http://dx.doi.org/10.7236/JIIBC.2013.13.2.83
JIIBC 2013-2-12

소프트웨어 글로벌화를 위한 제품 라인 아키텍처 설계 기술

Design Technology of Product Line Architecture for Software Globalization

이관우*

Kwanwoo Lee

요 약 소프트웨어 글로벌화란 다양한 나라 및 문화에 적합한 소프트웨어를 쉽게 체계적으로 개발할 수 있도록 지원하는 활동이다. 지금까지 소프트웨어 글로벌화를 위한 노력은 주로 소프트웨어 구현 기술 및 도구 개발에 초점을 맞추어왔다. 하지만 다른 나라 및 문화에 판매되는 글로벌 소프트웨어 제품은 다양한 환경 차이에 의해 요구사항이 다르게 되고, 이러한 요구사항의 차이는 상이한 아키텍처 설계를 유도하게 된다. 본 논문에서는 이러한 아키텍처 설계 지식을 체계화된 지식 모델로 정의하고, 이러한 지식 모델을 작성하고 이용하는 방법론을 제안한다. 또한 디지털 데이터 방송용 셋톱박스 미들웨어 시스템의 아키텍처 설계에 제안된 모델과 방법론을 적용하여 타당성을 평가한다.

Abstract Software globalization is the supporting process of adapting computer software to different country and culture. Most efforts for software globalization have been focused on the development of software implementation techniques or tools. However, global software sold worldwide has different requirements implied by different contexts, and the difference of requirements may derive different architectural design. In this paper, we define such architectural design knowledge as knowledge models and propose a method for developing and using the knowledge models for software globalization. Also we use set-top box middleware systems for digital data broadcast to validate the applicability of the proposed models and methods.

Key Words : Software globalization, product line architecture, design knowledge model, architectural design method, digital data broadcast.

Ⅰ. 서 론

소프트웨어 글로벌화 (Globalization) [1]란 지역 및 나라에 따라 다양한 언어 및 문화에 적합한 소프트웨어를 쉽게 체계적으로 개발할 수 있도록 지원해 주는 활동으로, 자사의 소프트웨어 제품을 세계 시장에 판매하거나 납품하는 소프트웨어 기업에게는 매우 중요한 도전 과제 중에 하나이다. 이는 크게 국제화 (Internationalization)와 지역화 (Localization) 활동으로 구분된다. 국제화란 다양한 언어 및 문화 환경에 쉽게 적응될 수 있

*정회원, 한성대학교 정보시스템공학과
접수일자: 2013년 2월 15일, 수정완료 2013년 3월 15일
게재확정일자: 2013년 4월 12일

Received: 15 Feburary 2013/ Revised: 15 March 2013 /
Accepted: 12 April 2013
*Corresponding Author: kwlee@hansung.ac.kr
Dept. of Information Systems Engineering, Hansung University, Korea

도록 일반화된 소프트웨어 제품을 설계하는 활동을 의미하고, 지역화란 특정 언어 및 문화 환경에 적합하도록 소프트웨어 제품을 맞추는 활동을 의미한다.

지금까지 소프트웨어 글로벌화를 위한 노력은 주로 언어 및 로케일의 변환을 위해서 코드와 리소스를 분리하고 이를 나중에 결합시킬 수 있는 소프트웨어 구현 기술 및 도구 개발[2, 3]에 초점을 맞추어왔다.

하지만 실제로 다른 지역 및 나라에 판매되는 소프트웨어 제품은 다양한 환경 차이에 의해 요구사항이 다르게 되고, 이러한 요구사항의 차이는 상이한 아키텍처 설계[7]를 유도하게 된다.

따라서 본 논문에서는 소프트웨어 글로벌화를 지원하기 위한 아키텍처 설계 기술에 초점을 맞추고자 한다. 특히, 소프트웨어 제품라인공학의 핵심 기술이 가변성 분석 기술과 이를 통한 제품라인 아키텍처 설계 기술이라고 할 수 있으므로, 소프트웨어 제품라인공학의 핵심 기술을 소프트웨어 글로벌화의 이슈에 맞게 차용하는 접근 방법을 택할 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구의 기반이 되는 핵심 개념에 대해서 간략히 설명하고, 이를 바탕으로 3장에서는 소프트웨어 글로벌화를 위한 아키텍처 설계 지식 모델링을, 4장에서는 제품라인 아키텍처 설계 방법을 차례대로 설명한다. 5장에서는 사례연구를 통해서 적용가능성을 평가하고, 6장에서 결론 및 향후 연구방향에 대해서 토의한다.

II. 기반 개념

1. 제품라인과 휘처 모델

제품라인 (Product Line)은 유사한 특징을 지닌 제품들의 집합을 의미한다. 가령, 디지털 TV 제품라인은 다양한 디지털 TV 제품들의 집합을 의미한다. 소프트웨어 글로벌화 관점에서 본 제품라인의 의미는 소프트웨어 글로벌화를 지원하기 위한 대상 제품의 범위를 말한다.

소프트웨어 글로벌화를 지원하기 위해서는 다양한 나라에서 판매 및 사용될 제품 간의 공통점과 차이점을 이해하는 것이 필수적이다. 휘처모델[4, 6]은 제품라인공학[5]에서 제품라인 범위 내의 제품들 간의 공통성과 가변성을 표현하고 관리하기 위해서 사용되는 대표적인 모델이다. 여기서 휘처는 제품 간의 공통성과 가변성을 나타내는 임의의 단위를 의미한다. 예를 들면, 자동차 제품라인은 여러 자동차 제품들의 집합으로 자동차 제품마다 구별되는 특징이 휘처가 될 수 있다. 가령 에어컨은 자동차 제품에 따라 포함될 수도 있고 아닐 수도 있는 특징이므로 이는 자동차 제품라인의 휘처가 된다.

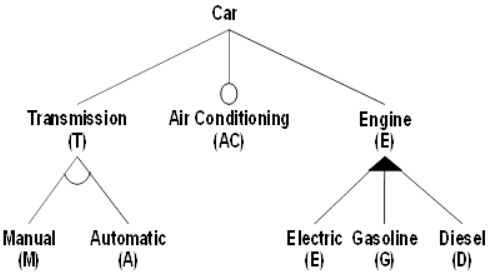

그림 1. 휘처모델의 예
Fig. 1. Feature Model Example

휘처 모델에서는 제품라인 내의 공통적인 특성을 필수 (Mandatory) 휘처로 표현하고, 제품 간의 구별되는 차이점을 선택적 (Optional) 휘처, 택일적 (Alternative) 휘처, 다중선택 (Multiple Choice) 휘처로 표현한다.

예를 들어 그림 1에서 “Transmission”은 필수 휘처이고, “Air Conditioning”은 선택적 휘처이며, “Manual”과 “Automatic”은 택일적 휘처이다. 필수 휘처는 제품들 간 공통적인 특성을 나타내는 것이고, 선택적 휘처는 제품에 따라서 포함되거나 포함되지 않는 휘처로서 그림 1에서 보는 바와 같이 동그라미로 표현된다. 택일적 휘처는 여러 휘처들 중에서 반드시 하나만 선택돼야 함을 나타내는 것으로, 표현 방식은 트리 사이의 반원으로 표현된다. 한편, “Electric”, “Gasoline”, “Diesel”은 다중선택 휘처로서 이 들 중에 적어도 하나가 선택되어야 함을 나타낸다.

2. 제품라인 아키텍처와 가변성

제품라인 아키텍처 (Product Line Architecture)는 제품라인 범위 내의 모든 소프트웨어 제품을 위한 아키텍처를 의미하며, 제품라인 범위 내의 모든 제품에 공통인 부분과 제품에 따라 차이가 나는 가변 부분으로 구성된다. 따라서 제품라인 아키텍처는 제품에 따라 달라지는 가변부에 대한 명확한 표현과 관리가 특징이라고 볼 수 있다.

제품라인 아키텍처의 가변성은 가변점(Variation Point) 관점에서 표현된다. 가변점은 아키텍처에서 변화 요소가 있는 지점을 의미한다. 가변점의 종류는 선택(optional), 택일(alternative), 선택적-택일 (optional-alternative) 중에 하나로 표현된다. 선택 가변점는 해당 아키텍처 요소가 선택적으로 포함될 수 있는 것을 의미하고, 택일 가변점은 주어진 여러 아키텍처 요소 중에서 단 하나만 해당 가변점에 결합될 수 있는 것으로, 선택적-택일 가변점는 여러 아키텍처 요소 중에 하나만 가변점에 결합되거나 하나도 가변점에 결합되지 않는 경우를 의미한다.

이와 같은 제품라인 아키텍처 가변성은 요구사항의 가변성으로부터 유도된다. 소프트웨어 글로벌화를 지원하기 제품라인 아키텍처를 설계하기 위해서는 아키텍처 설계에 영향을 미치는 요구사항의 가변성 추출이 가장 먼저 선행되어야 할 작업이다. 본 논문의 핵심 개념은 아키텍처 설계에 영향을 미치는 요구사항의 가변성은 제품이 설치되고 운영되는 글로벌 소프트웨어 제품 환경의 가변성과 밀접한 관련이 있다는 것이다. 제품 환경 요소와 요구사항과의 관계에 대한 자세한 사항은 다음 절에서 이어진다.

3. 제품 환경 요소와 요구사항과의 관계

글로벌 소프트웨어의 경우 다른 지역에 팔리는 제품에 따라 요구사항의 차이가 있을 수 있는데, 이는 각 제품이 처한 환경 요소의 차이에 의해서 유도될 수 있다. 제품 환경 (Context) 요소는 크게 사회적 환경, 법적인 환경, 물리적 환경, 사용자 환경, 비즈니스 환경으로 구분될 수 있고, 이들 각각에 대한 구체적인 설명은 아래와 같다.

사회적 환경은 문화 및 종교적 환경으로 세분화되며, 문화적 종교적 특수성에 대한 분석을 통해 새로운 요구사항을 찾을 수 있다. 가령, 유대교에서는 안식일에 버튼을 누르는 행동을 제한하므로, 유대인들 사회에서 사용되는 엘리베이터는 버튼을 누르지 않고도 매 층마다 자동으로 정지하는 기능을 반드시 제공해야 한다.

법적인 환경은 표준 및 법률에 의해 규정된 사항으로서, 설계에 영향을 줄 수 있는 요소가 있을 수 있다. 가령, 엘리베이터의 경우 각 나라마다 지켜야 하는 안전 코드 (Safety Code)가 있어서, 이러한 법률로 인해, 사용되는 하드웨어의 규격 및 소프트웨어 기능이 제한을 받거나 요구되기도 한다.

물리적 환경은 제품이 사용되는 물리적 특성 및 위치 등을 나타낸다. 가령, 엘리베이터 제품의 경우 엘리베이터가 배치되어 운영되는 건물의 층수는 물리적 환경을 나타내는 한 요소이다. 이러한 물리적 환경요소는 엘리베이터의 용량 및 속도에 영향을 미친다. 예를 들면, 층수 높을수록 많은 용량과 고속의 엘리베이터가 필요하고 층수가 낮을수록 그 반대가 된다.

사용자 환경은 사용자 혹은 사용자 그룹의 성향 및 특성을 나타내는 것으로, 이에 따라 제품의 요구되는 기능 상이할 수 있다. 예를 들면, 엘리베이터의 경우 사무실 근무자들이 주로 사용하는 엘리베이터와 방문고객이 사용하는 엘리베이터의 경우 사용자의 엘리베이터 이용 패턴이 상이하므로, 이에 따라 엘리베이터 대기 시간에 대한 요구사항이 다를 수 있다.

비즈니스 환경도 제품 설계에 영향을 주는 요구사항과 관련이 있다. 가령, 소형 아파트를 위한 엘리베이터 시장은 저가의 엘리베이터를 요구하므로, 저가의 장치를 사용하거나 높은 구현 비용이 드는 부가적인 휘처를 배제할 것이다

이와 같이, 환경적 요소에 대한 이해는 아키텍처 설계에 영향을 주는 요구사항의 도출을 위해 필수적인 요소이다.

Ⅲ. 소프트웨어 글로벌화를 위한 아키텍처 설계 지식 모델링

앞서 언급하였듯이, 제품 환경 요소, 요구사항, 아키텍처는 상호 밀접한 관계를 가지고 있다. 즉, 제품 환경 요소의 가변성은 요구사항의 가변성을 유도하고, 요구사항의 가변성은 아키텍처 설계의 가변성에 영향을 미친다. 이와 같은 아키텍처 설계 지식은 다양한 나라에 판매될 글로벌 소프트웨어의 개발 시에 체계적으로 관리될 필요가 있다.

본 장에서는 소프트웨어 글로벌화를 지원하기 위한 아키텍처 설계 지식 모델을 정의한다. 그림 2에 나타난 다양한 제품 환경 (Product Context) 가변성 모델과 요구사항 (Requirement) 가변성 모델은 제품라인 공학에서 대표적으로 사용되는 휘처 모델을 이용하고, 아키텍처의 가변성은 제품라인 아키텍처 모델을 이용하여 정

의한다. 또한 이들 간의 대응 관계를 각각 독립적인 모델로 정의한다.

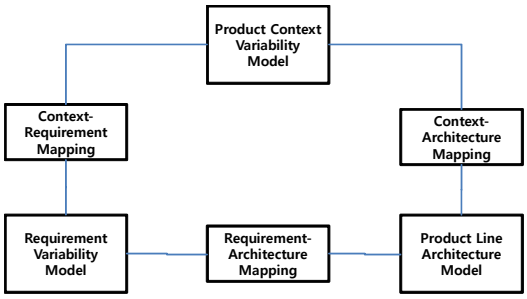

그림 2. 아키텍처 설계 지식 모델
Fig. 2. Architectural Design Knowledge Model

1. 제품 환경 가변성 모델

제품 환경의 가변성은 도메인 내의 제품들이 사용되고 운영되는 환경의 가변성을 나타낸다. 제품 환경의 가변성은 앞서 언급한 사회적 환경, 법적인 환경, 물리적 환경, 사용자 환경, 비즈니스 환경 관점에서 구체화될 수 있다.

이러한 제품 환경의 가변성은 휘처모델을 이용하여 표현가능하다. 다만 제품 환경 가변성을 나타내는 휘처 모델의 휘처는 II.3절에서 언급한 제품 환경요소를 나타낸다.

2. 요구사항 가변성 모델

기존의 휘처 모델은 주로 기능 요구사항에 초점을 두어 가변성을 표현하였다. 하지만 아키텍처 설계에는 기능 요구사항뿐만 아니라 비기능 요구사항도 중요한 영향을 미치므로, 비기능 요구사항의 가변성도 휘처 모델로 표현할 수 있어야 한다.

품질속성은 비기능 요구사항을 나타내는 추상적인 표현이라고 볼 수 있다. 따라서 추상적인 품질속성은 좀 더 구체적인 품질속성시나리오나 서브-품질속성으로 상세화 될 수 있다. 만약, 어떤 서브-품질속성이 특정한 품질속성을 만족시키는데 항상 요구된다면, 서브-품질속성을 품질속성 휘처의 필수적 서브휘처로 모델링하고, 그렇지 않다면, 선택적 서브휘처로 모델링하면 된다. 또한, 서브-품질속성들 중에서 하나 혹은 일부가 부모 품질속성을 위해서 필요한 경우에는 택일 혹은 다중휘처로 모델링한다.

3. 제품라인 아키텍처 가변성 모델

소프트웨어 구조를 나타내는 아키텍처 모델은 아키텍처 요소들의 집합, 아키텍처 요소 들 간의 연결 관계로 정의된다. 아키텍처 요소는 추상화 계층에 따라 시스템, 서브시스템, 컴포넌트 등으로 분류된다. 즉, 시스템은 서브시스템으로 분해될 수 있고, 서브시스템은 또다른 서브시스템 혹은 컴포넌트로 분해되어 계층적인 구조를 나타낸다. 각 아키텍처 요소는 다른 아키텍처요소와의 연결 관계를 통해서 이들 간의 상호작용을 나타낸다.

제품라인 아키텍처 모델의 가변성은 앞서 언급하였듯이, 가변성이 있는 아키텍처 요소를 가변점으로 표현한다. 본 논문에서는 아키텍처 모델을 UML 컴포넌트 모델로 표현하고, 가변점인 아키텍처 요소는 해당 UML 모델 요소의 스테레오타입으로 표현한다.

4. 가변성 모델 간의 대응관계

본 절에서는 그림 2에 나타난 환경-요구사항 대응, 환경-아키텍처 대응, 요구사항-아키텍처 대응에 대해서 차례대로 정의한다.

먼저, 제품 환경 휘처와 요구사항 휘처의 대응은 특정한 제품 환경에서는 특정한 요구사항이 요구된다는 도메인 지식을 나타낸 것이다. 이는 다음과 같이 정의된다.

정의 1 환경-요구사항 대응

PCVM을 제품 환경 가변성을 나타내는 모델이라 하고, RVM을 요구사항의 가변성을 나타내는 모델이라 하자. 또한 CF를 PCVM에 정의된 제품 환경 휘처의 집합이라 하고, RF를 RVM에 정의된 요구사항 휘처의 집합이라고 하자. 이 때, PCVM과 RVM 간의 대응관계 M_{CF-RF}는 $2^{CF} \times 2^{RF}$의 관계로 정의된다. 즉, M_{CF-RF} = {(C,R)| 제품 환경 휘처 집합 C⊆CF는 아키텍처 드라이버 휘처 집합 R⊆RF를 요구한다}.

제품 환경 휘처와 아키텍처의 대응은 특정한 제품 환경 조건이 특정한 아키텍처 설계 결정에 영향을 준다는 도메인 지식을 나타낸 것이다. 즉, 이는 제품 환경 휘처가 아키텍처 요소의 선택에 영향을 미치는 경우를 나타낸다.

정의 2 환경-아키텍처 대응

PCVM을 제품 환경 가변성을 나타내는 모델이라 하

고, AVM을 제품라인 아키텍처 모델이라 하자. 또한 CF를 PCVM에 정의된 제품 환경 휘처의 집합이라 하고, AE를 AVM에 정의된 아키텍처 요소의 집합이라고 하자. 이 때, PCVM과 AVM 간의 대응관계 M_{CF-AE}는 $2^{CF}\times 2^{AE}$의 관계로 정의된다. M_{CF-AE} = {(C,RAE)| 제품 환경 휘처 집합 C⊆CF는 아키텍처 요소 집합 RAE⊆AE의 선택을 요구한다}

마지막으로 특정한 요구사항은 특정한 아키텍처 요소의 선택에 영향을 준다. 즉 어떤 아키텍처 요소는 한 요구사항에 긍정적인 영향을 주지만, 다른 요구사항에 부정적인 영향을 줄 수가 있다. 따라서 요구사항과 아키텍처와의 대응은 다음과 같이 정의된다.

정의 3 요구사항-아키텍처 대응

AVM을 아키텍처 가변성을 나타내는 모델이라 하고, RVM을 요구사항의 가변성을 나타내는 모델이라 하자. 또한 RF를 RVM에 정의된 요구사항 휘처의 집합이라 하고, AE를 AVM에 정의된 아키텍처 요소의 집합이라고 하고, R을 강한긍정(++), 긍정(+), 부정(-), 강한부정(--)의 값을 가지는 집합이라 하자. 이 때, RVM과 AVM 간의 대응관계 M_{RF-AF}는 RF× AE× R의 관계로 정의된다. M_{RF-AF}= {(req,ae,r)| 아키텍처 요소 ae는 요구사항 req 의 달성에 r의 영향을 준다.}

다음 장에서는 이러한 아키텍처 설계 지식 모델을 어떻게 작성하고, 활용하는 가에 대한 전반적인 절차에 대해서 설명한다.

Ⅳ. 소프트웨어 글로벌화를 위한 개발 방법론

앞서 언급했듯이, 소프트웨어 글로벌화를 위한 소프트웨어 개발은 크게 국제화와 지역화 활동으로 구분된다. 국제화 활동은 여러 지역 및 나라에 쉽게 적응 및 재구성 가능한 소프트웨어를 개발하는 활동이고, 지역화 활동은 국제화 활동 과정에서 나온 아키텍처를 바탕으로 특정 지역 및 나라에 적합한 소프트웨어를 결정하는 활동을 말한다.

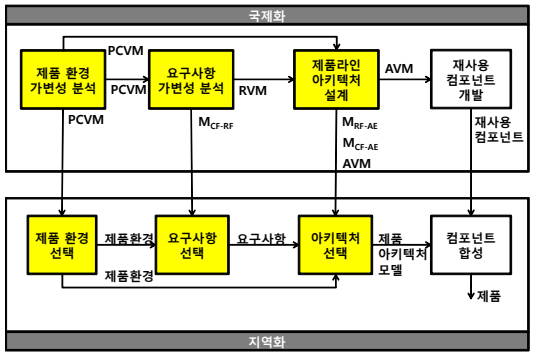

그림 3. 방법론 활동
Fig. 3. Method Activities

그림 3에서 보는 바와 같이, 국제화 활동은 제품 환경 가변성 분석, 요구사항 가변성 분석, 제품라인 아키텍처 설계, 재사용 컴포넌트 개발로 세분화 된다. 제품 환경 가변성 분석에서는 다양한 지역 및 나라에 따른 제품 환경 요소의 가변성을 분석하여, 제품 환경 가변성 모델 (PCVM)을 생성한다. 요구사항 가변성 분석에서는 다양한 제품 환경의 조합에 따라 요구되는 다양한 기능 및 비기능 요구사항을 도출하고, 요구사항들의 가변성을 요구사항 가변성 모델 (RVM)로 정의한다. 제품 환경과 요구사항 간의 상관관계는 환경-요구사항 대응 (M_{CF-RF})으로 정의된다. 제품라인 아키텍처 설계에서는 요구사항으로부터 도출되는 아키텍처 요소와, 제품 환경으로부터 영향 받는 아키텍처 요소들을 추출하여 이들 요소의 가변성을 제품라인 아키텍처 모델 (AVM)로 정의한다. 그리고 요구사항과 아키텍처 요소와의 상관관계와, 제품 환경과 아키텍처 요소와의 상관관계는 각각 요구사항-아키텍처 대응 (M_{RF-AE}), 환경-아키텍처 대응(M_{CF-AE})으로 정의된다. 마지막으로 재사용 컴포넌트 개발은 제품라인 아키텍처 모델을 바탕으로 글로벌 소프트웨어를 위한 구체적인 코드 컴포넌트를 개발하는 과정이다. 본 논문에서는 아키텍처 설계 기술에 초점을 두므로, 앞의 세 단계에 대해서만 초점을 두어 설명한다.

한편, 지역화 활동은 제품 환경 선택, 요구사항 선택, 아키텍처 선택, 컴포넌트 합성으로 세분화된다. 제품 환경 선택은 제품 환경 가변성 모델 (PCVM)을 바탕으로 특정 나라의 제품을 위해 필요한 환경 요소를 선택하는 과정으로, 이의 결과는 특정 나라에서 요구되는 제품 환경 집합이다. 요구사항 선택은 앞 단계에서 결정된 제품 환경 집합과 환경-요구사항 대응을 바탕으로 요구되는

요구사항을 도출하는 과정이다. 아키텍처 선택은 앞 단계에서 결정된 제품 환경, 요구사항, 환경-아키텍처 대응, 요구사항-아키텍처 대응을 바탕으로 특정한 제품 설계를 위한 아키텍처를 도출하는 과정이다. 마지막으로 컴포넌트 합성은 도출된 아키텍처를 바탕으로 재사용 컴포넌트를 적응시켜 구체화한 제품을 생성하는 과정이다.

V. 사례 연구 및 평가

본 장에서는 앞에서 설명한 전반적인 아키텍처 설계 과정을 좀 더 자세히 설명하고, 제안된 방법의 적용 가능성을 평가하기 위해서, 디지털 데이터 방송을 위한 셋톱박스 미들웨어 시스템을 이용한다. 이 시스템은 국내뿐만 아니라, 미주, 유럽 등 전 세계의 방송 사업자 및 시장 요구를 만족시키기 위해서 적응되어야 한다. 본 장에서는 이 시스템을 이용하여 소프트웨어 글로벌화를 지원하기 위한 아키텍처 설계 기술의 적용 가능성을 평가하고자 한다.

1. 제품 환경 가변성 분석

제품 환경 가변성 분석은 셋톱박스 미들웨어 시스템이 설치되고 운영되는 환경의 가변성을 사회적 환경, 법적인 환경, 물리적 환경, 사용자 환경, 비즈니스 환경 관점에서 분석하여 모델화한 것이다.

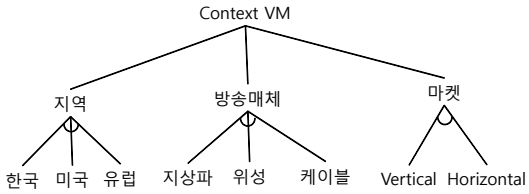

그림 4. 제품 환경 가변성 모델
Fig. 4. Product Context Variability Model

그림 4에서 보는 바와 같이, 지역 및 방송매체는 물리적 환경, 마켓은 비즈니스 환경 관점에서 찾아진 환경 가변요소이다. 소프트웨어 글로벌화를 위해 고려하고 있는 제품은 지역적으로 한국, 미국, 유럽 중에 하나를 지원하므로, 택일적 가변성으로 모델링하였다. 셋톱박스 미들웨어가 지원하는 방송매체도 지상파, 위성, 케이블 중 하나만을 지원할 수 있으므로, 택일적 가변성으로 모델링되었다. 마지막으로 제품이 팔리는 마켓의 특징은 Vertical 마켓과 Horizontal 마켓으로 구분되는데, Vertical 마겟은 서비스 가입자에게 서비스 사업자가 셋톱박스를 제공하는 시장을 의미하고, Horizontal 마켓은 소매점에서 개인에게 판매되는 시장을 의미한다.

2. 요구사항 가변성 분석

요구사항 가변성 분석을 위한 첫 번째 단계는 다양한 제품 환경에 대한 이해를 바탕으로 요구사항을 도출하는 것이다.

표 1. 제품 환경과 요구사항의 대응
Table 1. Mapping From Product Contexts to Architectural Drivers

환경-요구사항 대응	
제품환경	요구사항
{미국}	{OCAP}
{유럽}	{MHP, 절전}
{한국}	{부팅시간}
{케이블}	{EPG, 보안}
{위성}	{EPG, 채널스캐닝 설정, 채널전환시간,보안}
{Vertical}	{주문자기능}

표 1은 제품 환경에 따른 요구되는 요구사항을 나타낸다. 표 1에서 보는 바와 같이 미국지역에 팔리는 셋톱박스 미들웨어 시스템은 미들웨어 표준으로 OCAP (OpenCable Application Platform)을 요구하는 반면에 유럽지역에 팔리는 시스템은 MHP (Multimedia Home Platform)를 미들웨어 표준으로 요구하고, 절전 요구사항이 추가로 요구됨을 나타낸다. 한편 한국지역에 팔리는 시스템은 OCAP 혹은 MHP 중 특정한 것을 제한하지 않지만, 부팅시간에 대한 요구사항이 있음을 알 수 있다.

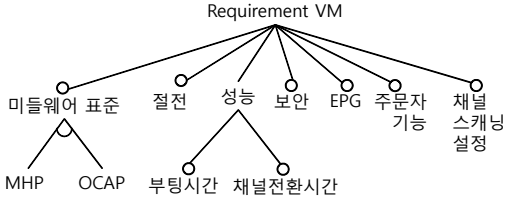

그림 5. 요구사항 가변성 모델
Fig. 5. Requirement Variability Model.

다음으로는 요구사항의 가변성을 훼처 모델을 이용하여 모델링한다. 그림 5는 요구사항의 가변성을 표현한 모델이다. 그림 5에서 보는 바와 같이, 셋톱박스 미들웨어 시스템은 미들웨어 표준으로 MHP와 OCAP 중에 하나를 따라야하므로, 택일 훼처로 모델링되었다. EPG (Electric Program Guide), 주문자 기능, 채널 스캐닝 설정 기능은 제품환경에 따라 선택가능 한 기능이므로, 선택 훼처로 모델링 되었다. 한편 비기능 요구사항으로 절전, 보안, 채널전환시간, 부팅시간 등도 제품환경에 따라 선택 가능한 요구사항이므로, 선택 훼처로 모델링 되었다.

3. 제품라인 아키텍처 가변성 모델링

제품라인 아키텍처의 가변성 모델링은 제품라인 범위 내의 모든 시스템에 공통적인 핵심 공통 아키텍처를 도출하고, 제품 환경과 요구사항의 가변성에 대한 이해를 바탕으로 아키텍처 가변성이 확장된다.

그림 6에서 보는 바와 같이, 디지털 데이터 방송을 위한 셋톱박스 미들웨어 시스템의 제품라인 아키텍처는 EmbeddedApplications계층과 dTVMiddleware계층으로 분리되고, dTVMiddleware 계층은 다시 dTVMiddlewareCore와 PortingLayer로 분리된다. EmbeddedApplications 계층은 셋톱박스에서 실행되는 다양한 응용 프로그램을 포함하는 계층이며, 하위 dTVMiddelware를 사용한다. Porting Layer는 하위계층의 하드웨어의 변화를 상위 dTVMiddleware계층에 차단시키기 위해서 추상화된 인터페이스를 정의한 계층이다. 여기서 노란색으로 표시된 컴포넌트는 제품라인 범위 내의 모든 시스템에 공통인 컴포넌트를 나타내고, 제품라인의 선택적 가변성을

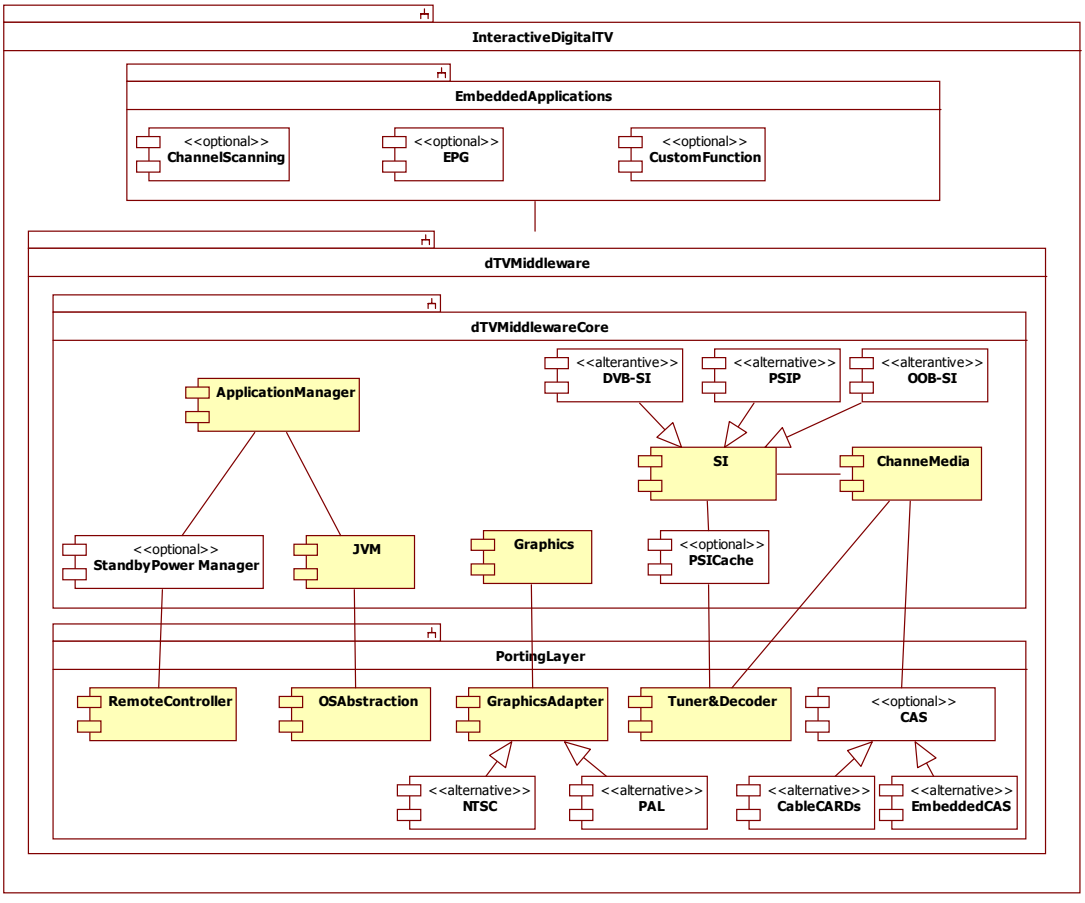

그림 6. 제품라인 아키텍처
Fig. 6. Product Line Architecture

나타내는 컴포넌트는“optional”이라는 스테레오타입으로 표시하였고, 택일적 가변성을 나타내는 컴포넌트는 “alternative”로 표시하였다.

이와 같은 제품라인 아키텍처의 가변성의 일부는 제품 환경의 가변성으로부터 유도되고, 일부는 요구사항의 가변성으로부터 유도된 것이다.

표 2. 제품 환경과 아키텍처의 대응
Table 2. Mapping From Product Contexts to Architectures

환경-아키텍처 대응	
제품환경	아키텍처
{한국}	{NTSC}
{미국}	{NTSC}
{유럽}	{PAL,DVB-SI}
{유럽,Horizontal}	{EmbeddedCAS}
{한국,Vertical}	{CableCARDs}
{미국,Horizontal}	{CableCARDs}
{한국,위성}	{DVB-SI}
{한국,지상파}	{PSIP}
{미국,지상파}	{PSIP}
{미국,위성}	{PSIP}
{한국,케이블}	{OOB-SI}
{미국,케이블}	{OOB-SI}

표 3. 요구사항과 아키텍처의 대응
Table 3. Mapping From Requirements to Architectures

Requirement \ 아키텍처		AVColorEncoder			SI								
		PAL	NTSC	CAS	DVB-SI	PSIP	OOB-SI	EPG	ChannelScanning	CustomFunction	StandbyPowerManaer	PSICache
미들웨어	MHP				+	-	-					
	OCAP	-	+		-	+	+					
절전											++	
성능	채널 전환 시간											++
	부팅 시간										-	
보안				++								
주문자 기능										++		
EPG								++				
채널스캐닝									++			

제품라인 아키텍처의 가변성과 제품 환경의 가변성과의 관계는 표 2와 같다. 표 2에서 보는 바와 같이, 제품 환경이 한국이나 미국인 경우에는 아날로그 비디오 색상 인코딩 방식으로 NTSC (National Television System Committee)를 요구하는 반면에, 유럽은 PAL (Phase Alternating Line)을 요구한다. 따라서 그림 6의 GraphicsAdapter 컴포넌트는 NTSC 혹은 PAL 둘 중의 하나의 방식으로 구현된 컴포넌트가 선택될 수 있는 가변성으로 모델링되었다. 마찬가지로, CAS (Conditional Access System)의 경우, 유럽 지역의 Horizontal 마켓의 경우에는 EmbeddedCAS가, 한국의 Vertical 마켓이나 미국 지역의 Horizontal 마켓의 경우에는 CableCARDs 방식이 요구된다. 따라서 그림 6의 CAS 컴포넌트는 EmbeddedCAS 혹은 CableCARDs 중 하나의 구현 컴포넌트가 선택될 수 있는 가변성으로 모델링되었다. 또한 방송안내를 담당하는 그림 6의 SI (Service Information) 컴포넌트의 경우도 제품 환경에 따라 다른 방식의 구현 컴포넌트가 요구된다. 즉, 한국의 위성 수신환경에서 DVB-SI (Digital Video Broadcasting-Service Information) 방식을, 한국의 지상파나 미국의 지상파/위성인 경우에는 PSIP (Program and System Information Protocol) 방식을, 한국이나 미국의 케이블인 경우에는 OOB-SI (Out of Band-Service Information) 방식을 요구한다.

한편, 요구사항의 가변성도 제품라인 아키텍처의 가변성에 영향을 준다. 표 3은 요구사항의 가변성과 아키텍처의 가변성 간의 상관관계를 정의한 것이다. 표 3에서 보는 바와 같이, MHP는 DVB-SI의 선택에는 긍정적인 영향을 주지만, PSIP과 OOB-SI의 선택에는 부정적인 영향을 준다. 또한, 절전은 StandbyPowerManager의 선택에 긍정적인 영향을 주고, 채널 전환 시간은 PSICache선택에 긍정적인 영향을 준다. 하지만 부팅시간은 StandbyPowerManager의 선택에 부정적인 영향을 준다. 보안, 주문자 기능, EPG, 채널스캐닝 설정은 각각 CAS, Custom Function, CAS, ChanelScanning의 선택에 긍정적인 영향을 준다.

이와 같이, 제품 환경, 요구사항, 아키텍처 간의 상관관계는 다양한 지역 및 환경에서 유도되는 글로벌 소프트웨어의 아키텍처 설계 지식을 표현한 것이다. 이러한 글로벌 소프트웨어의 아키텍처 설계 지식이 어떻게 활용되어 특정 지역의 소프트웨어 아키텍처 모델을 유도하는 지는 다음 절에서 설명된다.

4. 제품라인 아키텍처의 지역화

글로벌 소프트웨어에 대한 제품라인 아키텍처로부터 특정 지역의 제품 아키텍처를 결정하는 절차는 그림 3에서 보는 바와 같이, 제품 환경 선택, 요구사항 선택, 제

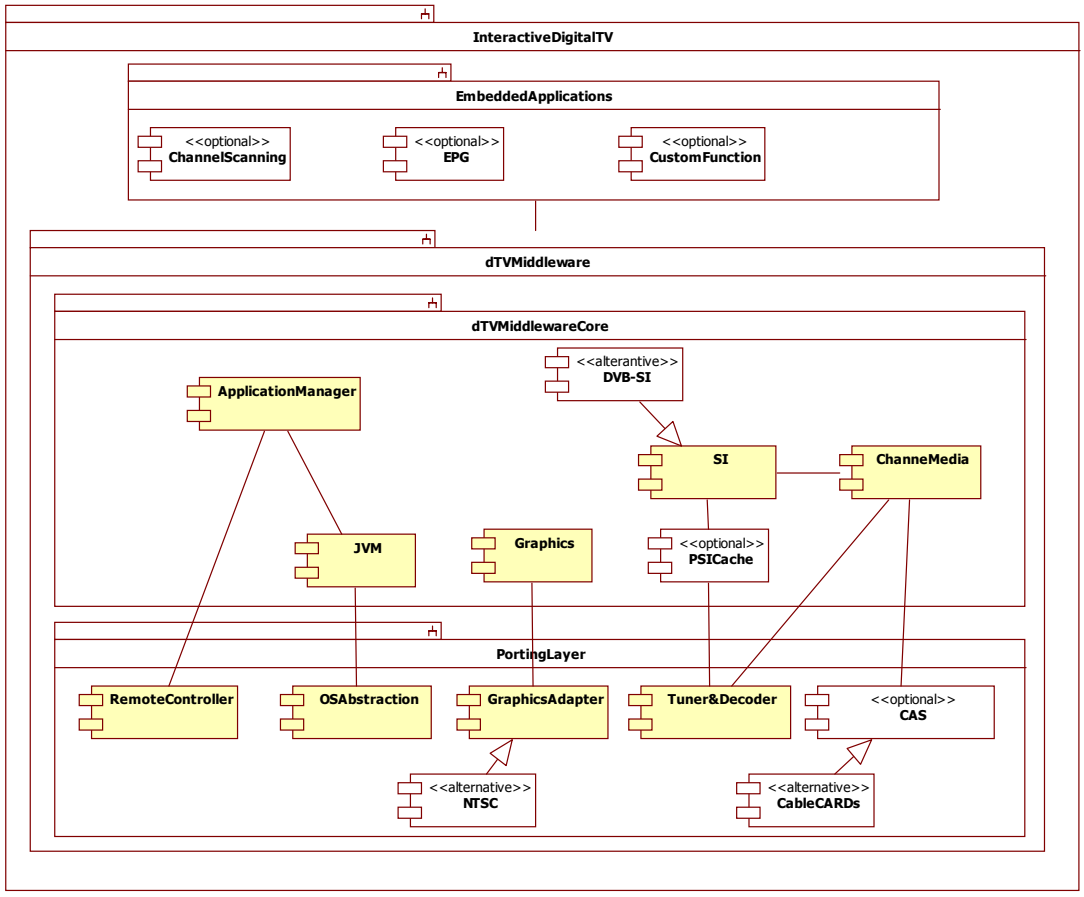

그림 7. 지역화 된 제품 아키텍처
Fig. 7. Localized Product Architecture

품 아키텍처 결정의 단계로 진행된다.

첫 번째 절차인 제품 환경 선택은 해당 지역의 특성에 맞는 제품 환경 요소를 결정하는 것이다. 가령, 그림 4의 제품 환경 요소 중에서 한국, 위성, Vertical을 선택하였다고 가정하자.

다음 단계인 요구사항 선택은 선택된 제품 환경 요소와 표 1의 제품 환경과 요구사항 대응으로부터 그림 5의 요구사항 요소 중에 부팅시간, EPG, 채널 스캐닝 설정, 채널 전환 시간, 주문자 기능이 선택된다. 한편, 그림 5의 미들웨어는 MHP나 OCAP 중 하나를 반드시 요구사항으로 선택해야 하나, 제품 환경 요소로부터 이들 중에서 어떤 것을 선택해야 하는 지에 대한 제약사항이 없으므로, 아무것이나 선택해도 된다. 만약 MHP 미들웨어로서 선택한다고 가정하고 마지막 단계를 진행한다.

마지막 단계에서는 선택된 요구사항과 표 3의 요구사항과 아키텍처의 대응으로부터 DVB-SI, EPG, ChannelScanning, PSICache, CustomFunction의 아키텍처 요소가 선택의 대안으로 추천되고, 선택된 제품 환경과 표 2의 제품 환경과 아키텍처의 대응으로부터 NTSC, DVB-SI, CableCARDs의 아키텍처 요소가 결정된다. 결론적으로 한국지역의 Vertical 마켓을 위한 위성 셋톱박스를 위한 지역화 된 제품 아키텍처는 그림 7과 같다.

VI. 결 론

본 논문의 주된 공헌은 다음 두 가지로 요약된다.

첫째, 소프트웨어 글로벌화를 지원하기 위한 아키텍처 설계 지식 모델의 개발이다. 즉, 이 모델을 통해서 하나의 제품라인 아키텍처 모델로부터 다양한 지역 및 문화 환경에 적합한 아키텍처 모델을 간단한 선택과정으로 도출할 수 있다. 뿐만 아니라 향후에 소프트웨어 글로벌화 범위가 확장되어 더 많은 지역 및 문화의 가변성이 고려된다고 하였을 때, 아키텍처의 어떤 부분이 영향을 받는 지 쉽게 추적할 수 있는 기반을 제공할 수 있다.

둘째, 소프트웨어 글로벌화 전략을 소프트웨어 제품라인 공학 방법론으로 체계화한 것이다. 지금까지 소프트웨어 글로벌화 전략은 특정 언어 및 로케일의 정보를 코드 상에서 쉽게 변환 가능할 수 있는 구현 기술 및 도구에 초점을 두었지만, 본 논문에서는 소프트웨어 글로벌화 전략을 아키텍처 관점으로 확장시킴으로써, 문화적 차이에서 오는 다양한 요구사항의 변화를 체계적으로 고려하는 글로벌 소프트웨어 개발 방법론을 제시한 것이다.

향후 연구로는 제품 환경, 요구사항, 아키텍처 간의 관계를 정립한 아키텍처 설계 지식모델이 일관적이고 완전한 모델인 지를 검사하는 방법을 개발할 것이며, 이러한 모델을 효율적으로 만들어 내기 위한 모델링 패턴 및 안내지침을 개발 할 예정이다.

참 고 문 헌

[1] E. Uren, R. Howard, T. Perinotti, “Introduction to software internationalization and localization”, John Wiley & Sonc Inc., 1993

[2] A. Deitsch, D. Czarnecki, “Java Internationalization”, O'Reilly, 2001.

[3] Nicholas Symmonds, "Internationalization and Localization using Microsoft .NET", Apress, 2002.

[4] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, A. S. Peterson, “Feature-Oriented Domain Analysis (FODA) Feasibility Study”, Technical report, SEI, Carnegie Mellon University, 1990

[5] P. Clements, L. Northrop. "Software Product Lines." Addison-Wesley, 2003

[6] S. Choi, "Validation of Extended Feature Model Constraints using Semantic Web Technologies", Journal of Korean Institute of Information Technology, vol. 9, issue 10, pp. 229-236, Oct. 2011

[7] D. Kim, W. Kim, Y. Kim, “A Study of Design for Embedded S/W based on Model Driven Architecture,” The J. of the Institute of Webcasting, Internet and Telecommunication, Vol. 6, No. 1, pp. 67~74, 2006

※ 본 연구는 한성대학교 교내연구장려금 지원과제입니다.

저자 소개

이 관 우

• 2003년 : POSTECH, 박사
• 2008년 : University of Limerick, 방문교수
• 2003년 ~ 현재 : 한성대학교 정보시스템공학과, 부교수

<주관심분야: Software Product Line Engineering, M2M, Aspect-Oriented Programming, Software Architecture