제26권 제2호(통권 98호) 2013년 6월 ISSN 1738-6322

소프트웨어공학소사이어티 논문지

Journal of Software Engineering Society

VOLUME 26, NUMBER 2, June 2013

사단법인 한국정보과학회
소프트웨어공학소사이어티

Journal of Software Engineering Society

VOLUME 26, NUMBER 2, June 2013

Korean Institute of Information Scientists and Engineers
Software Engineering Society

제품라인 공학을 위한 휘처 기반의 제품 구성 방법

(A Feature-based Product Configuration Method for Product Line Engineering)

배성진‡ 강교철¶
(Sungjin Bae) (Kyo Chul Kang)

요 약 소프트웨어 제품라인공학은 재사용성에 초점을 맞추어 소프트웨어의 높은 품질과 생산성을 만족시킬 수 있는 방법으로 제안되었다. 소프트웨어 제품라인에서 제품 구성 방법은 휘처모델로부터 주어진 제품을 위해 가장 최선의 휘처와 휘처속성을 선택해 나가는 프로세스이다. 성공적인 제품 개발을 위해서는 제품의 목표를 달성할 수 있는 휘처와 휘처 속성을 선택하는 것이 중요하다. 하지만 수천개의 휘처와 휘처 속성이 존재하는 경우에는 최적의 제품 구성을 하는 것이 매우 어렵다. 그렇기에 본 연구에서는 휘처와 휘처 속성간의 관계를 기반으로 제품의 목표를 달성하게 하는 휘처와 휘처 속성의 구성 조합을 찾는 휘처 구성 방법을 제안하여, 보다 정확한 제품의 목표 달성에 기여하는 휘처 구성이 될 수 있도록 한다.

키워드 소프트웨어 제품라인 공학, 휘처 모델, 휘처 구성 방법, 휘처와 휘처속성 선택

Abstract Software product line (SPL) engineering is a reuse paradigm that helps organizations increase productivity and improve product quality by developing product from reusable core assets. In SPL, product configuration is the process of selecting the desired features and feature attributes for a given product from a feature model. In order to develop a successful product, feature and feature attribute selection that can achieve the product goal is important. There can be thousands of features and feature attributes resulting in myriads of configurations and finding the best configuration efficiently is a hard task. This paper proposes a systematic process for feature-based product configuration. To support development of a product that satisfys all product goals(business goals and quality goals), a model showing how feature and feature attribute combinations are related to product goals is included and a method for deriving an optimal product configuration using the model is proposed.

Key words Software Product Line Engineering, Feature Model, Feature Configuration Method, Feature and Feature Attribute Selection

1. 연구배경

제품라인공학에서 성공적인 제품을 개발하기 위해서는 제품라인의 가변성(Product Line's Variability) 관리가 근본적으로 중요한 요소이다. 특히 휘처 모델링(Feature Modelling)과 휘처 구성(Feature Configuration)에서의 가변성 관리(Variability Management)는 제품라인을 통한 제품 개발의 복잡성(Complexity)에 큰 영향을 준다. 더욱이 산업체에서의 제품라인은 자산에서 제품을 개발하는데 수천가지의 가변점과 그 이상의 가변부를 가지고 있다[1]. 따라서 제품 개발을 위해 이런 가변성을 관리하며 제품의 목표(Goal)에 맞는

† 이 논문은 2014 한국 소프트웨어공학 학술대회에서 '제품라인 공학을 위한 휘처 기반의 제품 구성 방법'의 제목으로 발표된 논문을 확장한 것임
‡ 비 회 원 : 포항공과대학교 정보전자융합공학부
sjbae@postech.ac.kr
¶ 종신회원 : 포항공과대학교 정보전자융합공학부 교수
kck@postech.ac.kr

휘처를 구성하는 것은 아주 복잡하고, 비용이 많이 들어간다[2]. 특히 하드웨어와 소프트웨어의 통합개발에서는 휘처 구성의 복잡성에 의해 개발 초기의 잘못된 하드웨어 휘처를 선택하여 개발 목표를 달성하지 못할 경우, 하드웨어를 재개발해야 하는 문제까지 야기할 수 있어, 엄청난 개발 기간 지연 발생은 물론 제품 개발 자체의 실패로 귀결된다. 이러한 문제점을 해결하기 위해 제품 목표에 맞는 휘처 구성 방법에 대한 연구가 시도되고 있다.

휘처 구성 방법은 제품라인 자산인 휘처 모델이 가지는 가변성을 제거해 나가며, 특정 제품의 목표(Goal)에 부합하는 휘처를 선택하게 하는 것이다. 제품의 목표를 달성하기 위해서는 가격과 같은 비지니스 목표와 성능과 같은 품질 목표가 함께 고려 되어야 한다.

휘처와 휘처 속성(Feature Attribute)의 선택은 제품의 목표에 영향을 준다. 일반적으로 하나의 휘처나 휘처 속성을 선택할 때 동시에 여러 개의 제품 목표에 영향을 줄 수 있는데 영향을 주는 정도는 항상 고정되어 있는 것이 아니다. 그 이유는 휘처나 휘처 속성의 조합(Combination)이 어떻게 구성(Configuration) 되는냐에 따라 각각의 휘처나 휘처 속성마다 제품 목표에 영향을 주는 정도가 달라 질 수 있기 때문이다.

이처럼 휘처 구성이 성공적으로 제품 목표를 달성하기 위해서는 첫째, 제품의 목표에 영향을 주는 휘처와 휘처 속성간의 관계를 분석하고 둘째, 이 관계를 통해 제품의 모든 목표에 부합되는 휘처와 휘처 속성의 구성 조합을 찾는 것이 중요하다. 그러므로 본 연구에서는 이 목표를 달성하는 과정에서 직면하는 문제점과 기존 방법의 한계점을 파악하여 이를 해결 할 수 있는 방법을 제안한다.

기존의 휘처 구성 방법으로는 휘처 선택시 제품 목표에 미치는 영향을 나타내고, 그 목표에 영향을 미치는 기대값을 이용하여 휘처를 구성하는 방법이 존재한다[3][4][5][6]. 하지만, 기존 방법들은 제품 목표에 크게 영향을 줄 수 있는 휘처 속성에 대한 고려가 없고, 휘처 마다 제품 목표에 영향을 주는 기대값을 고정적으로 가지고 있어 휘처 구성 조합에 따라 휘처가 제품 목표에 영향을 주는 기대값이 달라질 수 있는 것을 고려하지 않았다. 이런 고려가 되지 않을 경우 정확한 제품의 목표를 달성하는데 한계가 있다. 따라서 본 연구에서는 이러한 문제점을 해결할 수 있는 방법으로서 휘처와 휘처 속성간의 관계를 기반으로 제품의 목표에 영향을 주는 휘처와 휘처 속성의 구성 조합을 찾는 방법을 제안한다. 휘처와 휘처 속성간의 관계란 기존의 휘처 모델[7]에서 휘처간의 관계만 표현되어 있는 것을 휘처 속성간의 관계를 추가하여 휘처 모델을 확장한 것이다.

또한 이전 제품 개발을 통해 축적된 휘처 선택에 필요한 개발 노하우가 신규 제품 개발시에 시스템적으로 적용될 수 있도록 하고, 휘처 구성을 위해 휘처 모델이 복잡해지지 않도록 한다. 휘처 모델은 휘처들의 의미를 표준화하고 통합함으로써, 고객, 도메인 전문가, 개발자 사이의 효과적인 의사소통의 매개체로 사용되고 있다. 그러므로 휘처 구성을 위한 정보는 휘처 모델과 분리하여 관리될 수 있는 방법을 제시 한다.

본 연구에서 제안하는 휘처 구성 방법의 목적은, 제품라인에서 제품을 개발하기 위한 휘처 구성을 할 때 1) 제품의 비즈니스와 품질 목표를 달성하고, 2) 제품 개발 노하우가 시스템적으로 다음 제품에 전수될 수 있도록 휘처 구성 방법을 제공하여, 제품 개발 초기에 제품의 목표에 맞는 휘처와 휘처 속성이 선택될 수 있도록 하는 것이다.

본 연구의 구성은 다음과 같다. 2장에서는 본 연구의 기반이 되는 소프트웨어 제품라인 방법론과 휘처 구성 방법에 대한 관련연구를 알아볼 것이다. 3장에서는 제품라인 방법론에서 휘처 구성 방법에 대해 설명할 것이다. 마지막으로 4장에서는 본 연구의 결론과 향후 연구에 대하여 기술 할 것이다.

2. 관련 연구

소프트웨어 제품라인 공학 분야에서는 휘처 구성 방법에 대한 다양한 연구가 진행되어 왔다. 제품의 기능을 나타내는 기능적(Functional) 휘처와 제품의 비기능을 나타내는 비기능적(Non-functional) 휘처를 고려한 휘처 구성 방법에 대한 연구가 존재하였다.

2.1 기능적 휘처에 대한 휘처구성방법

D. Streitferdt[8]는 OCL(Object Constraint Language) 언어를 사용하여 휘처간의 관계를 표현하는 방법을 제안하였다. D. Sellier[9]와 Botterweck[10]는 휘처간의 상호의존 관계를 시각화하는 모델을 제안하여 휘처간의 상호작용을 이해하기 쉽도록 하였다.

이들 연구는 제품의 기능적 휘처들 간의 의존관계를 통해 휘처를 구성할 수 있도록 가이드를 제공 하지만, 제품의 비기능적 휘처를 고려하는 것을 지원하지 못한다. 따라서 휘처 구성에 의해 제품의 목표가 얼마나 달성되는지에 대한 고려를 할 수 없는 한계가 있다.

2.2 비기능적 휘처를 고려한 휘처구성 방법

휘처 구성 방법에서 비기능적 휘처를 고려하는 것은 제품의 목표에 부합하는 휘처를 선택하기 위해서 반드시 필요한 것이다.

이런 비기능적 휘처를 정의하고, 품질(Quality)을 모델링하는 다양한 연구가 있었다[11][12][13][14][15]. 이런 품질 모델 방법들은 도메인 전문가, 제품라인 관리자, 고객에게 비기능적 휘처를 구별할 수 있게 한다. 하지만 제품라인 공학에 적용된 방법이 아니여서 도메인과 응용 공학의 경계가 없고, 가변성을 제거 하는(휘처를 선택하여 제품을 도출하는) 프로세스가 고려되지 않은 한계가 있다.

D. Zubrow[16], F. Hunleth[17], R. Lopez-Herrejon[18]은 각각 제품라인을 관리 하는데 들어가는 노력(Development Effort), 실행파일의 크기(Binary size), 가변점의 복잡도와 같은 품질 목표를 측정하는 연구를 하였다. J. Sincero[19][20]는 피드백 접근법 (Feedback Approach)으로 휘처 선택과 품질 목표 사이의 연관성을 찾는 연구를 진행하였지만 제품의 품질 목표에 대한 기대값(Expected value)이 없고, 제품을 도출(Derivation)하는 전체 프로세스를 제시하지 못하였다.

D. Benavides[21][22]와 N. Siegmund[3][4][5][6]는 휘처 선택시 각 휘처마다 품질 목표에 영향을 주는 값을 가지고 있고, 품질 목표에 대한 기대값을 이용하여 휘처를 구성하는 방법을 제시하였다. 이들 연구는 본 연구와 가장 근접한 휘처 구성 방법이지만, 제품 목표에 크게 영향을 줄 수 있는 휘처 속성에 대한 고려가 없고, 휘처 마다 제품 목표에 영향을 주는 기대값을 고정으로 가지고 있어 휘처 구성 조합에 따라 휘처가 제품 목표에 영향을 주는 기대값이 달라질 수 있는 것에 대한 고려를 하지 않아 정확한 제품의 목표를 달성하는데 한계가 있다.

위에서 살펴본 바와 같이 휘처 구성 방법과 관련이 있는 다양한 연구가 있었지만, 제품 목표 달성에 영향을 주는 휘처 속성의 가변성과 휘처와 휘처 속성의 구성 조합이 고려된 연구는 제시되지 않았다. 따라서 본 연구에서는 휘처와 휘처 속성간의 관계를 기반으로 제품의 목표를 달성하게 하는 휘처와 휘처 속성의 구성 조합을 찾는 휘처 구성 방법에 대해서 알아보도록 한다.

3. 휘처 구성 방법

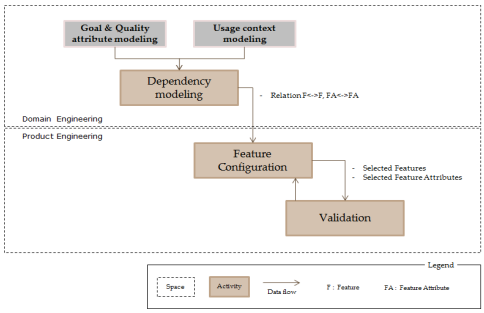
본 연구에서 제안하는 휘처 구성 방법은 소프트웨어 제품라인 공학에서 제품의 목표를 달성하기 위해 휘처와 휘처 속성의 구성 조합을 찾는 방법이다. 이 장에서는 휘처 구성 방법을 위해 제품 목표 달성에 영향을 주는 휘처 속성을 고려한 휘처 모델을 재정의하고, 이에 기반한 휘처 구성 방법에 대한 프로세스를 정의하고 각 활동이 어떻게 진행되는지를 명세할 것이다.

3.1 휘처 구성 방법 개요

본 연구에 제시하는 휘처 구성 방법의 궁극적인 목표는 제품의 목표를 달성할 수 있는 휘처와 휘처 속성의 조합을 찾는 것이다. 이 목표를 달성하기 위해서는 첫째, 제품의 목표에 영향을 주는 휘처와 휘처 속성간의 관계를 분석하고 둘째, 이 관계를 통해 제품의 모든 목표에 부합되는 휘처와 휘처 속성의 구성 조합을 찾는 방법이 제시되어야 한다.

이를 위하여 본 방법에서는 FORM[23] 방법론을 기반으로 하여, 휘처 모델링 활동과 제품 요구사항 분석과 휘처 선택 활동을 연구 범위로 하여 이들 활동을 확장하고 구체화 한다. 휘처 모델링 활동에서는 기존의 휘처 모델링에 휘처 속성간의 관계를 추가하고, 제품 요구사항 분석과 휘처 선택 활동에서는 휘처 구성 방법에 대한 프로세스를 정립하여, 휘처 모델링에서의 제품 가변성을 휘처 선택 활동에서 제거하여 제품의 목표에 부합되도록 휘처와 휘처 속성을 구성하게 된다. [그림 1]는 도메인 지식기반 휘처 구성 방법에 대한 개괄적인 프로세스를 나타낸 것이다. Dependency modeling 활동은 FORM 방법론의 휘처 모델링 활동을 확장한 것으로 휘처 모델에 휘처들 간의 의존관계와 휘처 속성들 간의 의존관계를 정의한다. Feature Configuration은 Dependency modeling 활동에서 정의된 휘처와 휘처 속성간의 관계를 기반으로 제품 목표에 부합하는 휘처와 휘처 속성을 선택하는 활동이다. Validation은 Feature Configuration 활동에서 선택된 휘처와 휘처 속성이 모든 제품 목표들에 부합되는지 확인하는 활동이다. Feature Configuration과 Validation은 FORM 방법론의 휘처 선택 활동을 구체화 한 것이다. 이들 Dependency modeling, Feature Configuration, Validation 활동에 대한 구체적인 내용은 이어지는 절들에서 상세히 설명하도록 한다.

3.2 휘처 의존 관계 정의



[그림 1] 도메인 지식기반 휘처 구성 프로세스

본 절에서는 휘처 속성이 제품 목표 달성에 영향을 주는 관련성과 휘처 속성들간의 관계에 대해서 분석하고, 휘처 속성들간의 관계가 고려된 휘처 모델을 정형적으로 정의 한다.

3.2.1 휘처속성과 제품목표간 관계분석

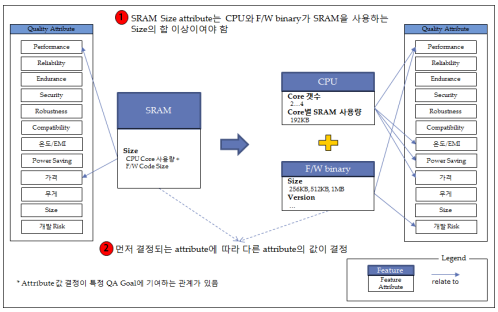
휘처 속성은 제품의 비지니스, 품질 목표 달성과의 관련성을 가지고 있다. 예를 들어 노트북 도메인에 있어서, 노트북 개발의 목표는 가격과 같은 비지니스 목표와 성능, 신뢰성, 견고성 등의 품질 목표가 경쟁사 대비 뛰어난 것이다. 이런 노트북을 개발하기 위해서는 CPU(Intel, AMD), 메모리, 메인보드, 저장 디스크(HDD, SSD), 파워 서플라이, BIOS, OS(리눅스, 윈도우즈, 맥) 등의 휘처와 CPU의 Core갯수, 메모리의 크기, 저장 디스크의 크기, 파워 서플라이의 용량 등의 휘처 속성을 제품의 목표를 달성할 수 있도록 선택하여야 한다. 제품의 목표 달성은 어느 제조사의 CPU를 선택할지, HDD와 SSD 중에 어떤 저장 디스크를 선택할지에 대한 휘처 선택도 연관성이 있지만, CPU의 Core 갯수, 메모리의 크기, 저장 디스크의 크기, 파워 서플라이의 용량과 같은 휘처 속성도 제품의 가격, 성능, 신뢰성 등의 제품 목표에 큰 영향을 주게 된다.

[그림 2]는 SSD(Solid-State Drive) 도메인에서 휘처 속성과 품질 속성간의 영향을 주는 관계를

분석한 예제이다. 이 예에서 SRAM휘처의 Size 속성, CPU휘처의 Core 갯수 속성, F/W binary 휘처의 Size속성과 같은 휘처 속성들은 제품의 품질 속성(Quality Attribute)에 영향을 주는 관계를 가진다. 이들은 각각SRAM 휘처의 Size속성은 품질 속성인 성능과 가격에, CPU 휘처의 Core 갯수 속성은 품질 속성인 성능, 온도/EMI, Power Saving, 가격에, F/W binary 휘처의 Size 속성은 품질 속성인 성능과 개발 Risk에 영향을 주는 관계가 있다.

위에서 살펴본 노트북과 SSD 도메인 예제에서 알 수 있듯이 제품 개발의 목표 달성을 위해서는 휘처 선택 뿐만 아니라 휘처 속성에 대한 선택도 중요한 결정이다. 따라서 제품 개발을 위한 휘처 구성시에는 휘처와 휘처 속성에 대한 선택을 함께 진행하여야 한다.

지금까지 휘처 구성 방법에서 제품 개발의 목표 달성을 위해서는 휘처 속성의 선택이 고려되어야 하는 이유에 대해 알아 보았다. 다음은 휘처 속성들 간의 관계에 대해서 설명하겠다.



[그림 2] SSD 휘처 속성과 품질 속성간의 관계

[그림 2]에서는 휘처 속성과 품질 속성간의 관계 외에 휘처 속성들 간의 관계에 대한 분석도 포함되어 있다. SRAM 휘처의 Size 속성값은 CPU 휘처의 Core 갯수와 Core별 SRAM 사용량 속성값을 곱한 값과 F/W binary 휘처의 Size 속성값을 더한 값이 되어야 하는 관계가 있다. 또한 이들 속성들은 제품의 품질 속성과 관련이 있고, 제품에서 중요한 품질 속성에 따라 그 품질 속성에 크게 영향을 주는 휘처의 속성값이 먼저 결정될 수 있다. 예를 들어 예제의 세가지 휘처 속성은 모두 성능에 영향을 주게 되는데, 개발하려는 제품에서 성능이 아주 중요한 품질 속성이라면 성능에 더 크게 영향을 주는 휘처의 속성값을 우선 결정되고, 나머지 휘처의 속성값들의 결정은 먼저 결정된 속성값에 영향을 받게 된다.

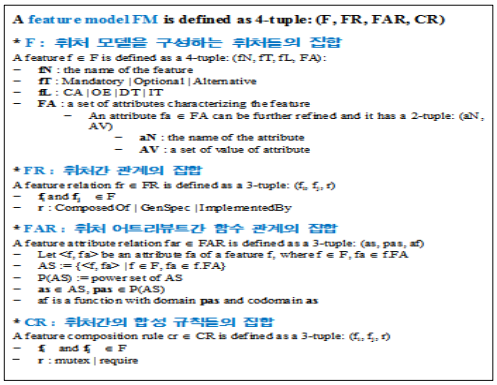
[그림 2]에서 살펴본 것 처럼 휘처 속성들 간에는 함수적 관계가 존재하는 것을 알 수 있다. 또한 이 함수적 관계를 통해 휘처 구성시에 휘처 속성 값을 결정하는데 영향을 주는 것을 알 수 있다.

휘처 모델에서는 휘처들간의 관계와 합성 규칙을 제공하여 휘처들간의 관계를 표현할 수 있도록 한다. 하지만 휘처 속성들간의 관계에 대한 것은 고려되어 있지 않다. 다음 항에서는 휘처 속성들간의 관계를 고려한 휘처 모델을 재정의 할 것이다.

3.2.2 휘처 모델의 정형화

휘처 모델은 제품라인의 제품들 간의 공통점과 차이점을 휘처 중심으로 표현한 모델로서 다양한 관심영역을 갖는 시스템 개발자 및 고객 간의 훌륭한 의사소통의 수단이 된다. 또한 휘처 모델은 분석 단계에서의 모델로만 그치는 것이 아니라, 소프트웨어 생명 주기 전반에 영향을 주는 모델이다.

이번 항에서는 Feature Model Formalization [24]에서 제시된 휘처 모델의 정의를 수정하여, 휘처 속성들간의 관계가 고려된 휘처 모델을 재정의 한다. 정의 1은 휘처 모델을 재정의 한 것이다.



[정의 1] 휘처 모델

휘처 모델 정의에서 하나의 휘처 모델은 F, FR, FAR, CR의 요소를 가지고 있다. 이들 요소들은 다음과 같이 정의된다.

- F: 휘처 모델을 구성하는 휘처들의 집합
- FR: 휘처들 간의 관계의 집합
- FAR: 휘처 속성들간의 함수 관계의 집합
- CR: 휘처간의 합성 규칙들의 집합

휘처들의 집합 F에서 하나의 휘처 f는 fN (휘처의 이름), fT(휘처의 종류), fL(휘처가 속한 계층), FA(휘처의 속성 집합)의 요소를 가진다. 정의 1에서 대문자 알파벳으로 시작되는 튜플의 요소는 집합을 의미한다. 휘처의 종류에는 제품라인 내의 모든 제품에 존재하는 휘처인 필수 휘처(Mandatory Feature)와 특정 제품에만 존재하는 선택적 휘처(Optional Feature), 그리고 여러 휘처중 하나의 휘처만이 제품내에 존재해야 하는 택일적 휘처(Alternative Feature)가 있다. 제품간의 공통점은 필수 휘처를 통해 나타내고, 차이점은 선택적 휘처와 택일적 휘처를 통해 나타낸다. 휘처는 그 특징에 따라 CA(Capability Layer), OE(Operating Environment Layer), DT (Domain Technology Layer), IT(Implementation Technique Layer)의 네가지 계층으로 분류된다. 휘처의 속성 집합 FA에서 하나의 휘처 속성 fa는 aN(휘처 속성의 이름), AV(휘처 속성값의 집합)의 요소를 가진다.

휘처들 간의 관계의 집합 FR에서 하나의 휘처 관계 fr은 fi, fj, r의 요소를 가진다. 휘처 fi와 fj는 휘처들의 집합 F의 한 원소이고, r은 휘처 간의 관계이다. 휘처 간의 관계는 ComposedOf, GenSpec, ImplementedBy가 있다. 하나의 휘처가 다른 여러 개의 휘처로 구성되는 경우에는 ComposedOf 관계가 존재하고, 실선으로 표현한다. 또한, 하나의 휘처가 다른 휘처들의 추상화된 개념이라면, 이들 휘처들 간에는 GenSpec (Generalization/Specialization) 관계가 존재하고, 점선으로 표현된다. 그리고 다른 계층간에 Feature의 연결은 ImplementedBy 관계로 맺어지고, 이것은 하나의 휘처가 다른 휘처를 구현하는 기술로 사용되는 경우로서, 점선으로 표현된다.

휘처 속성들간의 함수 관계의 집합 FAR에서 하나의 휘처 속성 관계 far은 “as”, “pas”, “af”의 요소를 가진다. “as”는 하나의 휘처 속성이고, “pas”는 하나 이상의 휘처 속성이다. 그리고 “af”는 “as”와 “pas” 간의 함수 관계이다. 즉 “af”는 하나의 휘처 속성과 여러개의 휘처 속성들 간의 함수 관계로 정의된다. 정의 1에서 다음과 같이 far을 정의하였다. 휘처 모델에서 모든 휘처 속성들의 집합을 AS로 정의하고, AS는 <f, fa>로 정의되는 원소들의 집합으로 나타내었다. <f, fa>는 휘처 f의 휘처 속성 fa로 정의되고, 각각의 f와 fa는 다음과 같은 의미를 가진다.

- f : 휘처들의 집합 F에서 하나의 휘처 f (f Î F)
- fa: 휘처 f의 요소인 휘처의 속성집합 FA의 한 휘처 속성 fa (fa Î f.FA)

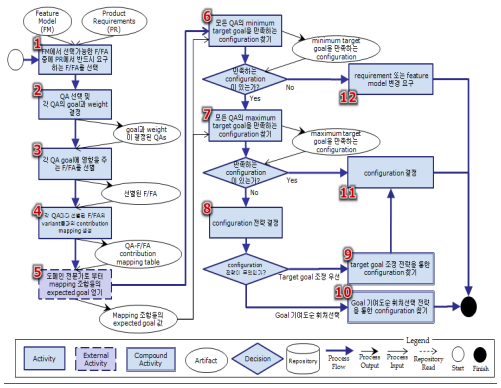
휘처 속성 관계 far의 요소인 “as”는 위에서 정의된 AS의 한 원소이고, “pas”는 AS의 멱집합 (power set)의 한 원소이다. 마지막으로 “af”는 “pas”를 domain으로 “as”를 codomain으로 가지는 함수로 정의하였다.

마지막으로, 휘처들 간의 합성 규칙(Composition Rule)들의 집합 CR에서 하나의 휘처 합성 규칙 cr은 fi, fj, r의 요소를 가진다. 휘처 fi와 fj는 휘처들의 집합 F의 한 원소이고, r은 휘처 간의 합성 규칙이다. 휘처 간의 합성 규칙은 requires와 mutex가 있다. requires 규칙은 휘처 집합의 한 휘처가 선택되어졌을 때 반드시 함께 선택 되어져야 하는 휘처와의 규칙이고, mutex 규칙은 휘처 집합의 한 휘처가 선택되어 졌을 때 절대 선택되면 안되는 휘처와의 규칙이다.

3.3 휘처 구성 프로세스

이번 절에서는 앞에서 재정의된 휘처 모델을 기반으로 휘처 구성 방법을 위한 휘처 구성 프로세스를 정의하고, 제품의 개발 목표를 달성하기 위해 정의된 각 활동들에 대해서 명세 한다.

휘처 구성 방법의 목표는 제품의 모든 목표에 부합되는 휘처와 휘처 속성의 구성 조합을 찾는 것이다. [그림 3]은 이런 휘처와 휘처 속성의 구성 조합을 찾기 위한 휘처 구성 방법에 대한 전체 프로세스이다.



[그림 3] 휘처 구성 프로세스

이들 휘처 구성 프로세스 내의 활동들의 역할을 정리하면 다음과 같다.

1. “FM에서 선택가능한 F/FA중에 PR에서 반드시 요구하는 F/FA를 선택” 활동은 휘처 모델(FM)과 제품 요구사항(PR)을 입력으로 받아, 제품 요구사항에서 반드시 요구하는 휘처(F)와 휘처 속성(FA)을 선택하여 휘처 모델에서 제품의 가변성을 줄이는 역할을 한다.

2. “QA 선택 및 각 QA의 goal과 weight 결정” 활동은 제품이 속한 산업 동향 분석과 경쟁사의 품질 목표 분석을 통해서, 제품에 요구되는 품질 속성(QA)을 결정하고 각 품질 속성의 Target Quality Goal과 Minimum Quality Goal을 결정한다. 여기서 Target Quality Goal은 제품이 속한 산업 동향과 경쟁사 분석을 통해 시장에서 성공할 수 있는 제품이 될 수 있는 품질 속성들의 목표값이고, Minimum Quality Goal은 품질 속성들의 우선 순위에 따라 목표값을 조정할 때 우선순위가 낮은 품질 속성이라고 하더라도 최소한 달성해야하는 품질 목표값을 의미한다. 그리고 결정된 품질 속성들이 제품 목표 달성에 있어서 상대적으로 중요한 정도를 나타내는 우선순위(Weight)를 결정 한다. <표 1>은 SSD 도메인에서의 노트북 제품에 대한 품질 속성의 목표와 우선순위 결정 예제이다.

<표 1> 품질 속성의 목표와 우선순위 결정

	QA1		QA2	QA3	QA4	QA5
	I/O 속도	Initialize Time	가격	Power Consumption	Temperature	Warranty
Target Goal	PCMark Score >100,000	<1.5 S	<$200	<46mW	Max <70도 Mean <50도	> 4 years
Minimum Goal	PCMark Score >90,000	<2.0 S	<$250	<50mW	Max <75도 Mean <60도	> 3 years
Weight (Priority)	40	5	25	15	10	5

3. “각 QA goal에 영향을 주는 F/FA를 선별” 활동은 2번 활동에서 결정된 품질 속성을 입력으로 받아, 이들 각각의 품질 속성 목표에 영향을 주는 가변점들을 선별하는 역할을 한다. 이런 가변점들은 품질 속성 목표에 영향을 주는 가변성을 가진 휘처와 휘처 속성을 말한다.

4. “각 QA마다 선별된 F/FA의 variant들과의 contribution mapping 생성” 활동은 각 품질 속성과 각각의 품질 속성에 영향을 주는 가변

치들(3번 활동에서 선별된 휘처와 휘처 속성의 가변치들을 의미함)과의 매핑 테이블을 생성하는 역할을 한다.

<표 2> 품질 속성과 가변치들의 매핑 테이블

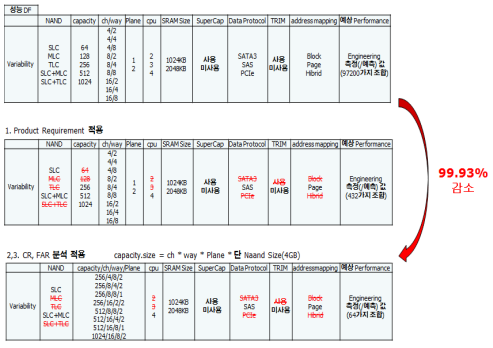
성능 QA

	NAND	capacity	ch/way	Plane	cpu	SRAM Size	SuperCap	Data Protocol	TRIM	address mapping	예상 Performance
Variability	SLC MLC TLC SLC+MLC SLC+TLC	64 128 256 512 1024	4/2 4/4 4/8 8/2 8/4 8/8 16/2 16/4 16/8	1 2	2 3 4	1024KB 2048KB	사용 미사용	SATA3 SAS PCIe	사용 미사용	Block Page Hibrid	Engineering 측정(/예측) 값 (97200가지 조합)

* **Configuration 조합의 Variability를 줄이는 방법**

1. Product Requirement에 의해 필수적으로 선택되어져야 하는 휘처와 휘처속성을 결정하여 Variability를 줄임
2. Mandatory Feature와 1에서 선택된 것들의 CR(휘처간의 합성 규칙들의 집합) 분석을 통해 Variability를 줄임
3. FAR(휘처 어트리뷰트간 관계의 집합) 분석을 통해 Variability를 줄임
4. Variability Optimization 방법[3-6]의 적용을 통해 Variability를 줄일 수 있음

[그림 4] 품질 속성에 영향을 주는 가변치의 조합을 줄이는 방법



[그림 5] 가변치의 조합을 줄이는 방법 적용 예

<표 2>는 성능 품질 속성에 영향을 주는 휘처와 휘처 속성의 가변치들을 매핑한 예제이다. 이 활동에서 생성된 매핑 테이블은 5번 활동에서 전문가의 공학 측정값(Engineering Data)을 요구하는데 사용된다. 하지만 <표 2>의 예제에서 처럼, 가변치들의 단순 조합(이 예제에서는 97,200개의 조합)에 의해 공학 측정값을 요구하는 것은 실현이 불가능한 일이다. 따라서 가변점들의 가변성을 줄여 가변치들의 조합을 줄일 수 있는 방법이 필요하다. 그림8은 가변치들의 조합을 줄이는 방법을 제시한 것이다. [그림 4]에서 제시한 방법 중에 1~3의 세가지 방법을 적용하여, <표 2>의 가변치 조합을 줄인 결과가 [그림 5]이다. 이 예제에서는 가변치의 조합이 99.93%가 줄어드는 효과를 보았다.

5. "도메인 전문가로 부터 mapping 조합들의 expected goal 얻기" 활동은 각각의 품질 속성에 영향을 주는 가변치들의 매핑 테이블을 입력받아 각 조합들의 예상 목표값(expected goal)을 해당 전문가로부터 얻는 역할을 한다.

6. "모든 QA의 minimum target goal을 만족하는 configuration 찾기" 활동은 전문가로부터 품질 속성에 영향을 주는 매핑 조합들의 예상 목표값을 기반으로 각 품질 속성 마다 Minimum Quality Goal을 만족하는 매핑 조합을 찾고([그림 6] 참조), 모든 품질 속성의 Minimum Quality Goal을 만족하는 휘처와 휘처 속성의 구성 조합([그림 7] 참조)을 찾는 역할을 한다. 이 활동을 하는 이유는 개발하는 제품이 시장에서 이윤창출 또는 비지니스 전략적 가치가 있게하기 위해서는 모든 품질 속성들의 최소한의 목표는 달성되어야 하기 때문이다. 만약 Minimum Quality Goal도 만족 시키는 구성 조합이 없을 경우에는 휘처 모델의 변경 또는 제품 요구사항이 변경되어야 한다. 6번 활동을 완료한 후에 아래 6-1에서는 모든 품질 속성의 Minimum Goal을 만족하는 휘처와 휘처 속성의 구성 조합이 있는지를 판단한다.

온도 QA — Minimum Goal: Max < 75도, Mean < 60도

	capacity	F/F	ch/way	cpu
Goal 만족	1024	2.5" 3.5"	8/8	3CPU 4CPU

성능 QA — Minimum Goal: PCMark Score > 90,000

	capacity	ch/way	Cpu	SRAM Size	...
Goal 만족	1024	8/8	3CPU 4CPU	2048KB	

[그림 6] 각 품질 속성의 Minimum Goal을 만족하는 조합 예

	capacity	F/F	ch/way	Cpu	SRAM Size
모든 Goal 만족	1024	2.5" 3.5"	8/8	3CPU 4CPU	2048KB

[그림 7] 모든 품질 속성의 Minimum Goal을 만족하는 조합 예

6-1. <Decision> 만족하는 configuration 이 있는가? 휘처 구성 조합이 있을 경우 7번 활동을 진행하고, 없을 경우 12번 활동을 진행한다.

7. "모든 QA의 maximum target goal을 만족하는 configuration 찾기" 활동은 전문가로부터 품질 속성에 영향을 주는 매핑 조합들의 예상 목표값을 기반으로 각 품질 속성 마다 Target Quality Goal을 만족하는 매핑 조합을 찾고, 모든 품질 속성의 Target Quality Goal을 만족하는 휘처와 휘처 속성의 구성 조합을 찾는 역할을 한다. 이 활동을 하는 이유는 개발하는 제품이 목표하는 최대한의 Quality Goal를 달성 시키는 구성 조합이 있을 경우에는 이후의 프로세스를 진행할 필요없이 휘처 구성을 종료할 수 있기 때문이다. 하지만 대부분의 경우에는 목표를 이상적으로 계획하기 때문에 제품이 목표하는 최대치의 품질 목표를 달성하기는 쉽지 않다. 7번 활동을 완료한 후에 아래 7-1에서는 모든 품질 속성의 Target Quality Goal을 만족하는 휘처와 휘처 속성의 구성 조합이 있는지를 판단한다.

7-1. <Decision> 만족하는 configuration 이 있는가? 휘처 구성 조합이 있을 경우 11번 활동을 진행하고, 없을 경우 8번 활동을 진행한다.

8. "configuration 전략 결정" 활동에서는 제품의 목표를 달성하는 휘처와 휘처 속성들의 구성 조합을 찾기위해, target goal 조정 전략과 Goal 기여도순 휘처선택 전략 중에 하나를 결정하는 역할을 한다. 본 활동을 진행하는 단계에서는 이미 이전의 6, 7번 활동을 통해 모든 품질 속성의 Minimum Quality Goal을 달성하는 휘처와 휘처 속성의 구성 조합은 존재하고, Target Quality Goal을 달성하는 조합은 존재하지 않는 상태이다. 따라서 휘처 구성 전략을 통해 최선의 품질 속성 목표를 달성하는 휘처와 휘처 속성의 구성 조합을 찾아야 한다. 본 활동에서 결정된 휘처 구성 전략에 따라 아래 8-1에서는 이후 진행할 활동을 선택한다.

8-1. <Decision> Configuration 전략이 무엇인가? Target goal 조정 전략일 경우 9번 활동을 진행하고, Goal 기여도순 휘처선택 전략일 경우 10번 활동을 진행한다.

9. "target goal 조정 전략을 통한 configuration 찾기" 활동은 품질 속성들의 Target Quality Goal 속성값을 조정하여 휘처 구성 조합을 찾아내는 역할을 한다. [그림 6]에서 이 활동은 Compound Activity로 정의되어 있다. Compound Activity는 본 활동이 다시 여러개의 세부 활동으로 구성되어 있다는 것을 의미한다. 본 활동의 세부 활동에 대해서는 3.3.1 항에서 다시 정의할 것이다. 이 활동이 완료된 이후에는 11번 활동을 진행한다.

10. "Goal 기여도순 휘처선택 전략을 통한 configuration 찾기" 활동은 휘처 모델에서의 가변점(휘처나 휘처 속성)들이 QA Goal(Quality Attribute의 Quality Goal)에 미치는 기여도 분석을 통해 기여도가 높은 순으로 가변점의 가변치(Variant)를 선택하며 휘처 구성 조합을 찾아내는 역할을 한다. 이 활동이 완료되면 휘처 구성 조합이 결정되고, 휘처 구성 프로세스는 종료된다. 본 활동도 Compound Activity로 정의되어 있고, 세부 활동에 대해서는 3.3.2 항에서 다시 정의할 것이다.

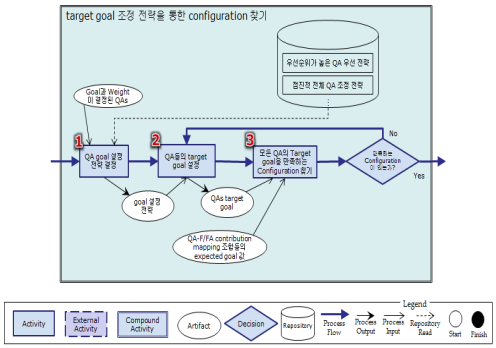
11. "configuration 결정" 활동은 7번 활동에서 모든 품질 속성의 Target Quality Goal을 만족하는 휘처 구성 조합이 있을 경우나 9번 활동이 완료되었을 경우에 진행된다. 본 활동이 진행되는 단계에서는 모든 품질의 Target Quality Goal을 만족하는 조합이 있고, 제품 개발 목표를 달성

할 수 있는 상태이다. 따라서 남아 있는 휘처와 휘처 속성의 선택은 2번 활동에서 결정된 품질 속성의 우선 순위에 따라 각 품질 속성을 최대한으로 달성하는 것을 선택한다. 이 활동이 완료되면 휘처 구성 조합이 결정되고, 휘처 구성 프로세스는 종료된다.

12. "requirement 또는 feature model 변경 요구" 활동은 6번 활동에서 제품 품질 속성의 Minimum Quality Goal도 만족 시키는 구성 조합이 없을 경우에 진행된다. 따라서 이 단계에서는 휘처 모델 변경 또는 제품의 요구사항 변경을 요구하고, 휘처 구성 프로세스를 종료한다.

3.3.1 Target Goal 조정 전략

앞에서 명세한 것처럼 "target goal 조정 전략을 통한 configuration 찾기" 활동은 제품 품질 속성들의 Target Quality Goal 속성값을 조정하여, 조정된 값을 만족하는 휘처 구성 조합을 찾아내는 역할을 한다. 이 활동의 세부 활동들의 프로세스는 [그림 8]과 같다. 이 활동의 세부 활동은 [그림 9]에 정리되어 있다.



[그림 8] Target Goal 조정 전략 프로세스

1. QA goal 설정 전략결정
2. QA들의 target goal 설정
3. 모든 QA의 Target goal을 만족하는 Configuration 찾기
   3-1. <Decision> 만족하는 Configuration 이 있는가?
   휘처 구성 조합이 없을 경우 2번 활동으로 돌아감, 있을 경우 QA goal 설정 전략 완료

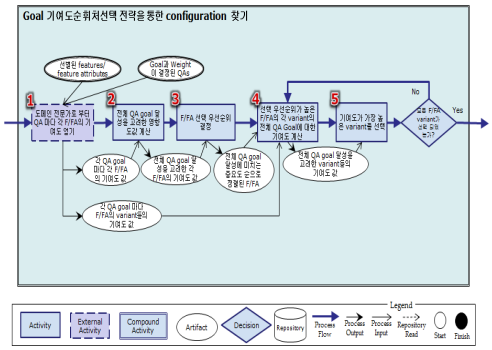
[그림 9] Target Goal 조정 전략의 세부 활동

이들 Target Goal 조정 전략 프로세스 내의 활동들의 역할에 대해 정리하면 다음과 같다.

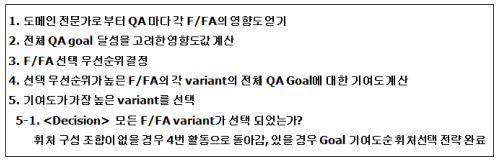
1. "QA goal 설정 전략 결정" 활동은 제품 품질 속성의 Target Quality Goal 속성값을 조정하는 세부 전략을 결정하는 역할을 한다. 세부 전략에는 '우선순위가 높은 QA 우선 전략', '점진적 전체 QA 조정 전략' 등이 있고 전략이 결정되면 이후 활동에서 저장소에 보관되어 있는 해당 전략의 알고리즘을 사용하게 된다.

3.3.2 Goal 기여도순 휘처 선택 전략

앞에서 명세한 것처럼 "Goal 기여도순 휘처 선택 전략을 통한 configuration 찾기" 활동은 휘처 모델에서의 가변점 (휘처나 휘처 속성)들이 QA Goal(Quality Attribute의 Quality Goal)에 미치는 기여도 분석을 통해 기여도가 높은 순으로 가변점의 가변치(Variant)를 선택하며 휘처 구성 조합을 찾아내는 역할을 한다. 본 전략에서는 휘처와 휘처 속성을 구성할 때, 전체 품질 속성에 미치는 기여도가 높은 가변점일수록 제품 개발의 목표를 달성에 있어서 중요한 선택이 된다고 가정한다. 따라서 본 전략은 제품 전체 품질 속성에 미치는 기여도 높은 가변점 순으로 가변치를 결정해 나간다. 이 활동의 세부 활동들의 프로세스는 [그림 10]과 같다. 이 활동의 세부 활동은 [그림 11]에 정리되어 있다.



[그림 10] Goal 기여도순 휘처 선택 전략 프로세스



[그림 11] Goal 기여도순 휘처 선택 전략 활동

이들 Goal 기여도순 휘처 선택 전략 프로세스 내의 활동들의 역할에 대해 정리하면 다음과 같다.

1. "도메인 전문가로 부터 QA 마다 각 F/FA의 기여도 얻기" 활동은 도메인 전문가로 부터 가변점들이 제품의 품질 속성들에 미치는 기여도와 가변점의 가변치들이 각 품질 속성에 상대적으로 영향을 주는 기여도를 얻어오는 역할을 한다. 전자의 기여도는 본 전략에서 어떤 가변점의 가변치를 먼저 결정할 지에 대한 우선순위 판단을 위한 것이고, 후자의 기여도는 실제 가변점의 가변치를 결정할 때 필요한 것이다. <표 3>은 SSD 제품의 가변점들이 품질 속성에 미치는 기여도를 분석한 예제이고, <표 4>는 가변치들이 품질 속성에 미치는 상대적인 기여도를 분석한 예제이다.

<표 3> 가변점들이 품질 속성에 미치는 기여도

VP	가격	Response Time	I/O 속도	Power Consumption	Warranty
NAND Type	40	30	35	40	100
Data Protocol	30	20	35	0	0
SuperCap	10	0	20	20	0
내부 CPU	20	50	10	40	0

<표 4> 가변부가 품질 속성에 미치는 기여도

NAND Type	가격	Response Time	I/O 속도	Power Consumption	Warranty
SLC	25	100	100	100	100
MLC	75	90	65	90	75
TLC	100	80	30	80	50
SLC+TLC	90	85	50	82	60
SLC+MLC	65	95	75	92	85

2. "전체 QA goal 달성을 고려한 기여도값 계산" 활동은 가변점들 마다 제품 전체의 품질 속성에 미치는 기여도를 계산하는 역할을 한다. 여기서 계산된 기여도를 기반으로 어떤 가변점의 가변치를 먼저 결정할 지에 대한 우선순위 판단하게 된다. <표 5>는 가변점들이 제품의 전체 품질 속성들에 미치는 기여도를 계산한 예제이다. 예제의 VPW_Total 열의 값들이 이 기여도를 계산한 결과이다.

<표 5> 가변점이 제품 전체의 품질 속성에 미치는 기여도 계산 예

	QA1	QA2	QA3	QA4	QA5	VPW_Total
QAW	40	5	30	20	5	
NAND Type	40%	30%	35%	40%	100%	41
Data Protocol	30%	20%	35%	0	0	23.5
SuperCap	10%	0	20%	20%	0	14
내부 CPU	20%	50%	10%	40%	0	21.5

QAW : 제품 목표 달성에 있어서 제품 품질 속성들의 우선 순위

3. "F/FA 선택 우선순위 결정" 활동은 이전 활동에서 계산된 가변점들의 기여도를 기반으로 어떤 가변점의 가변치를 먼저 결정할지에 대한 우선순위를 결정하는 역할을 한다. <표 6>은 가변점들의 기여도 우선순위를 분석한 예제이다. 우선 순위 열의 값은 가변점들의 기여도 값(VPW_Total)이 높은 순으로 결정되었다.

<표 6> 가변점의 기여도 우선 순위 분석예

	QA1	QA2	QA3	QA4	QA5	VPW_Total	우선 순위
QAW	40	5	30	20	5		
NAND Type	40%	30%	35%	40%	100%	41	1
Data Protocol	30%	20%	35%	0	0	23.5	2
SuperCap	10%	0	20%	20%	0	14	4
내부 CPU	20%	50%	10%	40%	0	21.5	3

QAW : 제품 목표 달성에 있어서 제품 품질 속성들의 우선 순위

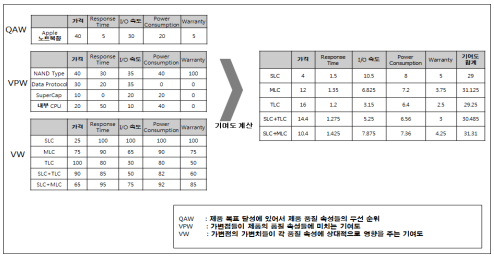
4. "선택 우선순위가 높은 F/FA의 각 variant의 전체 QA Goal에 대한 기여도 계산" 활동은 이전 활동의 가변점들 간의 우선 순위 결과를 입력으로 받아서 우선 순위가 가장 높은 가변점의 가변치를 선택하기 위해서 각 가변치들이 제품 전체 품질 속성에 미치는 기여도를 계산하는

역할을 한다. 가변치들이 제품 전체 품질 속성에 미치는 기여도를 계산하기 위해서는 이전 활동에 분석된 다음의 세가지 분석 결과를 사용한다.

- 품질 속성들이 제품 목표 달성에 있어서 상대적으로 중요한 정도를 나타내는 우선순위 (QAW)
- 가변점들이 제품의 품질 속성들에 미치는 기여도 (VPW)
- 가변점의 가변치들이 각 품질 속성에 상대적으로 영향을 주는 기여도 (VW)

[그림 12]는 가변점의 모든 가변치들이 제품 전체 품질속성에 미치는 기여도를 계산한 예제이다.

5. “기여도가 가장 높은 variant를 선택” 활동은 이전 활동에서 계산된 가변점의 모든 가변치들의 기여도를 기반으로 기여도가 가장 높은 가변치를 선택하는 역할을 한다. 하지만 선택된 가변치가 모든 품질 속성의 Minimum Quality Goal을 만족하는 휘처와 휘처 속성의 구성 조합([그림 12] 참조)에 없을 경우에는 기여도가 다음으로 높은 가변치를 선택한다. 왜냐하면 선택된 휘처와 휘처 속성이 제품의 모든 품질 속성의 Minimum Quality Goal 값은 만족하여야 하기 때문이다.



[그림 12] 가변치가 제품 품질속성에 미치는 기여도 계산 예

5-1. <Decision> 모든 F/FA variant가 선택되었는가? 모든 가변점들의 가변치가 선택되지 않았으면 4번 활동으로 돌아가고, 모든 가변점들의 가변치가 선택되었다면 모든 휘처 구성 조합이 결정되었으므로, 휘처 구성 프로세스는 종료된다.

4. 결 론

제품라인공학에서 성공적인 제품을 개발하기 위해서는 개발 초기 단계 부터 제품의 비지니스 목표와 품질 목표에 맞는 휘처와 휘처 속성을 선택하는 것이 중요하다. 특히 하드웨어와 소프트웨어의 통합 개발에서는 개발 초기에 선택된 휘처와 휘처 속성이 개발 목표를 달성하지 못 할 경우에는 하드웨어를 변경해야 하는 문제까지 야기할 수 있어서 그 중요성이 더욱 크다.

본 연구에서는 제품라인 공학에서 제품의 목표를 달성하기 위한 휘처와 휘처 속성의 구성 조합을 찾는 휘처 구성 방법을 제안하기 위해, 제품 목표 달성에 영향을 주는 휘처 속성을 고려한 휘처 모델을 재정의하였고, 이에 기반한 휘처 구성 방법에 대한 프로세스를 제시하였다.

휘처 구성시 제품의 개발 목표 달성에 주안점을 둔 본 연구는 기존의 연구에 비해 다음과 같은 차이점을 가진다. 첫째 휘처 속성이 제품 목표에 미치는 영향을 분석하였고, 휘처 구성시에 휘처 뿐만 아니라 휘처 속성도 제품 목표를 달성하는 것을 고려하여 선택되도록 하였다. 둘째, 휘처와 휘처 속성의 구성 조합에 따라 제품 목표에 영향을 주는 정도가 달라질 수 있는 것에 대한 고려를 하였다. 이런 차이점은 본 연구가 보다 정확한 제품의 목표 달성에 기여하는 휘처 구성이 될 수 있는 장점을 가진다.

참 고 문 헌

[1] M. Steger, C. Tischer, B. Boss, A. Müller, O. Pertler, W. Stolz, and S. Ferber, “Introducing PLA at Bosch Gasoline Systems: Experiences and Practices,” in SPLC 2004, Boston, MA, USA, 2004, pp. 34-50.

[2] S. Deelstra, M. Sinnema, and J. Bosch, “Product Derivation in Software Product Families: A Case Study,” Journal of Systems and Software, vol. 74, pp. 173-194, 2005.

[3] N. Siegmund, , S. Kolesnikov, C. Kastner, S. Appel, D. Batory, M. Rosenmuller, and G. Saake. Predicting performance via automated feature-interaction detection. In Proceedings of the 34th International Conference on Software Engineering, ICSE '12, New York, NY, USA, 2012. ACM.

[4] N. Siegmund, M. Rosenmuller, C. Kastner, P. G. Giarrusso, S. Apel, and S. S. Kolesnikov. Scalable prediction of non-functional properties in software product lines. In Software Product Line Conference (SPLC), 2011 15th International, pages 160 –169, August 2011.

[5] N. Siegmund, M. Rosenmuller, M. Kuhlemann, C. Kastner, S. Apel, and G. Saake. Spl conqueror: Toward optimization of non-functional properties in software product lines. Software Quality Journal, 1(3):1–31, June 2011.

[6] N. Siegmund, M. Rosenmuller, M. Kuhlemann, C. Kastner, and G. Saake. Measuring non-functional properties in software product lines for product derivation. In VAMOS, pages 1–31, June 2010.

[7] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, and A.S. Peterson. "Feature-oriented domain analysis (FODA) feasibility study," Technical Report, CMU/SEI-90-TR-21, 1990.

[8] D. Streitferdt, M. Riebisch, and I. Philippow. Details of Formalized Relations in Feature Models Using OCL. Engineering of Computer-Based Systems, 00:297–304, 2003.

[9] D. Sellier and M. Mannion. Visualizing Product Line Requirement Selection Decision Inter-dependencies. In Workshop on Visualisation in Software Product Line Engineering, 2007.

[10] Botterweck, G., Nestor, D., PreuBner, A., Cawley, C., & Thiel, S. (2007). Towards supporting feature configuration by interactive visualization.

[11] Glinz, M. (2007). On non-functional requirements. In Proceedings of the International Conference on Requirements Engineering (RE), IEEE Computer Society, pp. 21–26.

[12] Chung, L., & do Prado Leite, J. (2009). On non-functional requirements in software engineering. In Conceptual modeling: Foundations and applications (Vol. 5600, Chap 19, pp. 363–379). Berlin: Springer, LNCS.

[13] Mccall, J. A., Richards, P. K., & Walters, G. F. (1977). Factors in software quality. Vol. 1. Concepts and definitions of software quality. Technical Report ADA049014, General Electric Co Sunnyvale California.

[14] Boehm, B. W., Brown, J. R., Kaspar, H., Lipow, M., Macleod, G. J., & Merritt, M. J. (1978). Characteristics of software quality (TRW series of software technology). Amsterdam: Elsevier.

[15] Bøegh, J., Depanfilis, S., Kitchenham, B., & Pasquini, A. (1999). A method for software quality planning, control, and evaluation. IEEE Software, 16, 69–77.

[16] Dave Zubrow and Gary Chastek. Measures for Software Product Lines. Technical Report CMU/SEI-2003-TN-031, Carnegie Mellon University, 2003.

[17] F. Hunleth and R. Cytron. Footprint and Feature Management Using Aspect-Oriented Programming Techniques. In Proc. Int'l Conf. on Languages, Compilers, and Tools for Embedded Systems, pages 38–45. 2002.

[18] R. Lopez-Herrejon and S. Apel. Measuring and Characterizing Crosscutting in Aspect-Based Programs: Basic Metrics and Case Studies. In Proc. Int'l Conf. Fundamental Approaches to Software Engineering. 2007.

[19] Sincero, J., Spinczyk, O., & Schro¨der-Preikschat, W. (2007). On the configuration of non-functional properties in software product lines. In Software Product Line Conference (SPLC), Doctoral Symposium (pp. 167–173). Kindai Kagaku Sha Co. Ltd.

[20] Sincero, J., Schro¨der-Preikschat, W., & Spinczyk, O. (2010). Approaching non-functional properties of software product lines: Learning from products. In Proceedings of Asia-Pacific Software Engineering Conference (APSEC), IEEE Computer Society, pp. 147–155.

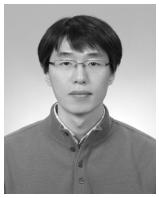
[21] D. Benavides, A. Ruiz-Cort´es, and P. Trinidad. Automated Reasoning on Feature Models. Proc. Int'l Conf. on Advanced Information Systems Engineering, 2005.

[22] D. Benavides et al. FAMA: Tooling a Framework for the Automated Analysis of Feature Models. In Workshop on Variability Modelling of Software-intensive Systems, 2007.

[23] K. Kang, S. Kim, J. Lee, K. Kim, E. Shin and M. Huh, "FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures," Annals of Software Engineering, Vol.5, pp.143-168, 1998,

[24] 김보경, "Feature Model Formalization," Technical Memo POSTECH/CS/SE-98-TM-1, August 1998.

저자 소개



배 성 진

2002년 한동대학교 전산전자공학부 졸업(학사)
2001년~2007년 안철수연구소 주임연구원
2007년~현재 삼성전자 책임연구원
2014년 포항공과대학교 정보전자융합공학부 졸업(석사)
관심분야는 소프트웨어 재사용, 소프트웨어 제품라인 공학

강 교 철

1973년 고려대학교 통계학과 졸업(학사)
1974년 콜로라도대학교 산업공학 졸업(석사)
1982년 미시간대학교 소프트웨어공학 졸업(박사)
1982년~1984년 미시간대학교 Visiting Professor
1984년~1985년(미국) 벨 통신 연구소 Technical Staff
1985년~1987년 AT&T 벨 연구소 Technical Staff
1987년~1992년 카네기멜론대학교 Project Leader
1992년~현재 포항공과대학교 교수
2000년~2001년 카네기멜론대학교 Visiting Scientist
관심분야는 소프트웨어공학, 실시간 내장형 시스템, 소프트웨어 재사용, 소프트웨어 제품라인 공학

SaaS 환경에서 SLA 보장을 위한 명세 및 교환 방법†

(A Specification and Exchange Method for Supporting SLA in SaaS Environment)

남태우‡ (Taewoo Nam) 강태준§ (Taejun Kang) 장문수* (Moonsoo Jang) 안영민¶ (Youngmin An) 염근혁‡‡ (Keunhyuk Yeom)

요 약 클라우드 컴퓨팅 서비스를 제공하는 사업자는 이용자에게 신뢰성 있고 일관된 품질을 제공하기 위해서 SLA를 보장해야 한다. SLA(Service Level Agreement)는 서비스 사업자가 제공하는 서비스를 대상으로 가용성 등 일정한 서비스 수준을 보장하기 위해 맺는 서비스 사업자와 고객간의 계약이다. 클라우드 컴퓨팅은 다양한 클라우드 서비스의 IT 자원에 따라 IaaS, PaaS, SaaS 등으로 구분되는데 기존의 SLA는 물리적인 네트워크 환경에 대한 요소만 고려하고 있어서 제공되는 서비스의 품질 요소는 반영하기 어렵다. 본 논문에서는 SaaS 레벨에서의 SLA 명세를 위한 XML 스키마를 가지는 명세 언어와 이를 교환하기 위한 UDDI 기반의 교환 프로세스 및 아키텍처를 제안한다. 클라우드 환경에서 SaaS의 품질 요구사항은 제안한 명세 언어로 정의되고 품질 명세 저장소에 저장되며 교환 아키텍처를 기반으로 서비스 바인딩 시 교환된다.

키워드 SLA(Service Level Agreement), 서비스 수준 합의, 클라우드 컴퓨팅 서비스, SaaS

Abstract A cloud computing service provider must assure Service Level Agreement (SLA) to provide reliable and consistent quality of service to a user. The SLA is a contract between the user and the service provider that connects to assure constant level such as availability to target provided service. The cloud computing is classified into IaaS, PaaS, and SaaS according to IT resources of the various cloud service. The existing SLA is difficult to reflect quality factors of service because it only considers factors about the physical Network environment. In this paper, we suggest the UDDI-based interchange process with the architecture and the specification language having a XML schema for the SLA specification. The quality requirements of SaaS are defined by a proposed specification language in the cloud environment. It is stored in the repository of a quality specification and exchanged on during the service binding time based on the exchange architecture.

Key words SLA, Service Level Agreement, Cloud Computing Service, SaaS

† "본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학IT연구센터 육성 지원사업의 연구결과로 수행되었음" (NIPA-2013-(H0301-13-1012))

‡ 학생회원 : 부산대학교 전자전기컴퓨터공학과
kaluas@pusan.ac.kr
§ 학생회원 : 부산대학교 전자전기컴퓨터공학과
erinlily1@pusan.ac.kr
* 학생회원 : 부산대학교 전자전기컴퓨터공학과
anstn088@pusan.ac.kr
¶ 학생회원 : 부산대학교 전자전기컴퓨터공학과
yurizz91@pusan.ac.kr
‡‡ 종신회원 : 부산대학교 정보컴퓨터공학부 교수
교신저자
yeom@pusan.ac.kr

1. 서 론

최근 스마트폰, 태블릿PC, 스마트TV 등 모바일 기술혁신과 각종 스마트 기기의 이용 및 데이터의 급증으로 국민, 정부, 기업들은 그들의 업무환경에서 편리하게 이용하되 보다 빠르면서 저렴하게 사용할 수 있는 클라우드 컴퓨팅 서비스에 대한 관심이 급증하고 있다.

이런 클라우드 컴퓨팅 서비스를 제공하는 사업자는 이용자에게 신뢰성 있고 일관된 품질을 제공하기 위해서 클라우드 컴퓨팅 환경에서 SLA (Service Level Agreement)의 보장이 필수적이다. IT 서비스를 제공받는 고객과 서비스 공급자는 서로의 상호관계를 올바르게 인식해야 하며 각자의 요구사항 및 기대치에 대한 정의가 필요하다. 이에 대한 명확한 합의가 반드시 이루어져야 하며, 그 관리 방법의 하나가 서비스 수준 합의서 (SLA)이다.

클라우드 컴퓨팅 서비스는 사용한 만큼 비용을 지불하는 서비스이므로 SLA가 잘 정의되어야 서비스 제공자와 사용자가 만족하는 서비스를 이루어 나갈 수 있다.

미국 정부는 USG 클라우드 컴퓨팅 기술 로드맵을 수립하였으며[1], 높은 신뢰 수준의 서비스를 위한 기술적인 세부사항을 명시하고 있다. 실질적인 SLA 평가를 위해 높은 수준의 품질과 완성도가 요구되고, 보증, 수행 측정 등은 클라우드 서비스의 핵심 필요 요소이며 서비스 제공자 마다 다른 SLA 기준, 자원, 보장 범위 등으로 혼란을 야기할 수 있어서 공통의 용어와 정의가 SLA에서 오해의 소지를 줄일 수 있다고 분석했다. 이를 위한 참조 모델을 만드는 과정에서 NIST에서 클라우드 SLA 연구를 수행 중이긴 하나 아직 클라우드 서비스 산업 전반에 걸친 표준 SLA는 존재하지 않는다.

클라우드 컴퓨팅은 다양한 클라우드 서비스의 IT 자원에 따라 IaaS(Infrastructure as a Service), PaaS(Platform as a Service) 및 SaaS(Software as a Service)로 구분된다[2]. IaaS는 대규모 연산 능력이 필요할 경우 확장성이 풍부한 가상화된, 즉 CPU, 메모리 등과 같은 전산 자원을 제공하거나 이미지 · 동영상 등의 자료를 저장할 수 있는 스토리지 자원을 제공하는 서비스를 말한다. PaaS는 사용자가 소프트웨어를 개발할 수 있는 토대를 제공하는 서비스를 말한다. 플랫폼 서비스 사업자는 애플리케이션을 개발하는데 필요한 개발환경, 프레임워크 등의 개발 플랫폼을 제공하고, 응용서비스 개발자들은 플랫폼 서비스 사업자가 제공하는 플랫폼 상에서 IT 자원을 활용하여 새로운 애플리케이션을 만들어 사용할 수 있다. SaaS는 애플리케이션을 서비스 대상으로 하며, 응용 소프트웨어 서비스 사업자가 인터넷을 통해 소프트웨어를 제공하고, 사용자가 인터넷상에서 원격으로 접속하여 해당 소프트웨어를 활용하는 모델이다.

이와 같은 클라우드 서비스 모델 중 SaaS가 제공하는 소프트웨어 서비스는 기존 SLA의 서비스 가용성(장애 없이 서비스를 이용할 수 있는 시간의 비율) 기반의 관리 요소만으로는 제대로 그 품질에 대한 요구사항 관리가 이루어질 수 없다. 실제로 아마존, 구글, MS 등 해외 클라우드 컴퓨팅 서비스를 제공하는 기업은 SLA를 통해, 서비스 책임 범위를 명확히 하고 있으며, 일반적으로 가용성을 중심으로 품질을 보장하고 있고 제시된 가용성 수준에 미달하는 경우 고객에게 무료 서비스 형태로 위약금을 제공한다. 그러나 현재 상용화된 서비스들은 주로 IaaS나 PaaS인 스토리지나 플랫폼 서비스라서 가용성 기반의 SLA 관리가 용이하지만 SaaS의 소프트웨어가 가지는 다양한 비기능/품질 요구사항은 단순히 가용성만 가지고는 관리하기 힘들다.

따라서 본 연구를 통해 SaaS 환경에 적합한 SLA 제공을 위해 소프트웨어적인 품질 특성을 명세할 수 있는 방법을 제시하고 이를 서비스 제공자와 소비자간에 교환할 수 있는 시스템과 구조를 제안하고자 한다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 본 논문과 관련된 선행 기술에 대해 살펴보고, 3장에서는 SaaS 환경에서 SLA 명세를 위한 메타모델과 명세 방법을 설명하고, 4장에서는 SLA 명세를 저장하고 교환할 수 있는 시스템을 제안한다. 그리고 5장에서는 제안한 명세 방법을 이용한 적용사례를 보이고, 마지막으로 6장에서 결론 및 향후 연구 목표를 도출한다.

2. 관련 연구

2.1 클라우드 컴퓨팅 서비스 유형

서비스 제공자가 다양한 컴퓨팅 자원들의 공유 풀(Pool)을 구축하여 사용자는 최소한의 관리 부담과 커뮤니케이션만으로 자원을 할당 받고 인터넷을 통해 쉽게 접근할 수 있도록 하는 새로운 컴퓨팅 모델인 클라우드 컴퓨팅은 제공되는 서비스 내용과 어떤 수준까지 자원 접근 권한을 제공하는가에 따라 인프라형 서비스(IaaS), 플랫폼형 서비스(PaaS), 그리고 소프트웨어형 서비스(SaaS)로 분류할 수 있다.

IaaS는 시스템 관리자의 수작업을 최소화하고 자동화된 프로세스에 의해 처리되는 특징을 가진다. PaaS는 애플리케이션 개발을 위한 개발 및 테스트 도구, DBMS, 미들웨어, API등을 제공하고, 개발된 애플리케이션을 실행하여 서비스를 제공할 수 있는 인프라와 소프트웨어 환경을 제공하는 서비스다. 그리고 SaaS는 비즈니스 애플리케이션 및 업무지원을 위한 소프트웨어를 제공하는 서비스다. 하위 인프라형에 대한 어떤 것도 관리하거나 통제할 필요가 없는 서비스로의 소프트웨어를 의미하고, IaaS기반의 인프라형을 통해 제공되는 소프트웨어 및 애플리케이션으로 사용자별로 개별적인 개인화(Personalization)가 제공되며 독립적인 서비스 사용 환경으로 구성된다.

따라서 SaaS 환경에서 서비스 수준에 대한 협약은 보다 구체적이고 소프트웨어의 품질 특성을 보다 잘 표현할 수 있어야 한다.

2.2 SLA

SLA(Service Level Agreement)[3,4,5,6]란 서비스의 제공자와 서비스 사용자 상호간에 IT 서비스를 제공할 대상 서비스에 대해 제공자에게 최소한의 요구 수준과 측정 기준, 문제 해결 방법을 명확한 기준으로 기술하여 상호간 사전에 협의하여 결정해둔 약속이다. 즉, SLA는 협상 과정을 통해 고객인 사용자가 사업을 수행함에 있어 직접적인 영향을 미치는 서비스가 어떤 것이며, 이러한 서비스가 어떠한 수준으로 제공되어야 하는지 언급되어야 한다. 그러므로 사전에 상호 연구, 협약을 맺을 서비스, 측정할 방법, 최소 요구 수준, 서비스 수행 결과에 대한 협의된 처벌과 보상, 문제 발생시의 대안과 해결 절차 및 방법을 넣어서 상호협의 하에 명시하게 된다. 그리하여 현실적인 목표와 서비스 수준을 측정하고 산출하여 서비스 비용 산정의 근거로 삼고 이것을 정보 시스템에 자세히 기록하여 목록을 만드는 것이 목적이다.

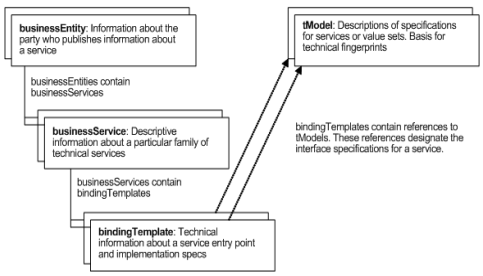
클라우드 환경에서 SLA는 고객에게 제공하는 클라우드 서비스의 품질 수준을 정량적으로 측정하고 서비스 성과를 평가, 보상하여 제공자와 사용자 간의 서비스를 보증하기 위한 품질 상세 약정이다. 현재 클라우드 서비스의 SLA로 정의되어있는 요소는 서비스 가용성, 데이터 백업 준수율 정도이다. 가용성은 예정 가동시간에 대한 실제 가동 시간의 비율이며 데이터 백업 준수율은 계획된 총 백업 건수에 대한 실제 백업 건수의 비율이다. 이런 SLA 요소로는 SaaS의 품질 특성을 반영하기 어렵다.

2.3 UDDI

UDDI(Universal Description, Discovery, and Integration)는 웹 서비스에 관한 정보를 등록, 탐색하기 위한 레지스트리로 서비스 설명과 서비스 발견에 대한 표준 스펙을 제공한다. UDDI는 각 표준과 프로토콜별로 가상의 모든 인터페이스를 사용하여 매우 유연한 서비스를 구성한다. [그림 1]은 UDDI Version 3.0.2의 표준 데이터 구조를 보인다[7].

UDDI의 각 구성요소는 XML 형태로 표현되어 저장되며, Business Entity, Business Service, Binding Template, tModel로 이루어진다. Business Entity는 제공자에 대한 정보를 담고 있으며, Business

Service는 제공자가 제공하는 웹 서비스에 대한 다양한 타입을 기술하고, Business Template은 서비스 진입점에 대한 정보를 담고 있다. 마지막으로 tModel은 웹 서비스와 어떻게 상호작용할 수 있는지에 대한 방법을 담고 있다.



[그림 1] UDDI Version 3.0.2 표준 데이터 구조

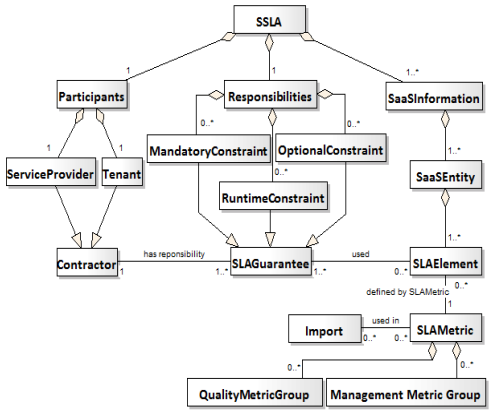
본 연구에서는 SaaS의 SLA 명세 기반 검색을 위해 tModel을 확장하여 명세 질의 처리가 가능한 SLA 명세 저장소로 이용하였다

3. SaaS SLA 명세

SaaS 환경에서 SLA 요소를 전사적으로 관리하기 위해서는 SLA 명세할 수 있는 언어가 필요하다. 본 연구에서는 이를 SSLA(SaaS Service Level Agreement)라고 명칭하고 메타모델과 XML 스키마를 제시한다.

3.1 SSLA 메타 모델

[그림 2]는 SSLA의 메타모델이다. SSLA 메타모델은 크게 세 부분으로 구성된다. SaaS 계약에 관계된 참여자들(Participants) 부분, 하나 혹은 그 이상의 SaaS에 관련된 정보(SaaS Information) 부분, 참여자들의 의무(Responsibilities) 부분이다.



[그림 2] SSLA의 메타모델

참여자들(Participants) 부분은 SaaS 환경에서 계약을 성립하는 두 가지 종류의 참여자들을 가진다. 하나는 SaaS를 제공, 관리, 모니터링 등의 의무가 있는 서비스 제공자(Service Provider)와 SaaS를 임차해서 사용하는 임차인(Tenant)이다. 이 둘은 SLA를 이용하여 서비스를 계약하는 주체이며 이를 목적으로 계약자(Contractor)로 일반화된다.

SaaS 정보(SaaS Information) 부분은 SLA를 기반으로 서비스의 오퍼레이션, 매개변수, 메트릭의 측면에서 계약 참여자들의 공통 이해를 기술한다. SaaS 정보는 하나 혹은 그 이상의 서비스 엔티티(Service Entity)를 가지는데 이는 SaaS가 단일 서비스 개체로 제공되기도 하지만 경우에 따라 멀티테넌시 보장을 위해 복합 서비스로 구성되기 때문이다. 또한 각각의 서비스 엔티티는 SLA 요소(SLA Element)에 의해 매개변수화 되어서 서술되고 이는 SLA 보증(SLA Guarantee)에서 사용된다. SLA 요소의 매개변수가 나타내는 의미는 SLA 메트릭(SLA Metric)에 의해 정의되는데 미리 정의된 품질 항목(Predefined Metric Group)의 메트릭은 메타모델로 정의되어 있고, 계약자들의 요구에 의해 추가할 수 있는 가져오기(Import)로 정의할 수 있다.

Predefined Metric Group의 메트릭 항목, 지표, 지표 값 범위는 <표 1>과 같다. Predefined Metric Group 이외의 품질 메트릭은 계약자(Contractor)의 필요에 의해서 가져오기(Import)를 통해 추가할 수 있다.

의무(Responsibilities) 부분은 그 성격에 따라 강제적으로 어떠한 상황에서라도 지켜야 하는 강제적 제약(Mandatory Constraint), 특정 외부 상황 조건에 따라서는 지켜야 하는 선택적 제약(Optional Constraint), SaaS가 가동되는 환경인 내부적인 상황 조건에 의해 결정되는 실행 시 제약(Runtime Constraint)으로 구성된다. 의무는 SLA 요소를 사용하여 계약자들 사이에 SLA보장(SLAGuarantee)으로 일반화되어 사용된다.

3.2 SSLA의 XML 스키마

SSLA는 XML 스키마로 정의된 XML 형식 기반이며 별도의 네임스페이스(saassla)를 가진다.

<표 1> SaaS SLA의 타입 정의

```
<xsd:complexType    name="SaaSSLAType">
  <xsd:sequence>
    <xsd:element name="Participants"
                 type="sassla:ParticipantsType"/>
    <xsd:element name="SaaSInformation"
                 type="saassla:ServiceDefinitionType"
                 maxOccurs="unbounded"/>
    <xsd:element name="Responsibilities"
                 type="saassla: ResponsibilitiesType"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="SSLA"
             type="saassla:SaaSSLAType"/>
```

<표 1>은 SSLA의 타입 정의이다. SaaSSLA Type의 SSLA는 SaaS SLA의 루트 요소이며 SSLA는 하나의 참여자 부분, 복수개의 SaaS 정보 부분, 하나의 의무 부분으로 구성된다. SSLA의 타입 정의를 이용하여 표2와 같이 특정 SaaS의 SLA에 대하여 최상위 구조를 작성할 수 있다.

<표 2> SSLA의 최상위 구조의 예

```
<?xml    version="1.0">
<saassla:SSLA
      xmlns:saassla="http://selab.pusan.ac.kr/saassla"
      name="BuyaBookSaaSSLA#102">
  <Participants>
  …
  </Participants>
  <SaaSInformation>
  …
  </SaaSInformation>
  <Responsibilities>
  …
  </Responsibilities>
</saassla:SSLA>
```

SSLA의 대표적인 특징인 SLA Metric 부분을 살펴보면 SLA Metric은 서비스 제공 시스템으로부터 측정되거나 다른 외부 메트릭으로부터 계산될 수 있는 서비스 속성 값에 대한 정의이다. SLA Metric은 SLA Element가 어떤 매개변수를 이용하여 어떻게 값을 측정하고 계산하는지 명시하기 위한 중요한 요소이다. SLA Metric은 complex 타입이며 메트릭 값의 형식에 대한 정보들을 찾을 수 있는 곳과 획득할 수 있는 곳 등의 모든 관련 정보를 정의한다. "Source"는 SLA Metric을 획득하고 사용하는 계약자(Contractor)에 대한 참조이며 "SLA Metric URI"는 런타임 중에 메트릭 값을 구할 수 있는 주소이다. 기술/비기술 특성에 대한 품질 요소와 외부의 참조 메트릭을 가져오기(Import) 위한 요소를 포함한다.

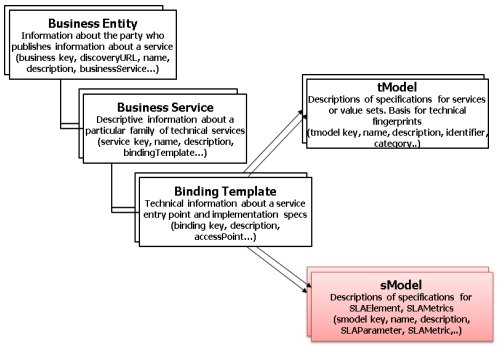
이와 같이 메타모델의 각 요소를 XML 형식으로 타입을 정의하였고 saassla라는 네임스페이스를 사용한 SSLA 정의 형식을 이용해서 SaaS의 서비스 수준 합의서를 전사적으로 명세할 수 있다.

4. SSLA 명세 교환 구조

3장의 SSLA 명세 언어를 이용하여 SaaS의 SLA를 명세하였으면 이를 교환하여 계약을 맺고 서비스를 바인딩할 수 있는 아키텍처가 필요하다.

본 연구에서는 SaaS의 단위 서비스를 플랫폼 독립적으로 구동이 가능한 웹서비스 방식의 소프트웨어로 정의하고 이런 SaaS 환경에서 SLA 명세를 등록하고 검색, 바인딩이 가능한 구조를 제안한다.

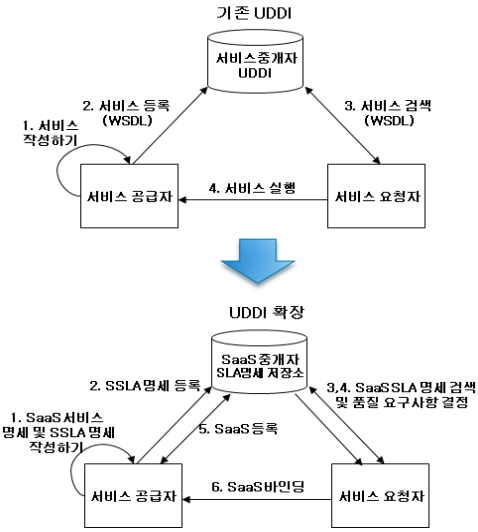
기존의 WSDL(Web Service Description Language)를 교환하기 위한 용도로 사용하던 UDDI의 표준 데이터 구조를 SSLA의 등록 및 검색 질의가 가능한 구조로 확장하였다. UDDI는 웹서비스의 등록과 검색을 위해 tModel이라는 데이터 구조를 이용한다. tModel은 웹서비스의 타입, 프로토콜, 시스템 카테고리 등 서비스 제공자나 세부 서비스가 사용하는 여러 가지 기술 정보, 분류 정보, 네임스페이스 등에 대한 메타 정보를 표현하는 객체이다. 이 tModel의 데이터 구조와 유사하지만 SSLA의 SLA 요소 (SLA Element)를 처리할 수 있도록 sModel을 추가하여 SLA 명세를 등록, 검색 질의 할 수 있는 구조로 표준 UDDI를 확장하였다. [그림 3]은 UDDI 표준 데이터 구조에 sModel을 추가하여 기능 확장한 SSLA 명세 저장소의 등록 데이터 구조이다.



[그림 3] UDDI 표준 데이터 구조 확장

SSLA 명세 저장소는 기존의 UDDI와는 달리 SLA 품질 요소 결정 단계가 추가된다.

[그림 4]에서와 같이 서비스 공급자는 품질 요소와 제공할 수 있는 품질 메트릭이 명세된 SSLA 명세서를 SSLA 명세 저장소에 저장하면 서비스 요청자는 sModel 기반의 서비스 검색을 통하여 특정 서비스의 품질 메트릭을 찾을 수 있고 이를 기반으로 자신이 원하는 서비스의 기능/비기능 요구사항을 결정하게 된다. 요구사항이 결정된 SSLA 명세서를 서비스 공급자가 채택하면 SLA 합의가 되며 서비스 공급자는 그에 따른 SaaS를 등록하고 서비스 요청자는 해당 서비스를 바인딩하게 된다.



[그림 4] SSLA 명세 저장소의 동작 단계

5. 사례 연구

사례 연구의 시나리오는 주식 시세를 알아보기 위한 SaaS를 서비스 제공자와 임차인이 SLA를 합의하기 위해 SSLA를 이용하여 명세를 작성한다. 서비스 제공자는 고객의 모든 요청에 대해 평균 응답 속도를 설정하며 지정한 응답 속도를 위반할 경우 계약자들에게 알림 메시지를 전송하는 것에 동의한다. <표 3>은 SLA 품질 특성으로 정의한 가용성, 성능, 관리 항목 등에 대한 SSLA 명세 작성의 간단한 예이다.

<표 3> SSLA 명세 작성 예

항목	SSLA 명세
가용성	<Metric name="PercentOverUtilized" type="float" unit="percentage"> <Source>%</Source> <Schedule>BusinessDay</Schedule> <Metric>utilizationTimeSeries</Metric> <Value> <LongScalar>0.96</LongScalar> </Metric>
	가용성 항목의 SLA Element는 Percent Over Utilized 라는 SLA Metric을 가지며 서비스 보장일의 96% 만큼의 사용 시간을 보장받아야 함
성능	<Metric name="averageResponseTime" type="double" unit="seconds"> <Source>ms</Source> <Metric>avgResponseTimeHost1</Metric> <Metric>avgResponseTimeHost2</Metric> </Metric>
	성능 항목의 SLA Element는 average Response Time은 SLA Metric을 가지며 단위는 "초"이고 Host1과 Host2 사이의 통신 시간을 ms로 측정 함
	<Metric name="averageResponseTimeHost1" type="double" unit="seconds"> <Source>ms</Source> <Function xsi:type="saassla:Mean" resultType="double"> <Metric>responseTimesHost1</Metric> </Metric>
	average Response Time Host1은 response Times Host1의 평균(mean)으로 측정하며 Measurement-URI로의 응답 속도로 측정함
가용성	<Action xsi:type="WSDLActionDescription-Type" name="Notification" contractorName="provider"> <WSDLFile>violationNotification.wsdl </WSDLFile> <SOAPBindingName> SOAPNotificationBinding </SOAPBindingName> <SOAPOperationName>Notify </SOAPOperationName> </Action>
	Notification.wsdl 서비스를 이용하여 장애 시간/내역을 통지 함

6. 결론 및 향후 연구

SLA는 클라우드 컴퓨팅 서비스가 비즈니스 시스템으로서의 역할을 제대로 수행해 낼 수 있을지를 객관적으로 판단할 수 있는 핵심적인 기준임에 틀림없다. 그러나 아직까지 클라우드 컴퓨팅 서비스 제공에 있어서 SLA 도입이 추세이나 이와 관련된 상세 지침이 존재하지 않을뿐더러 SaaS 환경의 SLA는 극히 초보적인 단계라서 SaaS 도입에 있어서 제약사항으로 작용하고 있다.

보다 구체적이고 체계적으로 SaaS의 SLA 명세를 위해 본 논문은 SSLA라는 XML 기반의 명세 언어를 제안하고 이를 저장하기 위한 SSLA 명세 저장소와 교환 아키텍처를 제시하였다. SSLA를 이용하여 기존의 SLA 품질 지표인 가용성이나 백업 준수률 등으로 표현하지 못하던 SaaS의 소프트웨어측면의 품질 요소를 기술할 수 있으며 SSLA 명세를 시스템적으로 관리 가능한 기반 구조를 마련하였다. 향후 SSLA 기반의 품질 컨택스트 추론 기술을 접목시켜 실시간으로 SaaS의 SLA 준수 여부를 모니터링하고 품질 추론 규칙을 통해 위반을 감지하는 관리 기술을 연구할 예정이다.

참 고 문 헌

[1] NIST, "High-Priority Requirements to Further USG Agency Cloud Computing Adoption," U.S. Government Cloud Computing Technology Roadmap Volume I, 2011.

[2] C. Gong, J. Liu, et al. "The characteristics of cloud computing," Parallel Processing Workshops (ICPPW), 2010 39th International Conference on. IEEE, pp. 275-279, 2010.

[3] H. Ludwig, A Keller, et al. "A service level agreement language for dynamic electronic services," Electronic Commerce Research 3.1-2, pp. 43-59, 2003.

[4] W. Maurer, R. Matlus, N. Frey., "Strategic Analysis Report, A Guide to Successful SLA Development and Management," Gartner Group, 2000.

[5] K. Xu, X. Zhang, et al. "Research on SLA management model in Service Operation Support System," Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing, published IEEE, pp. 4912-4915, 2009.
[6] HJ. Lee, MS. Kim, et al, "QoS Parameters to Network Performance Metrics Mapping for SLA Monitoring," The Asia-Pacific Network Operations and Management Symposium(APNOMS), pp. 42-53, 2002.
[7] Clement, Luc, et al. "UDDI Version 3.0. 2-UDDI Spec Technical Committee Draft." OASIS UDDI Spec TC, 2004.

저자 소개

남 태 우

2007년 부산대학교 정보컴퓨터공학부 졸업(학사)
2009년 부산대학교 컴퓨터공학과(석사)
2009년~현재 부산대학교 전자전기컴퓨터공학과 박사과정
관심분야: 클라우드 컴퓨팅(SaaS), 빅데이터, 상황인식 기법, 소프트웨어 아키텍쳐 등

강 태 준

2013년 부산대학교 정보컴퓨터공학부 졸업(학사)
2013년~현재 부산대학교 전자전기컴퓨터공학과 석사과정
관심분야: 클라우드 컴퓨팅, 모바일 시스템

장 문 수

2013년 동서대학교 컴퓨터정보컴퓨터공학부 졸업(학사)
2013년~현재 부산대학교 컴퓨터공학과 석사과정
관심분야: 클라우드 컴퓨팅(SaaS), 빅데이터, 소프트웨어 아키텍쳐 등

안 영 민

2010년~현재 부산대학교 정보컴퓨터공학부 학사과정
관심분야: 클라우드 컴퓨팅, 온톨로지 추론 기법

염 근 혁

1985년 서울대학교 계산통계학과(학사)
1992년 플로리다대학교(Univ. of Florida) 컴퓨터공학과(공학석사)
1995년 플로리다대학교(Univ. of Florida) 컴퓨터공학과(공학박사)
1985년~1988년 금성반도체 컴퓨터연구실 연구원
1988년~1990년 금성사 정보기기연구소 주임연구원
1995년~1996년 삼성SDS 정보기술연구소 책임연구원
1996년~현재 부산대학교 정보컴퓨터공학부 교수
관심분야: 소프트웨어 재사용, 프로덕트 라인 공학, 소프트웨어 아키텍쳐, 컴포넌트 기반 소프트웨어 개발, 적응형 소프트웨어 개발, 클라우드 컴퓨팅, 빅데이터 등

회원 가입 안내 및 회비 납부 요령

한국정보과학회 소프트웨어공학소사이어티는 회원 여러분에게 유익한 정보를 제공해 드리기 위하여 보다 충실한 내용의 논문지 발간 배포, 그리고 국제·국내 학술발표회 및 초청강연회와 단기강좌 등의 여러 가지 사업들을 추진하고 있습니다.

소프트웨어공학 소사이어티의 가입을 통해 정보 및 기술 교류, 그리고 인적 네트워크의 구성에 참여하시기를 기대합니다. 회원 가입을 위하여 아래의 회비 안내를 참고하시어 회비를 납부하시고, 다음 쪽의 입회원서를 작성하시어 아래 소프트웨어공학소사이어티 주소로 보내주시거나 팩스 또는 이메일을 통해 보내어 주시기 바랍니다.

한국정보과학회 소프트웨어공학소사이어티 연락처는 아래와 같습니다.

◈ **소프트웨어공학소사이어티**

주 소 : (우) 110-743 서울특별시 종로구 홍지문 2길 20, 상명대학교 소프트웨어대학관 G511 한국정보과학회 소프트웨어공학소사이어티 한혁수

전 화 : 02-2287-5033

팩 스 : 02-2287-0049

전자우편 : hshan@smu.ac.kr(한혁수 교수), jungwony@ajou.ac.kr(이정원 교수)

홈페이지 : http://www.sigse-kiss.or.kr/

◈ **회비 안내**

회원구분	•학생회원 : IT 분야 학과 또는 관심 있는 학생 •정회원, 종신회원 : IT 분야 종사자
가입비	학생회원, 정회원 : 20,000원, 종신회원 : 200,000원
년회비	학생회원, 정회원 : 20,000원

- 회비납부 방법

(1) 무통장입금 또는 계좌이체 후 입회원서 발송

: 계좌 번호 : 제일은행 150-20-358028 (이병정)

(2) 소사이어티 주관 학술행사 개최시, 행사장 당일 가입 및 납부 가능

개인회원용 입회원서

회원구분	학생회원 ( ) 정회원( ) 종신회원( )				
성명	한글		생년월일		
	영문				
연락처	직장전화		휴대전화		
	e-mail				
주소	직장명/부서		직급		
	직장주소	(우)			
학력	학사	년 월 - 년 월	대학교	과	
	석사	년 월 - 년 월	대학원	과	
	박사	년 월 - 년 월	대학원	과	
관심분야					

본인은 한국정보과학회 소프트웨어공학소사이어티의 취지에 찬성하여 회원으로 가입하고자 이에 입회원서를 제출합니다.

년 월 일

신 청 인: (인)

한국정보과학회 소프트웨어공학 소사이어티 회장 귀하

논문지 논문 모집 (Call for Papers)

한국정보과학회 소프트웨어공학 소사이어티에서는 매년 4회에 걸쳐 '소프트웨어공학 소사이어티 논문지'를 발간하고 있습니다. 이 논문지에는 소프트웨어공학 전반에 걸친 연구논문과 산업계 논문을 게재해 오고 있습니다. 다음과 같은 소프트웨어공학 주제에 관련된 논문을 모집하고 있으니 학계와 산업계의 여러분들의 적극적인 논문투고를 바랍니다.

◈ 논문 주제

- ○ 소프트웨어 설계 및 아키텍처
- ○ 소프트웨어 재사용 및 프로덕트라인
- ○ 요구공학
- ○ 소프트웨어 품질 및 테스트
- ○ 관리 및 프로세스
- ○ 소프트웨어 정형 기법
- ○ 서비스기반 소프트웨어 개발
- ○ 임베디드, 모바일, 웹기반 소프트웨어 개발
- ○ 기타 소프트웨어 응용 (국방, 자동차, 조선 등의 분야)

◈ 논문심사

- ○ 투고된 논문은 편집위원회에서 심사 선정하며, 필요 시 외부 심사위원을 위촉하여 심사를 합니다. 제출된 논문은 반환하지 않습니다.
- ○ 심사료 및 게재료: 없음

◈ 논문 제출

- ○ 소프트웨어공학 소사이어티의 논문지 투고 양식(http://www.sigse-kiss.or.kr/)을 사용하며, 논문의 분량은 10장으로 제한합니다.
- ○ 논문지 투고규정에 따라 작성된 심사용 논문파일은 온라인투고시스템을 통하여 투고하시기 바랍니다.

◈ 문의처 (편집위원회)

- ○ 편집이사 : 강성원 교수 (KAIST, 042-350-3512, sungwon.kang@kaist.ac.kr)
- ○ 편집이사 : 윤회진 교수 (협성대학교, 031-299-0841, hjyoon@uhs.ac.kr)
- ○ 편집이사 : 이관우 교수 (한성대학교, 02-760-5864, kwlee@hansung.ac.kr)
- ○ 편집이사 : 채흥석 교수 (부산대학교, 051-510-3517, hschae@pusan.ac.kr)
- ○ 편집이사 : 김문주 교수 (KAIST, 042-350-3543, moonzoo@cs.kaist.ac.kr)
- ○ 편집위원 : 김정아 교수 (관동대학교, 033-649-7801, clara@kwandong.ac.kr)
- ○ 편집위원 : 김현수 교수 (충남대학교, 042-821-6657, hskim401@cnu.ac.kr)
- ○ 편집위원 : 이우진 교수 (경북대학교, 053-950-6378, woojin@mail.knu.ac.kr)
- ○ 편집위원 : 박수진 교수 (서강대학교, psjdream@sogang.ac.kr)
- ○ 편집위원 : 이지현 교수 (대전대학교, jihyun30@dju.ac.kr)
- ○ 편집위원 : 최종무 교수 (단국대학교, choijm@dankook.ac.kr)
- ○ 편집위원 : 김태호 박사 (ETRI, taehokim@etri.re.kr)

투 고 요 령

1. 소프트웨어공학소사이어티 논문지에 실리는 원고는 주제 논문, 일반 논문, 산업체 기고 등으로 구분하며 다음과 같은 분야에 대하여 모집한다.
   가. 소프트웨어공학 및 그 응용분야에 대한 연구결과
   나. 강좌 및 관련 교육사항 소개 (목적, 과정, 일정, 대상, 특징)
   다. 소프트웨어 도구 및 방법론 소개 (가격, 특징, 종류, 적용사례)
   라. 소프트웨어 산업에 대한 학계, 업계의 주요 관심사
   마. 기타 관련 사항

2. 투고자는 원칙적으로 본 소사이어티의 회원으로 한다. 다만 공동 또는 초청 기고자는 예외로 한다.

3. 논문은 원칙적으로 한글로 작성한다.

4. 원고는 한글(hwp), 워드(MS Word), PDF 형식 중 하나를 택하여 A4용지에 작성하며, 그림과 표를 포함하여 10쪽 이내로 한다.

5. 논문 내용에 직접 관련이 있는 문헌에 대해서는 이들 문헌에 관련이 있는 본문 중에 참고문헌 번호를 쓰고 그 문헌을 참고문헌 난에 인용 순서대로 기술한다. 참고문헌은 학술지의 경우 저자, 제목, 학술지명, 권, 호, 쪽수, 발행 연도의 순서로, 단행본은 저자, 서명, 쪽수, 발행처, 발행 연도의 순서로 기술한다.
   [1]Cole, R., "Parallel Merge Sort", SIAM Journal of Computing, vol.17, No.4, pp.770-785, 1988.
   [2]김수형, 강명호, 조형제, 송주석. "안전하고 효율적인 침입자 역추적 시스템", 정보과학회논문지, 제25권, 제10호, pp.1123-1131, 1998

6. 논문은 소프트웨어공학 소사이어티(http://www.sigse-kiss.or.kr/)의 온라인 투고 시스템을 통해 제출한다.

7. 논문투고신청서를 반드시 작성하여 이메일(hjyoon@uhs.ac.kr)로 제출한다.

7. 원고 접수는 수시로 하며, 접수일은 온라인 접수일로 한다.

8. 기타 자세한 사항은 한국정보과학회 논문지 투고 요령을 따른다.

한국정보과학회 소프트웨어공학소사이어티 임원명단

구분		성 명	소속기관명	E-Mail
회 장		한 혁 수	상명대학교	hshan@smu.ac.kr
부회장 (기획)		권 기 현	경기대학교	khkwon@kyonggi.ac.kr
부회장 (편집)		강 성 원	KAIST	sungwon.kang@kaist.ac.kr
부회장 (학술)		홍 장 의	충북대학교	jehong@chungbuk.ac.kr
부회장 (조직)		이 병 걸	서울여자대학교	byongl@swu.ac.kr
부회장 (협력)		전 진 옥	비트컴퓨터	jojeon@bit.co.kr
운영위원회	총무이사	이 정 원	아주대학교	jungwony@ajou.ac.kr
		유 준 범	건국대학교	jbyoo@konkuk.ac.kr
	기획이사	이 병 정	서울시립대학교	bjlee@uos.ac.kr
		서 주 영	아주대학교	jyseo@ajou.ac.kr
	조직이사	백 종 문	KAIST	jbaik@kaist.ac.kr
		김 영 철	홍익대학교	bob@hongik.ac.kr
	학술이사	인 호	고려대학교	hoh_in@korea.ac.kr
		고 인 영	KAIST	iko@kaist.ac.kr
	편집이사	윤 회 진	협성대학교	hjyoon@uhs.ac.kr
		이 관 우	한성대학교	kwlee@hansung.ac.kr
		채 흥 석	부산대학교	hschae@pusan.ac.kr
		김 문 주	KAIST	moonzoo@cs.kaist.ac.kr
	홍보이사	조 은 숙	서일대학교	escho@seoil.ac.kr
		이 찬 근	중앙대학교	cglee@cau.ac.kr
		박 용 범	단국대학교	ybpark@dankook.ac.kr
	협력이사	이 세 영	NIPA	sarahlee230@gmail.com
감사		차 성 덕	고려대학교	scha@korea.ac.kr
		이 상 은	NIPA	selee@nipa.kr
자문위원회		강 교 철	포항공과대학교	kck@postech.ac.kr
		권 용 래	KAIST	kwon@cs.kaist.ac.kr
		성 기 수	KISTI 고문	Kss13@truefriend.com
		배 두 환	KAIST	bae@se.kaist.ac.kr
		신 규 상	ETRI	gsshin@etri.re.kr
		양 승 민	숭실대학교	smyang@ssu.ac.kr
		우 치 수	서울대학교	wuchisu@selab.snu.ac.kr
		이 경 환	중앙대학교	kwlee@object.cau.ac.kr
		이 단 형	KAIST	danlee@cs.kaist.ac.kr
		정 기 원	숭실대학교	chong@comp.ssu.ac.kr
		박 수 용	서강대학교	sypark@sogang.ac.kr
		황 선 명	대전대학교	sunhwang@dju.kr
		전 진 옥	비트컴퓨터	jojeon@bit.co.kr
		김 수 동	숭실대학교	sdkim777@gmail.com
		이 긍 해	항공대학교	khlee@kau.ac.kr
		최 병 주	이화여자대학교	bjchoi@ewha.ac.kr

구분	성 명	소속기관명	E-Mail
이사	김 정 아	관동대학교	clara@kwandong.ac.kr
	남 영 광	연세대학교	yknam@yonsei.ac.kr
	박 수 진	서강대학교	psjdream@sogang.ac.kr
	염 근 혁	부산대학교	yeom@pusan.ac.kr
	오 재 원	카톨릭대학교	jwoh@catholic.ac.kr
	윤 희 병	국방대학원	hbyoon37@hanmail.net
	이 우 진	경북대학교	woojin@knu.ac.kr
	이 은 석	성균관대학교	leees@skku.edu
	이 석 원	아주대학교	leesw@ajou.ac.kr
	이 은 주	경북대학교	ejlee@knu.ac.kr
	전 태 웅	고려대학교	jeon@korea.ac.kr
	정 인 상	한성대학교	insang@hansung.ac.kr
	최 승 훈	덕성여자대학교	csh@duksung.ac.kr
	최 종 무	단국대학교	choijm@dankook.ac.kr
	최 호 진	KAIST	hojinc@kaist.ac.kr
	현 창 문	탐라대학교	cmhyun@tnu.ac.kr
	계 승 교	삼성SDS	seankae@samsung.com
	권 경 룡	국방기술품질원	ka-ja17@hanmail.net
	권 원 일	STA컨설팅	wonil@softwaretesting.co.kr
	김 상 기	현대자동차	sangkikim@hyundai-motor.com
	김 진 태	SEEG	jtkim@swexpertgroup.com
	민 경 오	LG전자	davidmin@lge.com
	민 상 윤	솔루션링크	sang@sol-link.com
	박 복 남	핸디피엠지	pbnkhb@hitel.net
	박 찬 규	국방SW산학연합회	milspark@hotmail.com
	배 현 섭	슈어소프트테크	hsbae@suresofttech.com
	손 세 창	인천공항공사	scsohn@airport.kr
	손 진 규	삼성탈레스	Jinkyu.son@samsung.com
	신 석 규	SW시험인증센터	skshin@tta.or.kr
	양 상 욱	KAI	sangyang@koreaaero.com
	유 영 수	현대엠앤소프트	ysyoo@hyundai-mnsoft.com
	윤 태 권	한국SW기술진흥협회	tkyune@empal.com
	윤 형 진	케피코	Hyoungjin.yoon@kefico.co.kr
	이 근	삼성전자	gskeun.lee@samsung.com
	이 성 남	방위사업청	dapalee@korea.kr
	이 우 복	삼성전자	woobok.yi@samsung.com
	이 장 수	한국원자력연구원	jslee@kaeri.re.kr
	장 주 수	모아소프트	jsjang@moasoftware.co.kr
	정 연 대	N3SOFT	ydchung@n3soft.co.kr
	조 병 인	국방과학연구소	chobyun@dreamwiz.com
	조 상 윤	다한테크	sycho@dahan.co.kr

2012~2013 소프트웨어공학소사이어티 논문지 편집위원회

소프트웨어공학소사이어티 논문지 제26권 제2호 (통권 98호)

발 행 일 ॥ 2013년 6월 30일
발 행 인 ॥ 한혁수
편 집 인 ॥ 강성원
발 행 처 ॥ 사단법인 한국정보과학회 소프트웨어공학소사이어티
연 락 처 ॥ 서울특별시 종로구 홍지문 2길 20, 상명대학교 소프트웨어 대학관 G511
한국정보과학회 소프트웨어공학소사이어티 한혁수
전 화 : 02-2287-5033, 팩 스 : 02-2287-0049
전자우편 : hshan@smu.ac.kr(한혁수 교수), jungwony@ajou.ac.kr(이정원 교수)
홈페이지 : http://www.sigse-kiss.or.kr/
인 쇄 처 ॥ (주)참기획 (전화 : 042-861-6380, 팩스 : 042-861-6381)