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

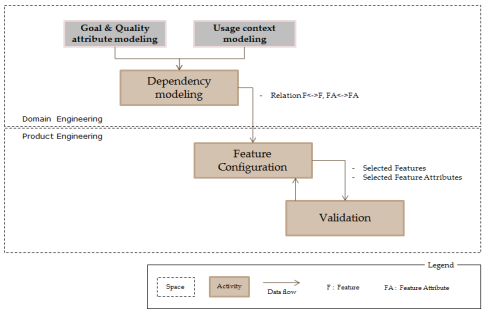
본 연구에서 제안하는 휘처 구성 방법은 소프트웨어 제품라인 공학에서 제품의 목표를 달성하기 위해 휘처와 휘처 속성의 구성 조합을 찾는 방법이다. 이 장에서는 휘처 구성 방법을 위해 제품 목표 달성에 영향을 주는 휘처 속성을 고려한 휘처 모델을 재정의하고, 이에 기반한 휘처 구성 방법에 대한 프로세스를 정의하고 각 활동이 어떻게 진행되는지를 명세할 것이다.

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

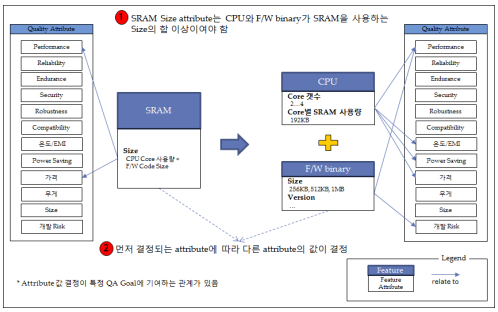
휘처 속성은 제품의 비지니스, 품질 목표 달성과의 관련성을 가지고 있다. 예를 들어 노트북 도메인에 있어서, 노트북 개발의 목표는 가격과 같은 비지니스 목표와 성능, 신뢰성, 견고성 등의 품질 목표가 경쟁사 대비 뛰어난 것이다. 이런 노트북을 개발하기 위해서는 CPU(Intel, AMD), 메모리, 메인보드, 저장 디스크(HDD, SSD), 파워 서플라이, BIOS, OS(리눅스, 윈도우즈, 맥) 등의 휘처와 CPU의 Core갯수, 메모리의 크기, 저장 디스크의 크기, 파워 서플라이의 용량 등의 휘처 속성을 제품의 목표를 달성할 수 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.

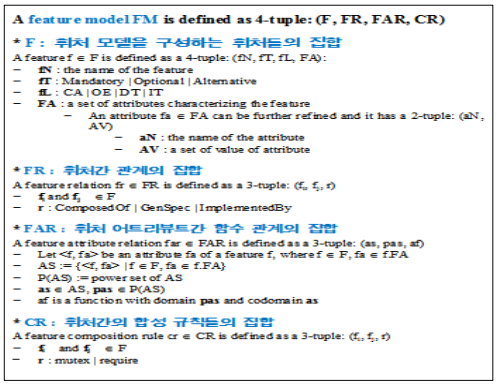
[그림 2]에서 살펴본 것 처럼 휘처 속성들 간에는 함수적 관계가 존재하는 것을 알 수 있다. 또한 이 함수적 관계를 통해 휘처 구성시에 휘처 속성 값을 결정하는데 영향을 주는 것을 알 수 있다.

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)

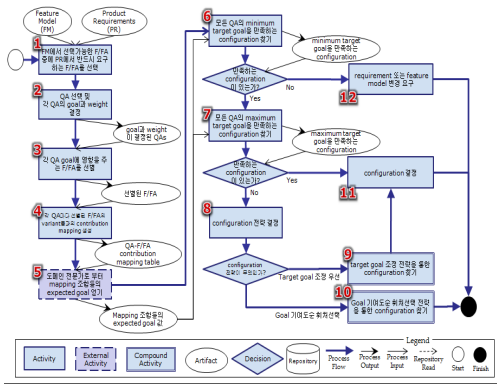
휘처 속성 관계 far의 요소인 “as”는 위에서 정의된 AS의 한 원소이고, “pas”는 AS의 멱집합 (power set)의 한 원소이다. 마지막으로 “af”는 “pas”를 domain으로 “as”를 codomain으로 가지는 함수로 정의하였다.

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

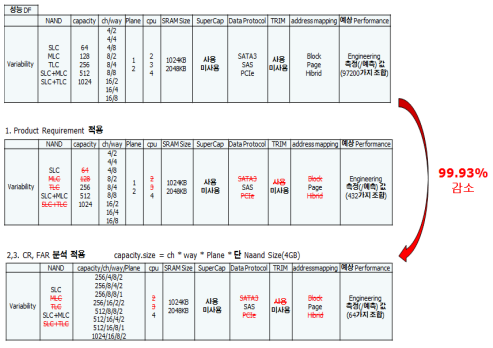
성능 QA

	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.

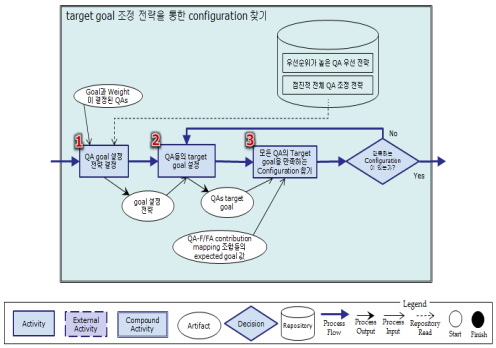
11. "configuration 결정" 활동은 7번 활동에서 모든 품질 속성의 Target Quality Goal을 만족하는 휘처 구성 조합이 있을 경우나 9번 활동이 완료되었을 경우에 진행된다. 본 활동이 진행되는 단계에서는 모든 품질의 Target Quality Goal을 만족하는 조합이 있고, 제품 개발 목표를 달성

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

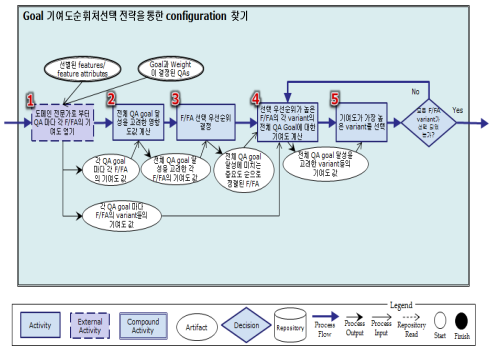
[그림 9] Target Goal 조정 전략의 세부 활동

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

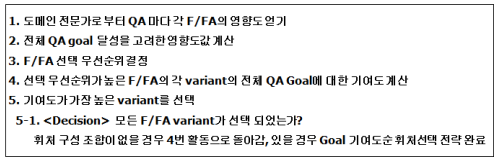
1. "QA goal 설정 전략 결정" 활동은 제품 품질 속성의 Target Quality Goal 속성값을 조정하는 세부 전략을 결정하는 역할을 한다. 세부 전략에는 '우선순위가 높은 QA 우선 전략', '점진적 전체 QA 조정 전략' 등이 있고 전략이 결정되면 이후 활동에서 저장소에 보관되어 있는 해당 전략의 알고리즘을 사용하게 된다.

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

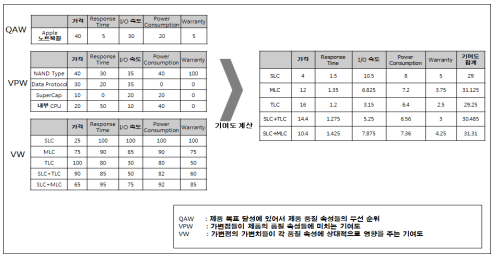
4. "선택 우선순위가 높은 F/FA의 각 variant의 전체 QA Goal에 대한 기여도 계산" 활동은 이전 활동의 가변점들 간의 우선 순위 결과를 입력으로 받아서 우선 순위가 가장 높은 가변점의 가변치를 선택하기 위해서 각 가변치들이 제품 전체 품질 속성에 미치는 기여도를 계산하는

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.

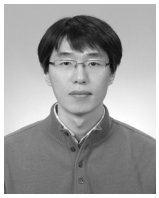
[21] D. Benavides, A. Ruiz-Cort´es, and P. Trinidad. Automated Reasoning on Feature Models. Proc. Int'l Conf. on Advanced Information Systems Engineering, 2005.

[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