 최적 운영 알고리즘에 관한 연구

구은희[1], 우찬일[2*]
[1]아주대학교 다산학부대학, [2]서일대학교 정보통신공학과

A Study on an Operational Optimization Algorithm of Software Basic Education

Eun-Hee Goo[1], Chan-Il Woo[2*]
[1]Dasan University College, Ajou University
[2]Department of Information and Communication Engineering, Seoil University

요 약 최근 들어 소프트웨어 경쟁력을 확보하기 위한 기술이 스마트폰과 IoT 기술이 맞물려 새로운 사업으로 확장되고 있어 소프트웨어 기술에 대한 중요성은 더욱 부각 되고 있다. 따라서 글로벌 소프트웨어 산업의 주도권 확보와 융합형 소프트웨어 인재 양성을 위해 우수한 소프트웨어 개발 인력의 필요성은 점점 더 증가하고 있다. 본 논문에서는 융합형 소프트웨어 산업 인력 확대를 위해 소프트웨어를 필수 교과로 운영한 사례를 기반으로 소프트웨어에 대한 기본 인식과 소프트웨어 개발의 필요성 그리고 소프트웨어 개발을 위한 코딩 능력 향상에 대하여 분석한다. 분석 결과, 코딩 능력 향상을 위한 실습 방법 중 학습자 중심에서 진행된 기술적 내용은 소프트웨어에 대한 인식과 개발의 필요성 측면에서 긍정적인 효과를 나타내고 있으며 코딩 능력 향상에 중요한 요소가 되는 것으로 분석되었다. 본 논문에 나타난 연구 결과에서 프로그램 개발의 필요성과 능동적인 참여는 실무 능력 향상을 위해 매우 중요한 부분이라는 것을 나타내고 있으며, 이러한 결과는 소프트웨어 개발 능력 향상을 위한 방법론 측면에서 의미 있는 결과를 제시하고 있음을 알 수 있다.

Abstract The importance of software technologies is becoming more prominent because of the competition to secure a competitive edge in software, which has been intensified since the emergence of smartphones and IoT. Thus, to assure the initiative in the global software industry and to foster superior human resources, there is a growing need for outstanding software development professionals. This paper analyzes the factors that affect the basic perception of software, the need for software development, and the enhancement of software coding ability based on a compulsory software class, which aims to increase the workforce of the converged software industry. The analysis shows that among other technical practices to enhance coding ability, learner-centered technical contents showed the most positive effect regarding the recognition and motive of development and are an essential factor in improving coding skills. The findings indicate that the need for program development and active involvement in the development of the program are the most important factors in improving the practical ability. The analysis presents meaningful results by suggesting a methodology for improving software development capabilities.

Keywords : Software, Algorithm, Coding, Correlation Coefficient, Computer Programming

1. 서 론

지식정보화 사회로의 변화를 의미하는 4차 산업혁명은 산업구조를 근본적으로 변화시키고 사회, 교육, 직업, 문화 등의 개인 삶의 방식에 엄청난 변화를 가져왔다. 빅데이터, 인공지능, 지능형 로봇, 자율주행차, AI변호사

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학사업의 연구결과로 수행되었음.(2015-0-00908)
*Corresponding Author : Chan-Il Woo(Seoil Univ.)
Tel: +82-2-490-7556 email: ciwoo@seoil.ac.kr
Received November 27, 2018 Revised December 28, 2018
Accepted February 1, 2019 Published February 28, 2019

등의 기술 기반인 지능화로 인하여 산업구조의 근본이 변화되었고 사물인터넷과 ICT 플랫폼 기술을 이용하여 산업 간의 경계가 허물어지고 있어 다양한 사업영역이 만들어지고 있다. 세계경제포럼에 따르면 2020년까지 선진국에서만 자동화로 대체되는 일자리가 확대되어 약 710만개의 일자리가 사라질 것으로 예측하였다. 또한 현재 초등학교 어린이의 약 65%는 기존에 존재하지 않았던 새로운 고부가가치 및 창의적인 직무에 종사하게 될 것으로 전망하여 고용 구조가 지식정보화에 따른 변화를 가져오고 있음을 보이고 있다[1,2].

이와 같은 변화는 다양한 학문과 접목된 ICT를 바탕으로 한 소프트웨어가 중심으로 구현되어 개인 · 기업 · 국가의 경쟁력 혁신과 성장을 통한 가치창출이 이루어지기 때문이다. 따라서 현대와 미래사회의 인재가 갖추어야 할 필수적인 역량으로 소프트웨어가 매우 중요하게 되었다.[2,3] 이에 우리나라는 “국가 차원의 아키텍트 육성을 통한 소프트웨어 산업 기반 강화”의 핵심 요소로 소프트웨어 전체 구조를 이해하고 기획, 설계 및 개발 능력을 가진 융합 소프트웨어 인력임을 인지하여 SW 중심 사회로의 전환에 따른 인재 육성 방안을 마련하였다[1].

SW 중심 대학의 SW 교육은 단순한 코딩 교육이나 프로그래밍 자체가 목적이 아니라 각 학문에서 발생한 문제를 해결하기 위해 컴퓨팅 사고를 기반으로 기본 개념과 원리의 이해를 갖추어야 할 필요가 있다[4]. 컴퓨팅 사고는 누구나 갖추어야 하는 기본 역량으로 읽기, 쓰기, 셈하기와 같이 인간의 가장 기본적인 항목으로 강조되어 현재 모든 분야에 걸쳐 일반적인 문제 해결에 필요한 능력으로 인식되고 있다[5]. 그리고 컴퓨팅 사고를 통하여 프로그래밍 과정속에서 학습자는 학습의 주체로써 해결방안을 찾고 이를 즉각적인 피드백을 통해 문제해결 능력, 절차적 사고 능력, 추론 능력 등을 기를 수 있다고 하였다[6,7]. 따라서 이 같은 컴퓨팅 사고를 기반으로 다양한 전공의 학습자들이 SW에 대한 관심과 흥미를 가지고 학습 활동에 적극적으로 참여할 수 있도록 교과과정이 운영될 필요성이 있다[4]. 이는 전공자는 당연히 소프트웨어 교육을 필수 수업으로 진행함에 있어 꼭 필요하다고 느끼고 있는 반면 비전공자들은 필수 SW 교육에 대한 부정적인 인식을 개선하기 위하여 꼭 필요한 사항이다. 오미자(2017)의 연구에 따르면 프로그래밍 교육에 대한 효과는 대학에서도 공감하고 있지만 비전공자들의 소프트웨어 교육에 대한 배려나 구체적인 방향 등이 정해지지 않아 반발이 있는 상태로 체계적인 교과과정이 필요하다고 조사되었다. 또한 일방적인 수업 보다는 학습자들에게 소프트웨어의 필요성과 각 전공에서 유용하게 사용할 수 있음을 보여준다면 성취도 향상을 가져올 수 있고 수업의 방법도 프로그래밍의 문법 이해만이 아닌 실습과 개인별 수준을 고려한 과제 제시를 통하여 성취도를 높이는 형태로의 수업 방법으로 변화를 하여야 한다고 연구되었다[7]. 소프트웨어 인재 양성을 위한 대학 교육은 비전공자들의 전공에서도 소프트웨어가 꼭 필요함을 인식시키고 학습자가 스스로의 학습 활동을 통한 자기주도적인 학습 방법을 활용하여 소프트웨어에 대한 긍정적인 인식을 가질 수 있는 형태로 이루어져야 할 필요가 있다[8].

본 연구에서는 비전공자들의 소프트웨어 교육의 필요성과 효과에 대하여 학습자 중심에서 프로그램 수업이 진행된 후 나타난 결과를 비교 분석한다. 이를 위해 소프트웨어 기초 수업의 시작과 함께 조사된 결과와 한 학기 수업이 모두 진행된 후의 결과를 조사하였으며, 본 연구의 결과는 소프트웨어에 대한 학생들의 중요도 인식 정도와 현재 수준을 파악하고 교과 운영에 효과적인 알고리즘을 제공하고자 한다.

2. 연구 배경

2.1 소프트웨어의 필요성

소프트웨어 교육은 단순히 응용소프트웨어 등의 활용법을 익히는 것이 아니라 다양한 문제의 해결법을 컴퓨터를 기반으로 찾는 것이다. 즉, 문제의 효율적인 해결을 위하여 다양한 자료를 수집하고, 분석하며 당면한 문제의 효율적인 해결 방법을 찾기 위한 컴퓨팅 사고력을 증진하는 것을 말한다. National Academy of Sciences 에서는 급변하는 사회와 발전하는 기술에 학생들이 자신의 일상생활에서 생산적인 대처를 위해 기초개념과 현대기술 그리고 지적능력을 제시하였다[9].

기초개념은 컴퓨팅의 기본 원리와 개념을 나타내고, 현대기술은 다양한 컴퓨터 응용 프로그램을 사용할 수 있는 능력을 나타낸다. 그리고 지적능력은 특정 상황에 대하여 알고 있는 정보 기술을 적용하고 이 기술을 사용하여 새로운 문제를 해결 할 수 있는 능력을 나타낸다

[10]. Walker(1993)의 연구에 의하면 컴퓨터과학 비전공 과정은 학생들이 컴퓨팅 사고를 함에 있어 자신의 기술을 향상 시키는데 도움이 되도록 노력하고 솔루션을 작성하고 비교 분석 할 수 있어야 한다고 하였다[11]. 따라서 소프트웨어 교육은 코딩 기술만을 배우는 것이 아니라 소프트웨어를 이용하여 문제 해결 능력을 향상 시킬 수 있도록 한다.

2.2 소프트웨어 교육 정책

소프트웨어 중심 사회는 컴퓨팅 사고력을 가진 창의·융합 인재를 양성하는 것을 목표로 하고 있다. 이를 위하여 초등학교에서는 2019년도부터 건전한 정보윤리 의식을 바탕으로 알고리즘과 프로그래밍의 창의적 체험 활동을 통해 연간 17시간 이상을 확보할 예정이며, 중학교에서는 2018년도부터 간단한 알고리즘을 설계하고 프로그램을 개발하는 정보 교과 등을 통하여 연간 최소 34시간 이상을 확보하여 소프트웨어 교육을 실시하고 있다. 그리고 고등학교에선 효율적인 알고리즘을 설계하여 문제를 해결할 수 있도록 운영하고 있다[12]. SW 중심대학에서는 필수 교과로 지정하여 운영하고 있으며, 2019년에는 전국 30개 대학이 SW 중심 대학으로 선정되어 전공자와 비전공자들에게 각 대학의 교육목표를 기준으로 소프트웨어 기술교육이 운영될 예정이다[7].

3. 연구 방법

3.1 연구 대상

본 연구는 SW 중심 대학의 2017학년도 소프트웨어 비전공자 Table 1의 학생들을 대상으로 연구를 진행하였다.

Table 1. Research Subjects

2017			Number of Respondents	Percentage (%)	Total (Number)
1st Semester	Pretest	Male	31	28	109
		Female	78	72	
	Posttest	Male	24	22	109
		Female	85	78	
2nd Semester	Pretest	Male	57	50	115
		Female	58	50	
	Posttest	Male	56	49	115
		Female	59	51	

연구 결과는 Table 1의 학생들을 대상으로 2017학년도 1학기 사전 조사 109명, 1학기 수업 후 109명, 2017학년도 2학기 사전 조사 115명, 2017학년도 2학기 수업 후 115명에 대하여 분석하였다. 성별로는 2017학년도 1학기에는 여학생 비율이 남학생에 비해 2배 이상이고, 2학기에는 남학생의 비율과 여학생의 비율이 거의 동일하게 나타났다.

연구 기간은 2017학년도 1학기, 2학기로 사전 조사는 학기 시작 후 수업 1~3주 이내에, 사후 조사는 학기가 끝나기 1~2주 사이에 이루어졌으며, 강의와 이론 중심의 수업이 아닌 학습자 중심의 실습이 매 시간마다 이루어졌다.

3.2 연구 내용

사전 조사와 사후 조사의 구성은 소프트웨어 기초 수업의 직접적인 성적과는 별개로 이루어졌으며, 사전설문은 학생들의 소프트웨어 교육 정도와 기초 인식 수준을 확인하기 위한 문항으로 구성되었고, 사후 조사는 수업이 이루어진 후 소프트웨어에 대한 인식 수준을 확인하는 문항과 학습자 중심 수업이 이루어진 후의 만족도 등을 파악하기 위한 것으로 구성되었다. 학생들은 5단계 평점 척도 문항에 5점은 '매우 그렇다'에서 1점인 '매우 아니다'의 응답을 하는 구조적인 설문과 자유로운 의사를 밝힐 수 있는 비구조적인 설문지를 만들어 진행하였다.

Table 2. Contents of the pretest

Content	Question Number
Software is essential (tool) in everyday life	Q_1
In general, software is a useful (tool) for me.	Q_2
Have you ever had any other software classes before entering the college?	Q_3
Software education received before entering college will be helpful when attending the software-related courses after admission.	Q_4
I am interested in programming classes that is opened for software non-majoring students.	Q_5

사전 조사는 Table 2에 나타낸 것과 같이 전반적인 소프트웨어 교육에 대한 인식 정도와 대학 입학 전 소프트웨어 교육 실시 여부와 실습수업에 대한 기대 정도를 확인하는 문항으로 구성되었고, 사후 조사는 Table 3에 제시된 것 같이 소프트웨어 교육의 인식 정도와 학습자

중심의 수업 진행 후에 대한 인식 변화 파악을 위한 것으로 구성하였으며, 조사 결과에 대한 분석은 다음과 같은 연구문제를 설정하여 결과를 분석하였다.

Table 3. Contents of the posttest

Content	Question Number
Software has the social importance.	QA_1
I think software is useful for me.	QA_2
SW Basic education for SW non-majoring students is required/needed.	QA_3
Have you ever taken a software course that is lecture-based after entering university?	QA_4
Practical software education was more helpful in understanding classes than theoretical software education.	QA_5
The difficulty of 'theories' in the lecture were adequate to understand the concept of data analysis.	QA_6
The difficulty of 'practices' were adequate.	QA_7
Through software courses, I was able to learn new knowledges and experience new things.	QA_8
I got interested in analyzing the data in my major field after taking the software courses/curriculum.	QA_9
I've got interested in SW education after I took the Software classes/curriculum.	QA_10

연구문제 1. 교육생들의 소프트웨어 교육에 대한 기본 인식은 어떠한가?

연구문제 2. 소프트웨어 교육 후 학습 효과에 대한 인식은 변화되었는가?

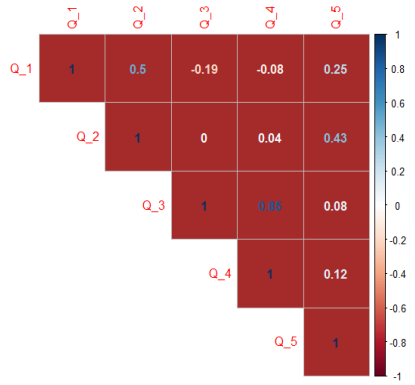
연구문제 3. 학습자 주도 교육 후 소프트웨어 기본교육에 대한 인식 개선이 되었는가?

또한 조사 결과에 대한 분석은 R-Studio인 데이터 분석 도구를 이용 하였으며, 정량 통계 처리는 기본 제공 도구를 사용하고, 분석 방법으로는 두 확률 변수 사이의 관련성을 파악하는 상관 분석 기법을 사용하였다. 상관분석에서는 피어슨 상관계수를 사용하여 두 변수 간의 선형적인 상관관계를 측정하는데, 0에 가까운 값이 나올수록 상관관계가 없다는 것을 나타내며, -1 또는 1에 가까운 값이 생성되면 상관관계가 있음을 나타낸다.

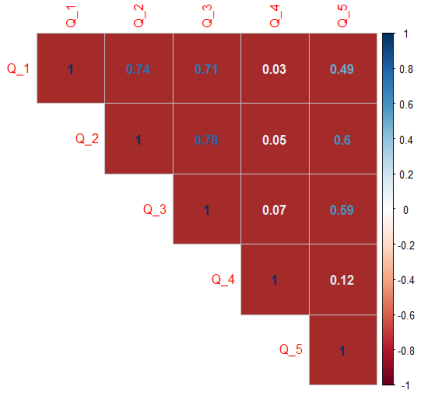
4. 연구 결과

4.1 연구 결과 분석

소프트웨어 교육의 기본 인식에 대한 항목별 상관관계 분석 결과는 Fig. 1과 같다.



(a) 1st Semester



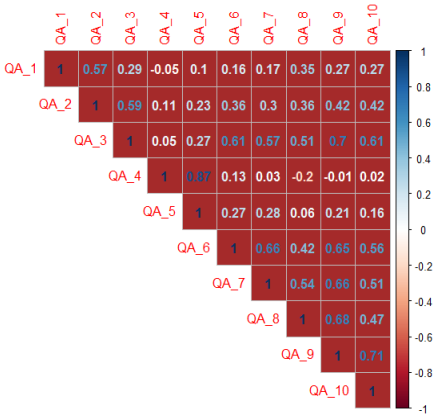
(b) 2nd Semester

Fig. 1. Pretest Correlation Coefficient Result

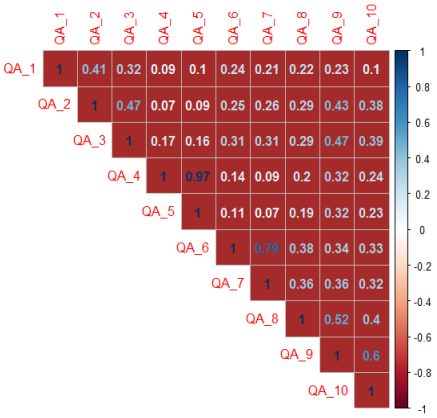
상관관계 분석을 통한 상관계수 결과를 살펴보면 1학기 사전 조사 Q_1에 대한 Q_5의 상관계수는 0.25로 약한 상관성을 나타내고 있다. 이러한 이유는 비전공자들의 경우 소프트웨어 기초교육에 대하여 부정적인 인식을 가지고 있기 때문이다. 또한 Q_2에 대한 Q_5의 상관계수도 0.43으로 약한 상관성을 나타내고 있어 유용한 도구임에도 불구하고 소프트웨어 교육에 대한 인식이 좋지 않은 것을 알 수 있다.

2학기 사전 조사 상관 계수로 Q_1에 대한 Q_5 상관계수는 0.49로 상관성을 나타내고, Q_2에 대한 Q_5의 상관계수는 0.6으로 상관성이 있음을 나타낸다. 2학기의 경우 1학기 때 보다는 상관계수가 높아진 것으로 학생들이 소프트웨어 교육에 관심이 생겼다는 것을 알 수 있다.

비전공자들의 소프트웨어 교육 후 학습 효과에 대한 인식은 변화되었는가에 대한 항목별 상관관계 분석 결과는 Fig. 2에 나타내었다.



(a) 1st Semester



(b) 2nd Semester

Fig. 2. Posttest Correlation Coefficient Result

1학기 사후 조사 QA_1에 대한 QA_2의 상관계수는 0.57로 상관성이 있음을 나타내고 있다. 또한 QA_2에 대한 QA_3의 상관계수는 0.59로 상관성이 있음을 나타낸다. 이것은 소프트웨어는 사회적으로 중요하고 나에게 유용하므로 비전공자들을 대상으로 하는 기초교육이 필요한 것임을 인지하였다는 결과로 보여진다. 즉, 수업 전에는 소프트웨어 교육의 필요성에 대한 부정적 인식이 강한 반면 수업이 이루어진 후에는 필요하다는 긍정적 인식이 늘었다는 것을 나타내는 것이다.

2학기 상관계수 결과인 QA_1에 대한 QA_2의 상관계수는 0.41로 약한 상관성을 나타내고, QA_2 항목에 대한 QA_3의 상관계수 또한 0.47로 약한 상관성을 나타낸다. 이 항목에서는 특이하게 소프트웨어 기초교육의 인식이 수업 전보다 조금 나빠진 것으로 조사되었다. 이러한 이유를 분석한 결과 1학기와 2학기의 수업과정은 동일하게 이루어졌으나 학습자 남·녀의 성비가 다르게 나타났으며, 이것은 남학생보다 여학생이 교육 후 소프트웨어에 대한 인식이 바뀌는 정도가 크다는 것을 의미한다. 학습자 주도 형태의 교육 후 비전공자들의 소프트웨어 기초교육 인식에 대한 개선에 대한 항목별 상관관계 분석 결과는 다음과 같다.

상관관계 분석을 통한 상관계수는 Fig 2의 1학기 사후 조사 QA_5에 대한 QA_6의 상관계수가 0.27로 아주 약한 상관성을 나타내고 있으며, QA_6에 대한 QA_7의 상관계수는 0.66으로 이론 중심의 수업보다는 본인이 참여하는 학습자 주도 교육의 상관성이 높다는 것을 나타내고 있다. 2학기 사후 조사에서는 QA_5에 대한 QA_6의 상관계수는 0.11로 1학기와 마찬가지로 아주 약한 상관계수를 보여주며, QA_6에 대한 QA_7의 상관계수는 0.79로 매우 높은 상관관계를 가지는 것으로 확인된다.

이러한 결과는 학습자가 주도적으로 임하면 암기식보다 학생들이 받아들이는 학습효과가 더 크다는 것을 나타낸다. 또한 1학기와 2학기의 QA_4에 대한 QA_5의 상관계수 결과를 살펴보면 1학기에는 0.87, 2학기에는 0.97이라는 아주 높은 상관계수 값을 나타내고 있다. 이는 이론 중심으로 운영되는 소프트웨어 교과를 수강한 학생들이 학습자 주도 교육을 받은 뒤 이론 중심의 소프트웨어 교육보다 실습 중심의 소프트웨어 교육이 수업을 이해하는데 더 도움이 되는 것으로 조사되었다.

5. 결 론

본 연구에서는 비전공자들의 경우 소프트웨어를 필수로 다룰 수 있는 능력이 필요함에도 불구하고 부정적인 인식을 가지고 있다는 것과 단기간의 노력으로 소프트웨어 기초 교육에 대한 부정적인 인식을 바꾸기에는 어려움이 있다는 것을 알 수 있었지만, 다양한 실습 문제를 통하여 학습자 주도로 이루어지는 소프트웨어 수업은 이론만을 진행하는 수업보다 친숙하게 관심을 유도할 수 있다는 것을 알 수 있었다.

따라서 학습자들이 쉽게 따라 할 수 있는 수준의 다양

한 실습 문제를 적용하여 소프트웨어 개발 방법과 기술을 기반으로 교육이 진행될 경우 학습효과를 높이는데 매우 효과적이라는 것을 알 수 있었으며, 나아가 소프트웨어 기초 교육에 대한 부정적인 인식이 수업이 진행된 후 긍정적인 인식으로의 변화가 발생하는 것을 알 수 있었다.

본 연구는 일부 학생들만을 대상으로 진행하였기 때문에 연구 결과를 일반화 하기에는 한계가 있을 수 있으나 기존의 암기식 수업 방법들에 비하면 학습자 중심의 수업이 보다 더 효과적이라는 결과를 얻을 수 있었다. 향후 연구에서는 보다 다양한 결과를 통하여 소프트웨어 교육의 필요성과 중요도 그리고 소프트웨어 개발 기술 향상에 대한 연구를 진행할 예정이다.

References

[1] Software Policy & Research Institute, “Softpower Korea 2025”, pp. 215, Software Policy & Research Institute, 2017.

[2] World Economic Forum, “The Future of Jobs-Employment, Skills and Workforce Strategy for the Fourth Industrial Revolution”, pp. 12, World Economic Forum, 2016.

[3] Ministry of Science and ICT, “Software-Oriented Society Realization Strategy”, pp. 26, Ministry of Science and ICT, 2014.

[4] Eunkyoung Lee, “Perspectives and Challenges of Computing Education: Interdisciplinary Approaches for Collaborative Problem Solving and Computational Thinking”, Proceedings of the Korean Society of Computer Information Conference, Vol. 21, No. 2, pp. 203-206, 2013.

[5] Wing JM, “Computational Thinking”, Communication of the ACM, Vol. 49, No. 3, pp. 33-35, 2006.

[6] Tae-Wook Lee, In-Hwan Yoo, Chul-Hyun Lee, ICT Education Theory, pp. 428, Hyungseul Publishing, 2001.

[7] Mi-ja Oh, “Non-Major Students' Perceptions of Programming Education Using the Scratch Programming Language”, The Journal of Korean Association of Computer Education, Vol. 20, No. 1, pp. 1-11, 2017.

[8] Chae Young Soog, “Research Trends on Project-based Learning”, The 33th International Conference of the Association of North-East Asian Cultures, pp. 143-147, 2016.

[9] Yuri Kim, Yongeun Moon, “Study on the Use of Public Open Data for Software(SW) Education”, The Journal of Internet Electronic Commerce Research, 15(6), pp. 245-261, 2015.

[10] Committee on Information Technology Literacy, Being fluent with information technology, National Academy Press, Washington, DC, 1999.

[11] Walker HM., “Computational thinking in a non-majors CS course requires a programming component”, ACM New York, NY, USA, Vol. 6, No. 1, pp. 58-61, 2015. DOI: https://doi.org/10.1145/2727126

[12] Ministry of Education, “Software Education Operating Guide”, pp. 23, Ministry of Education, 2015.

구 은 희(Eun-Hee Goo) [정회원]

- 2004년 8월 : 단국대학교 대학원 전자컴퓨터공학과 (공학석사)
- 2009년 8월 : 단국대학교 대학원 전자컴퓨터공학과 (공학박사)
- 2013년 3월 ~ 2014년 9월 : ㈜도넛시스템 LSI 이미징사업부 책임연구원
- 2014년 10월 ~ 2016년 8월 : ㈜이너트론 이동통신연구소 수석연구원
- 2016년 9월 ~ 현재 : 아주대학교 다산학부대학 교수

<관심분야>
정보보호, 암호 알고리즘, 서비스로서의 보안(ASCaaS), 소프트웨어 공학

우 찬 일(Chan-Il Woo) [종신회원]

- 1995년 2월 : 단국대학교 대학원 전자공학과 (공학석사)
- 2003년 2월 : 단국대학교 대학원 전자공학과 (공학박사)
- 1995년 11월 ~ 1997년 2월 : LG 이노텍(주) 연구원
- 2004년 3월 ~ 현재 : 서일대학교 정보통신공학과 교수

<관심분야>
정보보호, 암호 프로토콜, 디지털워터마킹, 소프트웨어 공학