컴퓨팅사고력의 역량 탐색 연구: 소프트웨어개발자를 중심으로

박성빈† · 안성진††

요 약

소프트웨어 기반의 새로운 기술들이 증가함에 따라 소프트웨어 개발자들에게 창의적 문제해결 방법이 요구 되고 있다. 특히 소프트웨어 개발자들이 신산업분야로의 성공적인 전환을 위해서는 소프트웨어의 분석 및 설계 단계가 중요하며 이러한 역량을 강화하는 것은 중요한 과제라고 할 수 있다. 본 연구는 두 개의 독립된 연구를 수행하였다. 먼저 소프트웨어 개발자 300명을 대상으로 소프트웨어 개발자들의 “표준 개발 프로세스” 적용으로 인한 사고의 획일성으로 인해 창의적 문제해결 접근하는 것이 어렵다는 것을 탐색하였고, 이 결과를 바탕으로 IT종사 경력 10년 이상의 소프트웨어 개발전문가 111명을 대상으로 하여 컴퓨팅사고력의 9가지 역량 중에서 어떠한 역량이 소프트웨어 개발전문가에게 영향을 주는지를 확인하였다. 그 결과 추상화가 중요한 역량인 것으로 확인되었다. 본 연구 결과 소프트웨어 개발자들은 설계단계를 가장 중요하다고 인식하였으며 CT의 추상화 역량이 다른 역량의 기본이라고 판단하였다.

주제어 : 컴퓨팅사고력, 추상화, 소프트웨어개발자, 표준개발프로세스, 설계 단계

An Exploratory Study of the Competence of Computational Thinking: for Software Developer

Seongbean Park† · Seongjin Ahn††

ABSTRACT

Due to increase in software-based new technologies, software developers are being required to creative problem solving. Especially, for a successful business change of software developers into a new industrial field, software analysis phase and design phase are important and it is an important task to strengthen these competences. In this study, two independent researches were conducted. First, a survey with 300 software developers was conducted and it was explored that uniform thinking of software developers caused by application of “standard software development methodology” made it difficult to approach creative problem solving. Based on this finding, the influences of 9 competences of computational thinking on software developers were analyzed after selecting 111 software developers with over 10 years of experience in the IT industry. As a result, it was revealed that abstraction was an important competence. The study finding demonstrates that software developers recognized design phase as the most important thing and abstraction of CT is the basics of other competences.

Keywords : Computational Thinking, Abstraction, Software Developer, Design Phase

† 정 회 원: 성균관대학교 교과교육학과 컴퓨터교육전공 박사수료
†† 종신회원: 성균관대학교 교과교육학과 교수(교신저자)
논문접수: 2016년 9월 2일, 심사완료: 2016년 9월 27일, 게재확정: 2016년 9월 28일

1. 서론

전 세계적으로 소프트웨어는 산업, 과학, 사회 전반에 걸쳐 모든 것을 변화시키고 있으며 제품과 서비스의 가치를 결정하는 핵심요인이 되었고, 나아가 기업뿐만 아니라 국가 경쟁력의 원천으로 부상하게 되었다[1]. 또한 소프트웨어는 여러 산업 분야에 '소프트파워'를 통한 공장과 제품의 '지능화'라고 정의되는 4차 산업혁명의 핵심기재로 활용 되고 있으며 우리나라에서는 이러한 세계적인 흐름에 따라 국가 정책적으로 소프트웨어가 기업 혁신과 가치창출이 중심이 되는 '소프트웨어 중심사회'를 만들기 위한 정책을 추진 중에 있다[1][2].

소프트웨어는 스마트기기, 자동차 등 제품의 고도화 뿐 만 아니라 인공지능, 빅데이터, 클라우드 컴퓨팅 등 소프트웨어를 통한 서비스 제공 추세로 진행되고 있으며, 전통적인 하드웨어 중심의 IT 영역의 성장률은 점차 하락하는 반면 소프트웨어 중심의 클라우드, 빅데이터 등의 시장은 20% 이상 고성장 추세로서 소프트웨어 기업들의 사업 영역과 비즈니스 모델은 다변화 되고 있다.

이러한 비즈니스의 대표주자인 Google, Facebook 등이 제공하는 서비스들은 H/W에 종속되지 않고 인터넷을 통해서 제공되는 소프트웨어 서비스로 이해할 수 있으며, 소프트웨어를 통한 인터넷 서비스 및 디지털 콘텐츠의 서비스 비중은 점차 증가할 전망이다. 이러한 변화에 따라 국내 소프트웨어 산업도 전통적인 하드웨어 중심의 SI, 패키지 중심에서 벗어나 새로운 소프트웨어 서비스 영역으로 전환하기 위하여 다각적인 노력을 하고 있다. 그리고 최근에는 소프트웨어 중심의 인공지능, 빅데이터, 클라우드 서비스, 정보보호 뿐만 아니라 제조업과의 융합 등 새로운 분야로 소프트웨어 산업이 확대되고 있는 추세이다[1][2][3][4].

이러한 동향에 발맞추기 위해서는 창의적이고 미래지향적인 소프트웨어 개발자들이 반드시 필요하다[5][6][7]. 그러나 현재 우리나라의 소프트웨어 개발자들이 이러한 흐름에 쉽게 적응하지 못하고 있다고 보여 진다. 그 이유는 4차 산업혁명으로 대표되는 IoT융합분야, 인공지능, 빅데이타와 같은 신산업분야로의 유입 및 업무전환과 적용이 쉽지 않아 보이기 때문이다[6][7]. 그래서 신산업으로 전환을 하기 위해서는 여러 가지 요소(정부정책, 기업투자 등)가 필요하지만 특히 소프트웨어 개발자들이 기존 시스템을 새로운 분야에 적용시킬 수 있는 능력이 뒷받침되어야 한다[3][6][7].

이를 위해 가장 필요한 역량중의 하나는 바로 소프트웨어 개발 전문가들의 창의적 문제해결 능력이다[4][6]. 그러나 우리나라의 소프트웨어 교육 및 산업구조로 인해 상당수의 소프트웨어 개발자들이 창의적 문제해결 역량을 습득하지 못해서 이를 제대로 활용하지 못하고 있다[6][7]. 따라서 소프트웨어 개발자들이 창의적 문제해결 역량을 획득하고 신사업분야에 성공적으로 진입하기 위해서는 제도적으로 또는 스스로 개인의 역량을 개발할 필요가 있다[6][7][8].

그러나 최근까지 우리나라에서는 제도권 교육과정을 마친 개발자들을 상대로 하여 진행되는 교육은 일반적인 업무수행능력향상을 위한 프로그래밍 및 개발방법론 위주의 교육에 치우쳐져 있기 때문에 새로운 창의적 사고가 필요한 신산업분야로의 전환이 쉽지 않다. 따라서 전세계적인 트렌드에 맞는 소프트웨어 중심의 신산업분야로의 전환을 위해서는 창의적인 문제해결 능력이 필요하고, 이를 위해서는 창의적으로 문제 해결하는 방법을 습득하는 것이 매우 중요하다[3][6][7].

그렇다면 지금까지 소프트웨어 개발자들은 어떠한 업무를 했으며, 이들에게 필요한 창의적인 문제해결 방법은 무엇인가? 우리나라의 SI분야 개발자들은 프로젝트 수행을 위해 일반적으로 널리 알려진 "소프트웨어 개발 표준 프로세스"를 업무 특성에 맞게 테일러링하여 업무에 적용한다[9][10]. 그러나 기본적인 업무흐름은 유사하기에 이로 인한 정형화된 개발 프로세스가 개발자들로 하여금 획일적인 사고를 유발함으로서 창의적 문제해결을 방해할 수 있다. 따라서 이를 극복하기 위해서 유연하고 지속가능한 사고의 전환이 필요하다. 최근에 Wing은 모든 영역에서 필요한 창의적 사고체계로 컴퓨팅사고력(Computational Thinking, 이하 CT)을 주장하였고 CT를 확대하여 여러 분야에 제안하였다[11][12]. 또한 다른 연구자들은 학생들을 대상으로 실제 교육장면에 이

를 적용하고 그 효과성을 입증하는 다수의 연구들을 진행하였다[13][14][15]. 만약 CT가 Wing의 주장대로 기본적인 사고체계라면 우리나라 소프트웨어 개발자들에게도 이를 적용할 수 있을 것이다. 그러나 우리나라의 소프트웨어 개발자들은 현재까지 CT에 대한 인식이 부족하고 이에 대한 필요성도 자각하지 못하고 있는 실정이다. 따라서 우리나라 소프트웨어 개발환경에서 실제 CT가 어떻게 인식되는지를 알아볼 필요가 있다. 요약하면 본 연구는 우리나라 소프트웨어 개발자들이 신산업분야로 성공적인 전환을 하기 위해서는 현재 개발자들이 수행하고 있는 업무에 대한 이해가 필요하며, 창의적 사고체계에 대한 필요성을 인식하고 있는가를 탐색할 목적으로 진행되었다. 이를 위해 먼저 소프트웨어 개발자란 무엇이고, 실제 우리나라에서 수행하는 "표준 개발방법론"에 대해서 기술할 것이다. 그리고 창의적인 사고체계로 CT에 대한 정의와 어떤 요소들로 구분되어 있는가를 살펴볼 것이다.

2. 이론적 배경

2.1 소프트웨어 개발전문가

일반적으로 소프트웨어 개발자는 소프트웨어 개발업무에 종사하는 사람으로 정의할 수 있다. 기존 연구에 따르면 어느 분야에서든 전문가가 되기 위해서는 1만 시간의 연습이 필요하며 이는 일반적으로 10년 정도를 연습한 것과 같다고 알려져 있다[16]. 그러므로 소프트웨어 개발전문가는 소프트웨어 개발자로서 10년 이상의 업무 수행 경력을 가진 사람으로 정의할 수 있다.

우리나라에서는 일반적으로 소프트웨어 개발자라는 용어는 사용하고 있으나 이에 대한 명시적인 정의는 없는 실정이다. 그 이유는 프로그래머와 소프트웨어개발자와 같이 직무로 개발자를 구분하는 외국과 달리 우리나라에서는 2008년 개정된 『소프트웨어산업 진흥법 시행령』에 따라 소프트웨어 개발자를 초급·중급·고급·특급 기술자와 같이 기술 등급으로 구분 하여 사용하였다[17][18]. 그리고 이 등급을 기준으로 매년 등급별 노임단가표를 발표하여 왔다[19]. 그러나 이 기준은 학력, 경력, 자격증 유무에 따라 실제업무 수행내역과는 무관하게 단순구분을 하였기에 역량과 등급이 매칭 되지 않는다는 점과 신기술 분야의 경우 실제 특급기술자 수준임에도 초급기술자로 구분된다는 문제점들이 지적되어 왔다.

이러한 지적에 따라 등급별 기준은 2012년 말 폐지되었으나 개발자의 수준을 평가하는 다른 기준이 존재하지 않아 현재까지도 대부분의 발주기관에서 이 등급기준을 준용하여 사용 중에 있다[20]. 정부에서는 등급기준을 보완하기 위해 현재는 국가직무능력표준 (National Competency Standards, NCS)을 바탕으로 한 새로운 체계를 만들기 위해 준비 중에 있다[21]. NCS에서는 <표 1>과 같이 직무경험 10년 이상의 개발자를 총괄아키텍트, 책임SW아키텍트, 응용SW통합개발자, 특급임베디드SW개발자 등으로 규정하고 있다. 또한 이러한 직무능력을 지니고 있는 소프트웨어 개발전문가들은 소프트웨어의 분석, 설계와 관련된 업무들을 수행하며, 어플리케이션설계, 요구사항분석, 개발방법론 선정 및 테일러링 등을 주요 직무수행능력으로 규정하고 있다[22].

<표 1> 산업현장 직무능력수준

	SW 아키텍처	응용SW 엔지니어링	임베디드 SW 엔지니어링
VII(직무경험: 15~20년)	총괄 아키텍트		
VI(직무경험: 10~14년)	책임 SW아키텍트	응용SW 통합개발자	특급 임베디드SW개발자
V(직무경험: 6~9년)	선임 SW아키텍트	응용SW 분석개발자	고급 임베디드SW개발자
IV(직무경험: 3~6년)	SW 아키텍처 설계 담당자	응용SW 실무개발자	중급 임베디드SW개발자
III(직무경험: 1~2년)	SW 아키텍처 구현 담당자	패키징 실무자	초급 임베디드SW개발자
II(직무경험: 0~1년)		UI개발자	

※ 국가직무능력표준 (National Competency Standards, NCS) 일부 재구성

그러나 NCS에서 제시하는 소프트웨어 전문가가 실제 어플리케이션설계, 요구사항분석, 개발방법론 선정 및 테일러링 등의 직무수행능력을 중요시한다고 해도 우리나라에서 진행되는 SI분야의 소프트웨어 개발 업무는 프로젝트 특성별로 테일러링 되어진 개발 방법론 표준프로세스에 따라 관리되거나 법으로 규정된 정보시스템 감리를 위해 단계마다 일정한 산출물들을 작성하고 관리하여야 한다. 그리고 이러한 업무가 개발전문가에게 요구되기에 단계마다 개발 업무의 경중을 분석하는 것은 쉽지 않다. 실제 소프트웨어 개발자들에게 요구되는 개발 방법론 표준 프로세스는 무엇이 있는가를 다음 절에 제시하였다.

2.2 개발방법론 표준 프로세스

소프트웨어 개발 방법론은 소프트웨어 위기 의식, 소프트웨어 이용범위의 확대, 소프트웨어 프로젝트의 대형화에 따라 출현하게 되었다. 소프트웨어 개발방법론은 초기 구조적 방법론부터 시작하여 이후 정보공학 방법론, 객체지향 방법론, CBD방법론 등을 거쳐 최근의 애자일 방법론, 린 스타트업 방법론까지 발전하게 되었다.

그러나 이러한 소프트웨어 개발방법론의 발전에도 불구하고 우리나라의 SI분야는 아직까지도 예전에 사용하던 CBD 개발방법론을 테일러링하여 일반적으로 사용하고 있다. 그러나 이 방법론은 무겁고 형식적인 면이 많아 시장의 변화에 적기에 대응이 어렵다는 지적이 있다. 이러한 문제점에도 불구하고 개발과정에서 단계별 감리를 받아야 하는 제도적인 측면과 프로젝트 관리의 용이성으로 인해 이 방법론은 계속 적용되고 있다.

그 결과 개발프로젝트에 참여하는 소프트웨어 개발자들은 이러한 테일러링된 개발방법론을 준수해서 개발하는 것이 일상이 되었다[23]. 각 기관별로 개발방법론을 명시하고 있으며 종류를 보면 SWEBOK, ISO12207, 정보시스템 감리의 기준이 되는 한국정보화진흥원의 CBD SW표준 산출물 가이드, 국방CBD 방법론, 특허청의 개발 방법론 표준 프로세스 등이 있다. 이러한 각각의 방법론들에 대해 소프트웨어개발 분야 20년 이상의 전문가 6명을 대상으로 검토한 결과 대다수의 전문가들이 특허청의 개발방법론 표준 프로세스 문서가 개발방법론 적용에 가장 적합하다는 의견에 동의하였으며 이 방법론을 <표 2>에 제시하였다.

<표 2> 특허청 개발방법론 표준 프로세스

단계 (Phase)	활동 (Activity)	작업(Task)
개발준비	TFT구성 및 테일러링	사업TFT구성, 방법론테일러링
	개발사전준비	정보화개발준비
분석	요구사항분석	요구사항수집, 요구사항정의, 유스케이스 기술, 요구사항추적
	업무/데이타 분석	업무 분석, 데이터 분석
	아키텍처분석	현행아키텍처 분석
	분석단계테스트계획	총괄테스트 계획
	분석단계점검	분석단계산출물점검
설계	아키텍처설계	SW아키텍처설계,시스템아키텍처 설계
	어플리케이션설계	클래스설계,사용자인터페이스설계,컴포넌트설계,인터페이스설계,배치프로그램 설계, 사용자 웹 구성설계
	DB설계	개념DB모델설계,논리DB모델설계,물리DB모델설계,데이터흐름도작성,데이터검증식작성
	데이터전환설계	데이터전환/검증계획,데이터정비계획
	설계단계테스트계획	단위테스트시나리오작성,통합테스트시나리오작성,시스템테스트시나리오작성,사용자테스트시나리오작성
	설계단계점검	설계단계산출물점검
구현	구현준비	개발환경준비
	개발	프로그램개발
	단위테스트	단위테스트
	구현단계점검	웹표준점검,소스품질검사,구현단계산출물점검
시험	테스트	테스트준비작업,통합테스트,사용자테스트
	시험단계점검	시험단계산출물점검
전개	리허설	리허설준비작업, 최종점검 및 리허설
	전개	전개준비작업,최종점검 및 전개
인도	인수인계	인수인계계획,EA현행화,매뉴얼작성,산출물현행화,산출물인수인계
	교육	교육준비 및 교육

특허청이 제시한 개발방법론 표준 프로세스는 프로젝트별 테일러링에 따라 약간씩 다른 부분도 존재하나 7단계, 23개의 활동, 52개의 작업 그리고 74개의 산출물로 구성되어 있다[10]. 이 문서는 다른 문서들과 달리 각각의 세부 항목들에 대해 구체적인 입출력 산출물을 명시하고 각각의 예시를 제공하고 있다는 장점을 가지고 있다. 개발 업무에서 개발자들이 작성하는 산출물은 프로젝트의 수행 근거로서 실제 매출과 직결되는 것이기에 산출물 작성은 매우 중요하다.

전문적인 개발자들은 자신들의 경험에 따라 업무에 대한 필요성을 다르게 구분할 수 있을 것이다. 즉 초급 개발자들은 구현에 특화되어 그 부분을 중요시 할 수 있으며, 개발 전문가들은 분석, 설계 혹은 개발프로젝트 진행에 필요한 제반 업무를 더욱 중요시하고 그 필요성을 더욱 인식할 수 있을 것이다. 그러나 현재까지 개발프로세스와 관련에서 전문가의 역량 혹은 경력을 바탕으로 그 필요성을 어떻게 지각하고 있는지를 밝힌 연구는 거의 없다. 이를 위해 3장에 제시한 연구(1)에서는 소프트웨어 전문가들의 경험이 실제 개발 방법론 표준 프로세스의 필요성을 어떻게 지각하고 있는가를 탐색하였다. 구체적인 설명은 3장 연구(1) 부분에 기술하였다.

2.3 컴퓨팅사고력

Wing이 모든 사람들이 읽기, 쓰기, 계산하기와 같이 21세기에는 CT가 기본적인 스킬로 사용되고, Computing과 Computer가 CT를 확장시킬 수 있을 것이다[12]. 라고 주장한 이후 CT관련하여 많은 연구가 진행되었다. 이러한 CT는 과학이나 공학 분야의 연구 및 교육 분야 등에서 다양하게 연구되고 적용이 되어왔다. 다시 말해 CT는 복잡하고 비구조화 된 문제들을 새로운 관점에서 접근하여 문제들을 창의적으로 해결할 수 있는 능력을 향상시킬 수 있는 사고체계라고 할 수 있다[12][24].

이것이 가능한 이유는 CT가 다음의 네 가지의 문제해결과정을 주장하고 있기 때문이다. 첫째, 문제 해결을 돕기 위해 컴퓨터 또는 다른 도구를 사용할 수 있도록 문제를 정형화 하기. 둘째, 자료를 논리적으로 분석하고 모델링, 시뮬레이션과 같은 추상화를 통해 자료를 표현하기. 셋째, 알고리즘적 사고를 기초로 해결점을 자동화하고 최적 해결책을 확인, 분석, 구현하기. 넷째, 위 세 가지의 과정을 다양한 문제로 일반화하고 전이하는 과정을 강조한다. 예를 들어, 현재 CT는 미국의 K-12학년과정과 AP(Advanced Placement)등에 적용하고 있으며, 대학과정인 컴퓨터과학에도 적용을 계획 중이다. 교육분야 외에도 머신러닝, 뇌과학, 생명공학 등과 같은 연구분야에도 CT를 적용하고 있으며 화학, 전자, 자동차, 항공, 전자상거래와 같은 산업분야에도 적용하고 있다[11]. 즉, CT는 일반적 문제해결을 위해 필요한 필수 역량이이라고 할 수 있다.

그러나 CT의 역량 및 요소를 구분하는 기준은 연구기관이나 학자에 따라 조금씩 다르게 구분되고 있다. 대표적으로 크게 4개의 그룹으로 나눌 수 있다. 먼저 Wing이 주장하는 추상화, 자동화로 구분을 하는 그룹[11][12], 그 다음으로 미국의 CSTA, 한국의 과학창의재단 등에서 구분하는 자료수집, 문제분석, 자료표현, 문제분해, 추상화, 알고리즘 및 절차, 자동화, 시뮬레이션, 병렬화 등의 9가지 역량으로 구분하고 있는 그룹[14][24][25], 세 번째로 Selby와 Wollard의 알고리즘적 사고, 분해, 일반화(패턴), 추상화, 평가로 구분하는 그룹[13], 마지막으로 KERIS의 자료수집, 자료분석, 구조화, 추상화(분해,모델링,알고리즘), 자동화(코딩, 시뮬레이션), 일반화로 구분하는 그룹 등이 있다[26].

<표 3>과 같이 연구자들 마다 CT를 조금씩 다르게 구분하고 있으나 추상화의 경우 모든 연구자들이 CT에서 필요로 하는 중요한 개념이라고 주장하고 있다[11][14][13][26]. 왜냐하면 추상화란 인간이 문제해결을 위해 사고하는 과정전반을 의미하고, 문제 해결에 필요한 핵심 요소를 선정하고 복잡함을 줄이는 단계로 정의하고 있기 때문이다[12][25]. 즉 추상화는 CT의 가장 중요한 CT의 역량이다.

요약하면 CT는 컴퓨팅 시스템의 역량을 활용하여 해결하고자 하는 문제를 효과적이고 효율적으로 해결할 수 있는 절차적 사고 능력으로 여러 개의 역량들로 구분할 수 있으며, 이러한 여러

가지 역량 중에서 특히 추상화 능력이 창의적인 사고에 가장 필요하다고 할 수 있다. 그러나 앞에서 언급했듯이 여러 연구자들이 CT의 중요성을 인식하고 있고, 특히, 컴퓨터나 도구사용을 위해 문제를 정형화 하는 일련의 과정이 있음에도 불구하고 실제 소프트웨어 개발자들이 CT에 대한 인식과 필요성을 자각하고 있는가에 대한 명확한 증거는 없다. CT의 개념이나 정의를 보면 모두가 소프트웨어 개발자들에게 필요한 것이라는 사후설명일 뿐이다. 전문 소프트웨어 개발자들에게 CT의 중요성을 평가하도록 하여 증거를 획득할 필요가 있다. 이를 위해 다음 4장에 제시한 연구(2)가 수행되었다.

<표 3> 연구자별 CT의 학습단계 및 요소와 상호 관련성[27]

<table>
<tr><th>Wing(2008)</th><th>CSTA & ISTE (2011)</th><th>Google for Education (2015)</th></tr>
<tr><td rowspan="9">추상화</td><td colspan="2">자료수집 (Data Collection)</td></tr>
<tr><td rowspan="2">자료분석 (Data Analysis)</td><td>자료분석 (Data Analysis)</td></tr>
<tr><td>패턴인식 (Pattern Recognition)</td></tr>
<tr><td colspan="2">자료제시(Data Representation)</td></tr>
<tr><td>문제분해 (Problem Decomposition)</td><td>분해(Decomposition)</td></tr>
<tr><td rowspan="2">추상화 (Abstration)</td><td>추상화 (Abstration)</td></tr>
<tr><td>패턴일반화 (Pattern Generalization)</td></tr>
<tr><td rowspan="4">자동화</td><td>알고리즘 및 절차 (Algorithm and Procedures)</td><td>알고리즘 디자인(Algorithm Design)</td></tr>
<tr><td colspan="2">자동화 (Automation)</td></tr>
<tr><td colspan="2">병렬화 (Parallelization)</td></tr>
<tr><td colspan="2">시뮬레이션 (Simulation)</td></tr>
</table>

※ 김석전, 전용주, 김태영 (2016)의 연구자별 CT의 학습단계 및 요소와 상호관련성을 인용함

3. 연구(1)

연구(1)은 소프트웨어 개발전문가들이 소프트웨어 개발업무의 필요성을 얼마나 자각하는지를 탐색하기 위하여 설문조사를 실시하였다. 구체적인 방법은 다음 절에 제시하였다.

3.1 연구방법

정보기술개발분야의 개발프로젝트 참여경험이 있는 소프트웨어개발전문가 300명을 대상으로 하여 설문을 진행하였으며 <표 2>에 제시한 특허청의 개발방법론 표준프로세스의 내용을 설문에 맞게 일부항목을 변형하여 소프트웨어 개발과정에서 필요한 각 단계별 필요성을 조사하였다. 연구에서 사용된 설문도구는 다음과 같이 구성하였다. 표준프로세스 중 개발준비단계에서 요구되는 과업인 사업TFT구성의 필요성을 측정하기 위해 리커드 6점 척도로 구성된 문항을 아래와 같이 구성하였다.

예시1) "귀하가 생각하시기에 SW개발자는 [사업 수행 시 발생하는 주요이슈사항을 신속하게 처리하기 위해 사업과 관련 있는 부서를 중심으로 사업TFT를 구성]하는 [사업TFT구성]작업이 얼마나 필요하다고 생각하십니까?"

예시2) "귀하가 생각하시기에 SW개발자들은 [개발 시스템의 컴포넌트, 사용자 인터페이스 기능 점검을 위한 테스트대상을 분석하고 테스트케이스를 작성]하는 [단위 테스트 케이스작성]작업이 얼마나 필요하다고 생각하십니까?"

위와 같은 문항에 참가자들은 전혀 필요하지 않다(1), 필요하지 않다(2), 조금 필요하지 않다(3), 약간 필요하다(4), 필요하다(5), 매우 필요하다(6)의 6개 리커드 척도에 반응하도록 하였다. 짝수 척도를 사용한 이유는 필요성에 대한 질문이기에 홀수 척도에서 흔히 나타나는 중간 척도의 반응을 피하기 위해서였다.

<표 2>의 표준 개발 프로세스에 제시되어 있는 Task(과업) 항목을 위와 같은 형태로 구성하였고 총 52개의 문항이 연구에 사용되었다. 그리고 기본적인 인구통계학적 변인들을 추가로 구성하였다.

3.2 연구결과

조사대상자의 일반적인 특성은 <표 4>와 같다.

<표 4> 조사대상자의 일반적 특성

구분	항목	빈도	%
실무 경력	6-9년	100	33.3%
	10-14년	100	33.3%
	15년 이상	100	33.3%
	합 계	300	100%
학력	전문학사 이하	35	11.6%
	학사	216	72%
	석사	38	12.7%
	박사	9	3%
	기타	2	.7%
	합 계	300	100%
자격증	기술사	18	6%
	기사	178	59.3%
	산업기사	30	10%
	기타 및 없음	74	24.7%
	합 계	300	100%

참가자들의 IT업무 종사경력과 개발프로젝트 참여경력, SW프로그램 개발경력에 대한 평균 경력 년수가 <표 5>에 제시되어 있다. 참가자들은 세 영역 모두 평균 11년 이상 종사하였다. 또한 소프트웨어 개발 전문가가 되기 위해 필요한 실무경력에 대해서는 8.79년 약 9년 이상 실무 경력이 필요하다고 응답하였다.

<표 5> 조사대상자의 경력 및 전문가에게 필요한 실무경력(단위 년, n=300)

	IT 업무 종사 경력	개발 프로젝트 참여경력	SW 프로그램 개발경력	전문가에 게필요한 실무경력
평균	12.65	11.16	11.19	8.79
표준 편차	5.12	5.13	5.82	4.61

<표 6>에 소프트웨어 개발방법론 프로세스간 상관과 평균이 제시되어 있다. 각 프로세스간 변수들의 상관이 모두 높으며 통계적으로 유의한 결과가 나타났다($rs > .63$, $p < .001$). 그리고 소프트웨어 개발자들은 일곱 개의 단계중에서 “설계” 단계를 소프트웨어 개발에 가장 중요하다고 응답하였다(52.7%).

<표 6> 소프트웨어 개발방법론 프로세스간 상관, 평균, 척도의 신뢰도, 단계의 중요성 비율

단계	1	2	3	4	5	6	7
1. 개발 준비	1.00						
2. 분석	.64	1.00					
3. 설계	.66	.82	1.00				
4. 구현	.69	.86	.84	1.00			
5. 시험	.65	.74	.70	.80	1.00		
6. 전개	.63	.72	.70	.77	.82	1.00	
7. 인도	.74	.74	.74	.80	.77	.78	1.00
평균	4.52	5.02	4.83	4.98	4.96	5.00	4.84
표준편차	.85	.60	.62	.58	.70	.67	.70
Cronbach 알파	.68	.84	.93	.73	.80	.72	.83
가장 중요한 단계	2.3 %	21.3 %	52.7 %	10.3 %	10.0 %	0.7 %	1.0 %

IT종사경력, 개발프로젝트 참여경력에 따라 표준 프로세스 7단계 필요성을 어떻게 생각하는가를 탐색하기 위해 회귀분석을 실시하였다.

회귀분석결과를 요약하면 다음 <표 7>과 같다. 첫째 IT종사년도가 증가할수록 7개의 표준프로세스 중 전개를 제외한 나머지 6개(개발준비, 분석, 설계, 구현, 시험, 인도) 단계에서 통계적으로 유의미하였다. 둘째, 프로젝트 참여경력이 증가할수록 분석을 제외한 나머지 6개(개발준비, 설계, 구현, 시험, 전개, 인도) 단계에서 통계적으로 유의미하였다. 그러나 <표 7>의 결과를 단순히 받아들이면, IT종사경력과 프로젝트 참여경력이 증가할수록 필요성에 대한 인식을 중요시한다고 설명할 수 있으나 회귀 방정식의 계수가 약 .1수준이었고, 설명량(R^2=.01 이하)도 매우 작아 통계 결과로만 해석해서 받아들이기 힘들다. 이를 방증하는 결과로 SW프로그램 개발경력을 6년에서 9년, 10년에서 14년, 15년 이상으로 구분한 후 각 단계의 필요성에 대해 차이검증을 하였을 경우에 통계적으로 의미 있는 결과가 나타나지 않았다. 즉 IT종사경력, 프로젝트 참여경력에 따라 각 단계의 필요성에 대한 차이를 구분하지 못한다고 할 수 있다.

<표 7> IT,프로젝트 경력과 프로세스간 단순회귀

IV	DV	계수	R2	STE	t	p
IT 종사 경력	개발 준비	0.13	0.02	0.01	2.31	0.02
	분석	0.10	0.01	0.01	1.74	0.08
	설계	0.10	0.01	0.01	1.65	0.10
	구현	0.11	0.01	0.01	1.97	0.05
	시험	0.12	0.02	0.01	2.10	0.04
	전개	0.09	0.01	0.01	1.57	0.12
	인도	0.13	0.02	0.01	2.26	0.02
프로젝트 참여 경력	개발 준비	0.16	0.02	0.01	2.71	0.01
	분석	0.09	0.00	0.01	1.53	0.13
	설계	0.12	0.01	0.01	2.03	0.04
	구현	0.11	0.01	0.01	1.82	0.07
	시험	0.12	0.01	0.01	2.16	0.03
	전개	0.10	0.01	0.01	1.76	0.08
	인도	0.12	0.02	0.01	2.11	0.04

IV=독립변인, DV=종속변인, R2=설명량, STE=표준오차

연구(1)을 요약하면 소프트웨어 개발자들은 표준개발 프로세스에 대한 필요성을 모두 다 중요하다고 인식(평균 4.5이상)하고 있었다.

특히 흥미로운 것은 개발 프로세스 중 분석, 설계 단계를 매우 중요하다고 생각했으며 이중 설계를 가장 중요하게 판단한 것이다. 필요성에 대한 인식은 동일한데 왜 설계를 가장 중요하다고 결정했는가?

이는 소프트웨어의 비가시성(Invisibility)이 가장 중요한 이유로 소프트웨어 완제품의 구조가 개발된 코드 안에 숨어 있어 파악하기 힘들기 때문이다. 물리적인 형태가 없는 무형의 논리적인 요소로 구성이 되어 있기에 최종산출물이 개발 마지막 단계에 이르러서야 그 실체에 대해 확인이 가능하다.

그러므로 성공적인 소프트웨어 개발을 위해서는 주어진 문제를 개발자들이 이해할 수 있는 형태로 만드는 설계단계가 중요하고 할 수 있다. 그리고 소프트웨어공학에서는 설계단계의 가장 기본 원리중의 하나를 추상화로 보고 있다[26][27][29].

정리하면 주어진 문제해결을 위해서 소프트웨어 개발자들은 설계단계를 가장 중요한 단계로 인식하고 있고 그 단계에서 요구되는 가장 핵심적인 기본원리는 추상화이다. 그러므로 추상화를 소프트웨어 개발자들이 어떻게 인식하고 있는가를 탐색할 필요가 있다. 왜냐하면 앞에서 언급한 바와 같이 창의적인 문제해결을 위해서 CT가 필요하고, 그 중에서 특히 추상화 역량이 가장 중요하기 때문이다. 이를 위해 연구(2)가 수행되었으며 다음 장에 제시하였다.

4. 연구(2)

연구(2)는 소프트웨어 개발전문가들이 CT의 필요성을 얼마나 자각하는지를 탐색하기 위하여 설문조사를 실시하였다. 구체적인 방법은 아래와 같다.

4.1 조사대상자의 일반적 특성

정보기술개발분야에 종사하고 있는 만 37세 이상에 해당하는 전국의 150명을 대상으로 하여 진행하였으며 소프트웨어 개발과정 중에서 분석 및 설계 단계에 필요한 CT역량을 확인하기 위해 소프트웨어 기술 분야의 실무경력 10년 이상의 소프트웨어 개발전문가를 기준으로 하였다.

<표 8> 조사대상자의 일반적 특성

구분	항목	빈도	%
실무 경력	10-14년	31	28%
	15-19년	57	51%
	20-33년	23	21%
	합 계	111	100%
학력	전문학사	21	19%
	학사	65	58%
	석사	22	20%
	박사	3	3%
	합 계	111	100%
자격증	기술사	8	7%
	기사	43	39%
	산업기사	8	7%
	기타 및 없음	52	47%
	합 계	111	100%

이에 전체응답 150건 중 ① 학위가 없는 경우, ② IT경력 10년 미만, ③ 소프트웨어개발 프로젝트 참여경험이 없는 경우, ④ 소프트웨어 개발경험이 없는 경우에 해당하는 39건을 제외한 111건을 대상으로 설문결과를 분석하였다. 수집된 설문의 인구통계학적 분석은 <표 8>와 같다.

4.2 연구결과

연구결과는 크게 세 단계로 구분하여 진행하였다. 첫째 IT종사년도에 따라 CT의 필요성에 대해 차이가 나는지 알아 보기 위해 일원변량분석을 실시하였다. 둘째 소프트웨어 개발자들이 중요CT역량과 성공적인SW개발을 위해 필요한 CT역량에 대한 빈도분석을 실시하였다. 마지막으로 추상화역량이 다른 CT역량에 얼마나 필요한지를 알아보기 위해 내용타당도비율(Content Validity Ratio, CVR)을 실시하여 내용타당도를 검증하였다[15][28][30].

소프트웨어 개발전문가가 소프트웨어 개발단계에 있어서 CT의 필요성이 IT종사 기간별로 차이가 있는지를 검증하기 위하여 일원변량분석을 실시하였다. 분석결과는 아래의 <표 9>와 같다. 분석은 IT종사기간 별로 3개의 집단으로 나누었기 때문에 일원변량분석을 실시하였으며 사후검증(Post Hoc)을 실시하였다.

첫째, 문제를 해결하기 위한 '자료수집역량'이라는 항목에 있어서 집단 간에 유의미한 차이가 나타나($F_{(2,108)}$=3.573, P<.05), 사후검증으로 Scheffe을 실시하였다. 그 결과 경력 10년에서 14년까지의 개발자들과 20년 이상 33년 이하의 개발자들 사이에서 그 차이가 유의미한 것으로 나타났다.

둘째, 소프트웨어 개발자들은 수집된 자료와 문제에 대해 주어진 자료를 분류하고 분석하는 '자료분석 역량'에 있어서 경력 집단 간 유의미한 차이가 나타났다($F_{(2,108)}$=4.249, P<.05). 사후검증으로 Scheffe검증을 실시한 결과 경력 15년 이상 19년 이하의 경력을 가진 개발자 집단과 20년부터 33년까지의 경력을 가진 개발자들 사이에서 유의미한 차이가 나타났다.

셋째, 개발자들은 문제에 대해 다양한 수단을 활용하여 표현할 수 있는 '자료표현역량'에 대해서도 경력집단별로 다르게 나타났는데($F_{(2,108)}$=2.766, P<.05), 사후검증으로 Scheffe검증을 실시한 결과 경력 10년부터 14년까지의 경력을 가진 개발자 집단과 20년부터 33년까지의 경력을 가진 개발자들 사이에서 차이가 나타났다.

<표 9> IT종사기간 별 CT의 필요성

컴퓨팅 사고력	경력 집단	평균	표준 편차	F	유의도	사후검사
1. [자료수집역량]의 필요성	1	3.71	.106	3.573*	.031	1<3*
	2	3.88	.097			
	3	4.22	.153			
2. [자료분석역량]의 필요성	1	4.06	.113	4.249*	.017	2<3*
	2	3.96	.100			
	3	4.48	.152			
3. [자료표현역량]의 필요성	1	3.55	.138	2.766	.067	1<3*
	2	3.75	.101			
	3	4.04	.160			
4. [문제분해역량]의 필요성	1	4.06	.092	1.716	.185	N.S
	2	4.04	.103			
	3	4.35	.149			
5. [추상화역량]의 필요성	1	3.81	.117	.940	.394	N.S
	2	3.81	.101			
	3	4.04	.160			

6. [알고리즘과 절차 역량]의 필요성	1	4.03	.135	.611	.545	N.S
	2	4.16	.103			
	3	4.26	.157			
7. [자동화 역량]의 필요성	1	3.68	.134	1.862	.160	N.S
	2	3.88	.103			
	3	4.09	.165			
8. [시뮬레이션역량]의 필요성	1	3.84	.161	.854	.428	N.S
	2	3.75	.080			
	3	4.00	.189			
9. [병렬화 역량]의 필요성	1	3.65	.109	2.095	.128	N.S
	2	3.75	.095			
	3	4.04	.183			

경력집단 1= 10-14년, 2=15-19년, 3=20-33년

* p<.05, ** p<.01, N.S(유의하지 않음)

또한 참가자들에게 소프트웨어 개발전문가에게 중요한 CT역량을 조사한 결과 <표 10>과 같이 문제분해, 자료분석, 알고리즘 및 절차 역량 순으로 나타났다. 특히 이 3가지 역량은 케이스 퍼센트가 50%가 넘는 항목으로 소프트웨어 개발전문가들이 중요한 CT역량이라고 인식하고 있었다.

한편 성공적인 소프트웨어 개발에 필요한 CT역량을 조사한 결과 <표 11>과 같이 알고리즘 및 절차, 자료분석, 문제분해 역량 순으로 나타났다.

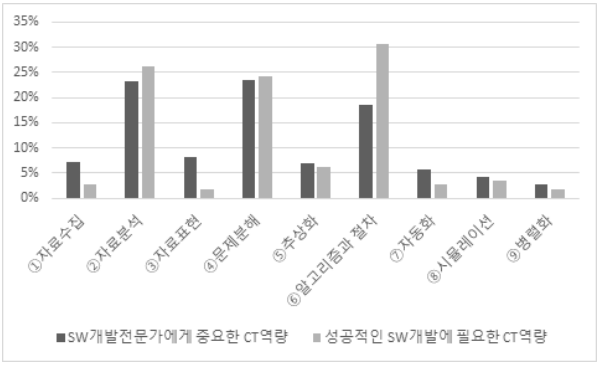
그리고 이 두 개를 비교한 결과는 다음의 [그림 1]과 같다.

<표 10> SW개발전문가에게 중요한 CT역량(3개선택)

CT역량	N	퍼센트	케이스 퍼센트
①자료수집	24	7%	22%
②자료분석	77	23%	69%
③자료표현	27	8%	24%
④문제분해	78	23%	70%
⑤추상화	23	7%	21%
⑥알고리즘과 절차	62	19%	56%
⑦자동화	19	6%	17%
⑧시뮬레이션	14	4%	13%
⑨병렬화	9	3%	8%
합 계	333	100%	300%

<표 11> 성공적인 SW개발에 필요한 CT역량

CT역량	N	비율
①자료수집	3	3%
②자료분석	29	26%
③자료표현	2	2%
④문제분해	27	24%
⑤추상화	7	6%
⑥알고리즘과 절차	34	30%
⑦자동화	3	3%
⑧시뮬레이션	4	4%
⑨병렬화	2	2%
합 계	111	100%



[그림 1] 중요CT역량과 성공적인 SW개발에 필요한 CT역량 비교

소프트웨어 개발전문가들이 기존 선행 연구결과와 달리 중요 CT역량으로 추상화역량을 선택하지는 않았으나 별도 항목으로 조사한 개발과정에 추상화 역량이 필요한지에 대한 질문에 88%가 필요하다고 답하였다. 또한 소프트웨어 개발전문가의 추상화역량 보유 유무 및 습득방법은 <표 12>와 같다. 소프트웨어 개발전문가는 대다수가 추상화역량을 보유하고 있으며, 일반적으로 업무를 통해서 추상화역량을 습득했다는 것을 확인할 수 있다.

<표 12> 추상화역량 보유 유무 및 습득방법

보유유무		추상화 역량 습득 방법		
항목	N	항목	N	합계
보유	85	①학교	3	76.6%
		②학원(사내직무 교육과정제외)	2	
		③사내직무교육	3	
		④업무를 통한 습득	70	
		⑤개인학습(독서, 웹서핑, 논리적 사고 훈련 등)	7	
미보유	26	추상화역량 미보유	26	23.4%
합계			111	

9가지로 구분한 CT역량 중 추상화 역량이 다른 CT역량에 어느 정도 필요한지에 대해 내용타당도비율(Content Validity Ratio, CVR)조사한 결과는 <표 13>과 같다. 자동화역량 및 병렬화 역량을 제외한 나머지 역량에 대해 추상화역량은 해당 역량에 있어서 필요한 기본역량이라는 것을 확인 할 수 있다.

<표 13> CT역량별 추상화 필요성

CT역량	적합수	CVR	결과
①자료수집역량	82	0.48	채택
②자료분석역량	88	0.59	채택
③자료표현역량	79	0.42	채택
④문제분해역량	80	0.44	채택
⑤추상화역량	-	-	-
⑥알고리즘과 절차역량	90	0.62	채택
⑦자동화역량	61	0.10	기각
⑧시뮬레이션역량	74	0.33	채택
⑨병렬화역량	54	-0.03	기각

연구(2)의 결과를 요약하면 첫째 IT종사년도에 따라 CT의 필요성은 자료수집, 자료분석, 자료표현에서 차이가 나타났다. 둘째 소프트웨어 개발자들이 인식하는 중요CT역량과 성공적인SW개발을 위해 필요한 CT역량을 알아본 결과 자료분석, 문제분해, 알고리즘과 절차가 매우 중요하다고 인식하였다. 셋째 추상화역량이 다른 CT역량에 얼마나 필요한지를 내용타당도비율(Content Validity Ratio, CVR)을 실시하여 검증한 결과 자동화, 병렬화 역량을 제외한 나머지 6가지 CT역량에 대해 추상화가 필요하다는 것을 확인하였다.

5. 결론

본 연구에서는 소프트웨어 개발전문가의 업무가 무엇이고 이 업무를 담당하는 소프트웨어 개발전문가들이 CT를 필요로 하는지를 검증하기 위해 실시하였다. 연구결과를 요약하면 다음과 같다. 첫째 IT종사경력 및 개발프로젝트 참여경력이 많을수록 개발단계의 모든 프로세스가 중요하다고 인식했으나 설계단계를 가장 중요한 단계로 인식하였다. 둘째 소프트웨어 개발전문가들은 CT에서 가장 중요하다고 간주되는 추상화역량의 필요성을 인식하고 있었으며 추상화 역량은 다른 6가지 CT역량(자료수집, 자료분석, 자료표현, 문제분해, 추상화, 알고리즘과 절차, 시뮬레이션)들에 대해 영향을 주는 것으로 확인되었다. 또한 IT종사경력에 따라 CT역량의 필요성 중 자료수집, 자료분석, 자료표현에서 중요성에 차이가 나타났다.

본 연구에서 가장 흥미로는 점은 소프트웨어 개발자들의 반응이었다. 소프트웨어 개발자들이 표준개발 프로세스와 CT역량에 필요성 및 중요성을 인식할 때는 대부분 동일하게 반응하였으나 각 개발단계 혹은 각 CT역량을 경쟁시키는 문항, 예를 들어 어떤 단계 또는 역량이 소프트웨어 개발자들에게 중요하다고 생각하느냐는 질문에 대해서는 설계와 추상화를 가장 중요하다고 판단하였다는 것이다. 이를 추정해보면 표준 개발프로세스는 소프트웨어 개발자들 입장에서는 반드시 수행해야 하는 업무이기에 동일하게 필요성을 인식했지만 소프트웨어 개발업무에서 가장 중요한 단

계는 설계이고 설계의 기본원리는 추상화이기에 개발자들이 이와 같이 판단한 것 같다.

CT역량에서도 연구자들이 가장 중요하다고 주장하는 추상화 역량을 다른 역량들과 비교하였을 경우 그 중요성이 발견되지 않았으나, 각 역량에 대한 추상화의 필요성을 알아본 결과 각 역량에 영향을 미치는 기본원리라는 것을 확인 할 수 있었다. 이것은 추상화가 다른 역량들의 수행에 있어 필요한 가장 기본적인 것이라는 연구자들의 주장과 일치한다.

마지막으로 본 연구는 두 가지 시사점을 제공한다. 첫째, 이번 연구를 정리해볼 때 소프트웨어 개발전문가를 양성하는 학교 및 교육기관 등에서는 현재 업무를 통해 습득하고 있는 CT의 가장 중요한 개념인 추상화역량을 학습할 수 있는 학습과정이 필요하다고 할 수 있다. 또한 기존의 소프트웨어 개발전문가의 분석 및 설계능력 강화를 위해서는 KOCW(Korea Open CourseWare)와 같은 다양한 학습방법 제공 등을 통해 추상화역량에 대한 습득방법 제공이 필요하다고 할 수 있다.

둘째 현재까지 CT와 관련된 연구는 다양하다. 그러나 CT의 역량을 어떻게 측정하고 이것을 어떻게 평가하는가에 관한 연구는 거의 드물다. 따라서 CT와 관련하여 보다 더 심도 있게 연구를 수행하기 위해서는 CT를 타당하고 신뢰롭게 측정할 수 있는 도구 개발이 선행될 필요가 있다.

참 고 문 헌

[1] 지은희(2014). 2014 소프트웨어 산업 실태조사 분석 연구. **소프트웨어정책연구소**, 연구보고서.

[2] 김진형(2016). 제4차 산업혁명이 가져올 'SW 중심사회'. **소프트웨어정책연구소**,SPRi칼럼.

[3] 최계영(2013). SW 미래전략. **정보통신정책연구원**, Premuim Report.

[4] 지은희(2015). 국내 소프트웨어 기업 경쟁력에 관한 연구. **소프트웨어정책연구소**, 연구보고서.

[5] 양병석(2015). 소프트웨어 중심사회를 맞은 소프트웨어의 역할 변화. **KT 경제경영 연구소**, 디지에코 보고서.

[6] 류지성(2011). 소프트웨어 우수 인재 양성·확보를 위한 제언. **삼성경제연구소**, CEO Information.

[7] 나성현(2013). SW 인력양성을 위한 정책 제언. **정보통신정책연구원**, KISDI Premium Report.

[8] 정경원(2011). 국내 소프트웨어 산업 발전 전략. **한국통신학회지(정보와통신), 29**(1), 17-22.

[9] 한국정보화진흥원 (2011). **CBD SW개발 표준 산출물 관리 가이드**. 한국정보화진흥원.

[10] 특허청(2014). **특허청SW개발방법론**. 특허청.

[11] Wing J. M. (2012). *Microsoft Research Asia Faculty Summit 2012*. Microsoft Research.

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

[13] 최숙영(2016). 문제해결의 관점에서 컴퓨팅 사고력 증진을 위한 교수학습에 대한 연구. **한국컴퓨터교육학회논문지, 19**(1).

[14] 이영준 외(2014). **초중등 단계 Computational Thinking 도입을 위한 기초 연구**. 한국과학창의재단.

[15] 박정희(2014). **초등교사의 다문화교육역량 척도개발 및 타당화**. 박사학위논문, 이화여자대학교.

[16] Malcolm Gladwell(2008/2009). *Outliers: The Story of Success.* New York: Back Bay Books, 2008/2009, p.44.

[17] UNITED STATES DEPARTMENT OF LABOR (2016). *Computer Programmers* (http://www.bls.gov/oes/current/oes151131.htm), *Software Developers, Applications* (http://www.bls.gov/oes/current/oes151132.htm). Bureau of Labor Statistics.

[18] 소프트웨어산업 진흥법 시행령 (2008). [별표 1] 소프트웨어기술자의 기술등급 및 인정범위(제1조의2 관련). **법제처**.

[19] 2015년도 적용 SW기술자 평균임금 공표. **한국소프트웨어산업협회**(2015).

[20] 창조경제 첨병 SW개발자에 연차-학위 따지

는 '노임단가표'적용 여전. (2015.1.4). **전자신문**.
[21] SW개발자 인건비 산정기준, SW노임등급제에서 NCS로 바뀐다. (2015.5.22). **전자신문**.
[22] 국가직무능력표준(2014). 표준 및 활용 패키지, 소분류:정보기술개발, 세분류(직무):응용SW엔지니어링. **고용노동부**.
[23] 양단희(2014). 소프트웨어 개발방법론의 패러다임 변화. **한국인터넷정보학회논문지, 15**(2).
[24] 권정인(2013). **Computational Thinking 기반의 교수-학습이 학습자의 창의적 문제해결에 미치는 효과성 연구**. 박사학위논문, 성균관대학교.
[25] CSTA(2011). *Computational Thinking teacher resource second edition* (http://csta.acm.org/Curriculum/sub/CompThinking.html). Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE).
[26] 송영재(2004). **소프트웨어공학: 객체지향모델링과 CBD중심**. 이한출판사, 251-256.
[27] 김석전, 전용주, 김태영(2016). Google CT 수업지도안 분석. **한국컴퓨터교육학회 동계학술대회, 20**(1), 67-71.
[28] 최은만(2007). **소프트웨어공학(4차개정판)**. 정익사, 172-179.
[29] 김치수(2015). **쉽게 배우는 소프트웨어공학**. 한빛아카데미.
[30] Lawshe, C. H.(1975). *A quantitative approach to content validity*. Personnel psychology, 28, 563 - 575.

박 성 빈

1998 중앙대학교 졸업(학사)
2007 중앙대학교대학원졸업(석사)
2011～현재 성균관대학교 대학원 컴퓨터교육과 박사과정
관심분야: SW교육, Computational Thinking, Software development
E-Mail: gerbera4@skku.edu

안 성 진

1988 성균관대학교 정보공학과(학사)
1990 성균관대학교 정보공학과(석사)
1998 성균관대학교 정보공학과(박사)
1990～1995 KIST/SERI 연구원
1996 정보통신기술사
1999～현재 성균관대학교 컴퓨터교육과 교수
관심분야: SW교육, 정보윤리, 정보보안
E-Mail: sjahn@skku.edu