시맨틱 웹 기술을 이용한 특성 구성 검증☆

Feature Configuration Validation using Semantic Web Technology

최 승 훈*
Seung Hoon Choi

요 약

소프트웨어 제품들 사이의 공통된 개념과 서로 다른 개념들을 표현한 특성 모델과, 특정 제품에 포함될 특성들을 선택한 결과인 특성 구성은 소프트웨어 프러덕트 라인 개발 방법론에서 핵심 요소이다. 이들에 대한 정형적 시맨틱과 논리적 추론에 대한 연구가 진행 중이지만 시맨틱 웹 기술을 이용한 특성 모델 온톨로지 구축과 특성 구성 검증에 대한 연구는 아직 부족한 상황이다. 본 논문에서는 온톨로지와 시맨틱 웹 기술을 이용하여 특성 모델의 정형적 시맨틱을 정의하고 특성 구성을 검증하는 기법을 제안한다. 특성 모델과 특성 구성에 포함된 지식을 시맨틱 웹 표준 언어인 OWL(Web Ontology Language)로 표현하고 특성 구성을 검증하기 위한 규칙은 시맨틱 웹 규칙 언어인 SWRL(Semantic Web Rule Language)로 정의한다. 본 논문의 기법은, 특성 모델의 정형적 시맨틱을 제공하며 특성 구성 검증을 자동화할 뿐 만 아니라 SQWRL과 같은 다양한 시맨틱 웹 기술 적용을 가능하게 한다.

ABSTRACT

The feature models representing the common and variable concepts among the software products and the feature configurations generated by selecting the features to be included in the target product are the essential components in the software product lines methodology. Although the researches on the formal semantics and reasoning of the feature models and feature configurations are in progress, the researches on feature model ontologies and feature configuration validation using the semantic web technologies are yet insufficient. This paper defines the formal semantics of the feature models and proposes a feature configuration validation technique based on ontology and semantic web technologies. OWL(Web Ontology Language), a semantic web standard language, is used to represent the knowledge in the feature models and the feature configurations. SWRL(Semantic Web Rule Language), a semantic web rule languages, is used to define the rules to validate the feature configurations. The approach in this paper provides the formal semantic of the feature models, automates the validation of feature configurations, and enables the application of various semantic web technologies, such as SQWRL.

☞ KeyWords : software product lines, semantic web, feature model, feature configuration validation, 소프트웨어 프러덕트 라인, 시멘틱 웹, 특성 모델, 특성 구성 검증

1. 서 론

급변하는 컴퓨팅 환경에서 고객들의 다양한 요구 사항들을 만족시키기 위해서는 동일한 문제 영역에 속하는 응용 소프트웨어들 사이의 공통점과 차이점을 효과적으로 활용하는 것이 중요하다. 이를 위해 최근 소프트웨어 프러덕트 라인 개발 방법론이 주목을 받고 있다.

소프트웨어 프러덕트 라인 개발 패러다임은, 소프트웨어 개발 단계 초기에 소프트웨어 제품군에 속하는 멤버들 사이의 차이점과 공통점을 미리 예측하고 분석함으로써 보다 전략적인 재사용이 가능하도록 하여 소프트웨어 개발 생산성을 향상시키고자 한다. 이를 위해 특정 도메인에 존재하는 공통된 핵심 자산들을 초기 단계에 개발한 후 특정 소프트웨어 생산 시 가변성을 지원하도록 한다. 성공적인 소프트웨어 프러덕트 라인

* 종신회원 : 덕성여자대학교 컴퓨터학과 교수
csh@duksung.ac.kr
[2010/03/01 투고 - 2010/03/12 심사(2010/05/24 2차) - 2010/07/12 심사완료]
☆ 본 연구는 덕성여자대학교 2009년도 교내연구비 지원에 의해 수행되었음.

개발을 위해서는 도메인 분석 과정이 필수적이며 이 과정을 통해 도메인에 존재하는 지식을 분석하여 재사용 가능한 형태로 표현한다. 도메인 분석 과정에서 가장 핵심적인 역할을 하는 것이 특성 모델이다.

특성 모델은, 동일한 소프트웨어 프러덕트 라인에 속하는 제품들 사이의 공통점과 차이점을 특성이라는 기본 단위로 표현하고 특정 소프트웨어 제품 생산 시 가변성을 지원하기 위한 모델이다[1]. 특성 모델은 또한 유효한 제품 생산을 위한 여러 가지 규칙과 제한 조건들을 포함한다. 특성 구성은 특정 소프트웨어 제품 생산 시 이 제품을 위해 특성 모델로부터 선택된 특성들의 집합을 의미한다. 즉, 특성 구성은 특성 모델에 표현되어 있는 프러덕트 라인 멤버들 사이의 차이점들 중에서 생성하고자 하는 특정 제품이 포함해야할 특성들에 대한 요구 사항을 나타낸다.

최근 특성 모델 및 특성 구성에 대한 연구가 활발히 진행되고 있다. 특히, 소프트웨어 제품들이 포함하는 특성의 종류가 많아지고 특성들 사이의 관계가 복잡해짐에 따라, 특성 모델의 정형적 시맨틱 정의, 특성 모델과 특성 구성의 일관성 및 정확성 검증, 특성 모델로부터의 여러 가지 정보 분석 등에 대한 연구가 활발히 진행 중이다.

본 논문에서는 온톨로지와 시맨틱 웹 기술을 이용하여 특성 모델의 정형적 시맨틱을 정의하고 특성 구성을 검증하는 기법을 제안한다. 특성 모델과 특성 구성에 포함된 지식을 시맨틱 웹 표준 언어인 OWL(Web Ontology Language)[2]로 표현하고 특성 구성을 검증하기 위한 규칙은 시맨틱 웹 규칙 언어인 SWRL(Semantic Web Rule Lanague)[3]로 정의한다. 또한, 특성 구성 검증은 Jess 규칙 엔진을 이용한다.

본 논문의 기법은, 특성 모델을 위한 온톨로지 기반의 메타모델을 제공함으로써 특성 모델에 대한 정형적 시맨틱을 제공하며 SQWRL과 같은 다양한 시맨틱 웹 기술을 적용할 수 있게 해준다. 또한, 자동화된 특성 구성 검증 기능을 제공함으로써 대규모의 특성 모델에서 특성의 종류가 많아지고 제한 조건이 복잡한 경우 유용하게 사용될 수 있다. 특성 모델을 위한 메타 모델, 특성 모델 인스턴스, 특성 구성 검증 규칙 등이 분리되어 있어 각각을 수정하거나 확장하는데 용이하다는 장점을 가진다.

본 논문의 구성은 다음과 같다. 제 2 장에서 특성 모델 및 특성 구성과 시맨틱 웹 기술에 대해서 간략히 살펴보고, 제 3 장에서 시맨틱 웹 기술을 이용한 특성 구성 검증 기법을 기술한다. 제 4 장에서 사례 연구를 통한 특성 구성 검증 기법 과정을 살펴본 후 본 논문에서 제안한 기법을 평가한다. 제 5 장에서 관련 연구를 살펴보고 제 6 장에서 결론 및 향후 연구 과제를 기술한다.

2. 연구 배경

2.1 특성 모델과 특성 구성

특성 모델은 특성들 사이의 계층 구조를 나타내는 특성 다이어그램과 제한 조건 등을 나타내는 부가적인 정보로 구성된다. 특성 다이어그램에서 제품군에 속하는 멤버들이 공유하는 공통점은 필수적 특성(mandatory feature)으로 표현되며, 멤버들 간의 차이점은 선택적(optional) 특성과 택일적(alternative) 특성, OR 특성 등으로 표현된다.

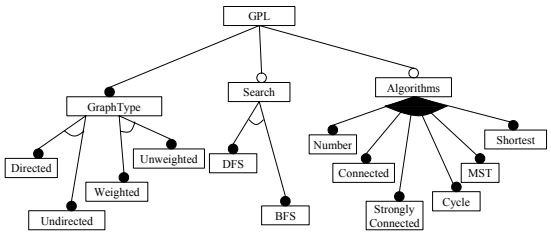
필수적 특성은 어플리케이션에 반드시 포함되어야 하는 특성을 의미하고, 택일적 특성은 여러 개 중에서 하나 만이 선택되는 특성을 의미한다. 선택적 특성은 특정 제품에 포함되거나 또는 포함되지 않거나 둘 중의 하나로 선택되는 특성을 의미한다. OR 특성은 여러 개 중에서 적어도 하나가 선택되어야 하는 특성을 의미한다. 이러한 특성 모델은 응용 프로그램 개발에 있어서 '결정 공간(decision space)'을 정의한다[4].

그림 1은 소프트웨어 제품 라인 기술들에 대한 평가를 위해 [5]에서 제안한 표준 문제인 Graph Product Line(GPL) 특성 모델을 나타내며 본 논문에서는 이 특성 모델을 예제로 하여 특성 구성 검증을 위한 기법을 기술한다. 그림 1에서 까만 원

은 필수적 특성, 빈 원은 선택적 특성을 의미한다. 빈 원호로 연결된 링크는 택일적 특성 그룹을 의미하며 까만 원호로 연결된 링크는 OR 특성 그룹을 위미한다.

특성 구성이란 특성 모델로부터 특정 제품 개발자가 선택한 특성 선택 결과를 의미한다[6]. 즉, 특성 모델에 표현되어 있는 제품 라인 멤버들 사이의 차이점들 중에서 생성하고자 하는 특정 제품이 포함해야할 특성들에 대한 요구 사항을 의미한다. 특성 모델에 포함된 특성들 사이에는 다음과 같이 두 가지의 추가적인 관계 또는 제한조건이 존재하며, 특성 구성은 이 제한 조건에 맞게 생성되어야 한다.

- requires(필요로 한다): 특성 구성에 어떤 특성이 포함된다면 다른 어떤 특성도 반드시 포함되어야 한다
- excludes(배제한다): 특성 구성에 어떤 특성이 포함된다면 다른 어떤 특성은 반드시 포함되어서는 안 된다.



(그림 1) Graph Product Line의 특성 모델

2.2 온톨로지와 시맨틱 웹 기술

온톨로지란 원래 사물(thing)의 기본적인 범주나 세상을 구성하는 구성 요소들을 상징하는 일반적인 개념을 다루는 학문이다. 정보 기술이 발달되고 많은 양의 정보가 축적됨에 따라 IT 분야에서 지식을 표현하고 활용하기 위해 온톨로지에 대한 연구가 최근 다양한 분야에서 진행되고 있다. 온톨로지의 궁극적인 목적은 컴퓨터가 해석, 이해, 처리할 수 있는 특정 영역의 지식 체계를 모델링하는 것이다[7]. 즉, 인간의 개념화 과정 결과로 형성된 온톨로지를 컴퓨터가 처리할 수 있는 형태로 모델링하여 표현함으로써 정보시스템 분야에 활용하고자 한다. 대부분의 온톨로지는 개념, 속성, 관계, 제약조건, 공리, 인스턴스 등의 여섯 가지 구성 요소로 이루어진다.

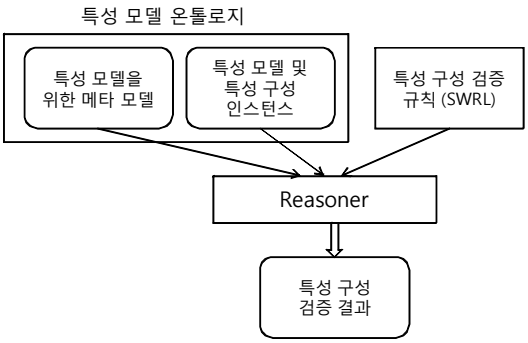
이러한 온톨로지 개념을 현재의 웹에 적용하여 웹 정보에 잘 정의된 의미를 부여하고, 인간 뿐만 아니라 컴퓨터도 그 정보의 의미를 이해하고 처리하고자 하는 웹을 시맨틱 웹이라고 한다. 이를 위해서 여러 계층의 웹 관련 기술이 필요하며 OWL(Web Ontology Langauge)는 온톨로지를 표현하는 웹 표준 언어이다. 최근 시맨틱 웹과 관련하여 다양한 기술들이 개발되고 있으며, 시맨틱 웹에 질의어를 던져 정보를 검색하는데 사용되는 SPARQL(SPARQL Protocol and RDF Query Language)와 SQWRL(Semantic Query-Enhanced Web Rule Language), 온톨로지에 표현된 지식에 규칙을 제공하여 새로운 지식 추론을 가능하게 해 주는 SWRL(Semantic Web Rule Language) 등이 대표적인 예이다. 시맨틱 웹 구성 요소들을 자바 클래스와 객체 등을 사용하여 프로그래밍적으로 다룰 수 있도록 해 주는 프레임워크(예, Jena Semantic Web Framework)도 개발되었다.

본 논문에서는 특성 모델에 포함된 지식을 온톨로지를 이용해서 표현하고 SWRL을 이용하여 특성 구성을 검증하기 위한 규칙을 정의한다. 온톨로지 개발 환경을 제공하는 Protege 도구[8]와 Jess 추론 엔진[9], 이 둘을 연동시켜주는 SWRLJessTab 플러그 인 등을 이용하여 특성 구성 검증 기법을 적용해 본다.

3. 특성 구성 검증을 위한 시맨틱 웹 기술

시맨틱 웹 기술을 적용한 특성 구성 검증 기법의 전체 구성은 그림2와 같다. 특성 모델을 위한 온톨로지에는 특성 모델을 위해 정의된 메타 모델과 이 메타 모델을 따라 구축된 특성 모델 및 특성 구성 인스턴스가 존재한다. 이 온톨로지와 SWRL을 이용해 정의된 특성 구성 검증 규칙을

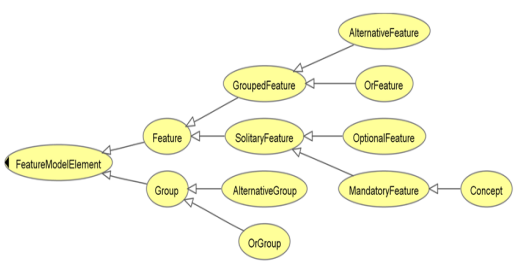
이용해서 지식 추론기가 동작하여 특성 구성 검증 결과를 추론한다.



(그림 2) 특성 구성 검증 기법의 전체 구조

3.1 특성 모델 및 구성을 위한 온톨로지

먼저 특성 모델을 위한 메타 모델을 OWL 기반의 온톨로지로 정의하기 위해 클래스 계층 구조를 정의한다(그림3 참조). 특성 모델을 구성하는 요소(FeatureModelElement 클래스로 표현)들 중에서 가장 중요한 요소인 특성은 크게 두 가지로 분류된다. 먼저 항상 그룹의 한 멤버로 존재하는 택일적 특성과 Or 특성들이 속하는 GroupedFeature 클래스가 있다. 두 번째로 그룹에 속하지 않고 독자적으로 존재하는 필수적 특성과 선택적 특성들이 속하는 SolitaryFeature 클래스가 있다. 특성 모델에 존재하는 그룹으로는 택일적 그룹(AlternativeGroup 클래스)과 OR 그룹(OrGroup 클래스)이 존재한다. 그림3에서 Group 클래스와 Feature 클래스는 FeatureModelElement 클래스를 상속한다.



(그림 3) 특성 모델을 위한 클래스 계층 구조

특성 모델의 트리 구조를 표현하기 위해서 여러 가지 객체 속성(object property)이 필요하다. SolitaryFeature들은 다른 일반 Feature를 부모로 가질 수 있으며(hasParent로 표현), 일반 Feature들은 다른 SolitaryFeature들을 자식으로 가질 수 있다(hasChild로 표현). 일반 Feature 들은 Group을 가질 수 있으며(hasGroup으로 표현) 각 Group은 GroupedFeature들을 멤버로 가질 수 있다(hasMember로 표현). 표1은 이러한 사항들을 나타내기 위해 정의한 객체 속성들과 그 의미 및 정의역(domain)과 치역(range)을 보여준다.

(표 1) 특성 모델 온톨로지에서의 객체 속성들

객체 속성	의미(Domain/Range)
hasParent	한 특성의 부모 특성을 의미함 (D:SolitaryFeature/R:Feature)
hasChild	한 특성의 자식 특성을 의미함 (D:Feature/R:SolitaryFeature)
hasGroup	한 특성이 가지는 그룹을 의미함 (D:Feature/R:Group)
isGroupOf	한 그룹의 부모 특성을 의미함 (D:Group/R:Feature)
hasMember	한 그룹의 멤버를 의미함 (D:Group/R:GroupedFeature)
isMemberOf	한 멤버가 속하는 그룹을 의미함 (D:GroupedFeature/R:Group)
requires	한 특성과 requires 관계에 있는 특성을 의미함 (D:Feature/R:Feature)
excludes	한 특성과 excludes 관계에 있는 특성을 의미함 (D:Feature/R:Feature)

특성 구성을 위해서 몇 가지 데이터형 속성(data type property)을 정의한다. 먼저 각 특성이 특성 구성에 선택되었는지를 나타내는 isSelected 속성이 필요하다. 또한, 그룹의 경우 멤버들 중에서 몇 개가 특성 구성에 선택되었는지를 나타내는 hasSelectedMemberCount 속성이 필요하다. 이 두 속성에 대한 의미와 정의역 및 치역은 표2와 같다. 특정 영역의 도메인 분석을 통해 만들어지는 특성 모델은 위에서 정의한 클래스와 속성을

기반으로 하여 생성되는 인스턴스로 표현된다.

(표 2) 특성 구성을 위한 데이터형 속성들

데이터형 속성	의미(domain/range)
isSelected	한 특성이 특성 구성에 선택되었는지를 나타냄 (Feature/boolean) (OWL의 open world assumption 때문에 특성 구성 시 true 또는 false 값을 반드시 할당해야 함)
hasSelectedMemberCount	한 그룹의 멤버들 중에서 몇 개가 특성 구성에 선택되었는지를 나타냄 (Group/int)

3.2 특성 구성 검증을 위한 SWRL 규칙

특성 구성에서 검증해야 할 제한 조건은 다음과 같으며, SWRL 규칙은 생성된 특성 구성이 이러한 제한 조건들을 만족하는지 검사하는 규칙들을 제공한다.

1) 필수적 특성에 대한 제한 조건: 특성 모델에서 필수적 특성은 반드시 특성 구성에 포함되어야 한다.
2) 택일적 특성에 대한 제한 조건:
   - 특성 모델에서 택일적 특성 그룹에 속하는 특성들은 그 그룹 중에서 반드시 하나만이 특성 구성에 포함되어야 한다.
   - 또한, 택일적 특성 그룹 중에서 적어도 하나의 특성은 특성 구성에 포함되어야 한다.
3) OR 특성에 대한 제한 조건: 특성 모델에서 OR 특성 그룹을 이루는 특성들 중 적어도 하나의 특성은 특성 구성에 포함되어야 한다.
4) requires 관계에 대한 제한 조건: 특성 구성에 포함된 각 특성에 대해서, 특성 모델에서 이 특성과 requires 관계에 있는 모든 특성들은 특성 구성에 포함되어야 한다.
5) excludes 관계에 대한 제한 조건: 특성 구성에 포함된 각 특성에 대해서, 특성 모델에서 이 특성과 excludes 관계에 있는 모든 특성들은 특성 구성에서 제외되어야 한다.

• 필수적 특성을 위한 SWRL 규칙

그림 4는, 필수적 특성이 모두 특성 구성에 포함되었는지를 검사하기 위한 규칙을 보여주는 코드이다. 필수적 특성이면서 특성 구성에 선택되지 않았으면 MandatoryFeatureError 클래스의 인스턴스를 생성함으로써 그 특성이 필수적 특성 오류를 일으킨다는 사실을 추론한다.

```
MandatoryFeature(?f) ∧ isSelected(?f, false) →
MandatoryFeatureError(?f)
```

(그림 4) 필수적 특성을 위한 SWRL 규칙

• 택일적 특성을 위한 SWRL 규칙

그림5와 그림6은, 택일적 특성에 대한 검사를 수행하는 SWRL 규칙을 보여준다. 그림 5는, 택일적 특성 그룹의 멤버 중에서 특성 구성에 선택된 멤버의 개수가 1이 아닌 경우 FaultyAlternativeGroup 인스턴스를 생성함으로써 그 택일적 특성 그룹이 오류를 일으킨다는 사실을 추론한다. 그림 6은, 이 경우에 속하는 모든 멤버들이 FaultyAlternativeFeature 임을 추론하는 규칙이다.

```
AlternativeGroup(?g) ∧ hasSelectedMemberCount(?g, ?n)
∧ swrlb:notEqual(?n, 1) ∧ isGroupOf(?g, ?f)
→ FaultyAlternativeGroup(?f)
```

(그림 5) 택일적 특성을 위한 규칙 I

```
AlternativeGroup(?g) ∧ hasSelectedMemberCount(?g, ?n)
∧ swrlb:notEqual(?n, 1) ∧
isMemberOf(?m, ?g) → FaultyAlternativeFeature(?m)
```

(그림 6) 택일적 특성을 위한 규칙 II

• OR 특성들을 위한 SWRL 규칙

그림7은, Or 특성 그룹에서 어떠한 특성도 선택되지 않은 경우를 검사하는 규칙으로서, OR 그룹의 멤버 중에서 특성 구성에 선택된 멤버가 하나도 없는 경우 FaultyOrGroup 인스턴스를 생성함으로써 그 OR 특성 그룹이 오류를 일으킴을 추론한

다. 그림8은, 이 경우에 속하는 모든 멤버들이 FaultyOrFeature 임을 추론하는 규칙이다.

```
OrGroup(?g) ∧ hasSelectedMemberCount(?g, ?n) ∧
swrlb:lessThan(?n, 1) ∧ isGroupOf(?g, ?f)
→ FaultyOrGroup(?f)
```

(그림 7) OR 특성을 위한 규칙 1

```
OrGroup(?g) ∧ hasSelectedMemberCount(?g, ?n) ∧
swrlb:lessThan(?n, 1) ∧ isMemberOf(?m, ?g)
→ FaultyOrFeature(?m)
```

(그림 8) OR 특성을 위한 규칙 2

• requires 관계를 위한 SWRL 규칙

그림9는, 특성 구성이 특성 모델에 표현되어 있는 requires 관계를 만족하도록 생성되었는지를 검사하는 규칙이다. 특성 구성에 선택된 각 특성에 대해서 requires 관계에 있는 특성이 선택이 되지 않은 경우 FaultyRequiredFeature 인스턴스를 생성함으로써 이 특성이 오류를 일으킴을 추론한다.

```
Feature(?f1) ∧ isSelected(?f1, true) ∧ requires(?f1, ?f2)
∧ isSelected(?f2, false)
→ FaultyRequiredFeature(?f2)
```

(그림 9) Requires 관계를 위한 규칙

• excludes 관계를 위한 SWRL 규칙

그림10은, 특성 구성이 특성 모델에 표현되어 있는 excludes 관계를 만족하도록 생성되었는지를 검사하는 규칙이다. 특성 구성에 선택된 하나의 특성에 대해서 excludes 관계에 있는 특성이 선택된 경우 FaultyExcludedFeature 인스턴스를 생성함으로써 이 특성이 오류를 일으킴을 추론한다.

```
Feature(?f1) ∧ isSelected(?f1, true) ∧ excludes(?f1, ?f2)
∧ isSelected(?f2, true)
→ FaultyExcludedFeature(?f2)
```

(그림 10) excludes 관계를 위한 규칙

4. Protege를 이용한 사례 연구

이 절에서는 온톨로지 개발 도구인 Protege를 이용하여 특성 구성을 검증하는 과정을 보여준다.

4.1 특성 모델 및 특성 구성 온톨로지 구축

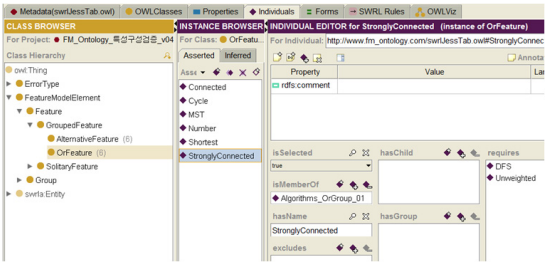
특성 구성을 검증하기 위해서 먼저 특성 모델과 특성 구성을 위한 온톨로지를 구축한다. 제3절에서 기술한 온톨로지를 이용하여 그림1의 GPL 특성 모델 온톨로지를 구축한다. 또한, SWRL 규칙이 특성 구성의 오류를 잘 검증하는지 확인하기 위해서 예제 특성 구성에 오류를 일으키는 조건을 표3과 같이 추가하였다.

(표 3) 특성 구성 검증 기법을 위한 테스트 케이스

검증하고자 하는 오류의 종류	특성 구성에 추가한 오류	검증 결과
택일적 특성 오류	같은 택일적 특성 그룹에 속하는 Undirected 특성과 Directed 특성을 동시에 특성 구성에 포함시킨다.	특성 구성이 택일적 특성 제한 조건을 만족시키지 않음(FaultyAlternativeGroup(GraphType), FaultyAlternativeFeature(Directed), FaultyAlternativeFeature(Undirected))
requires 관계 오류	StronglyConnected 특성은 특성 구성에 포함시키고, StronglyConnected 특성과 requires 관계에 있는 DFS 특성과 Unweighted 특성을 특성 구성에 포함시키지 않음	특성 구성이 requires 제한조건을 만족하지 않음(FaultyRequiredFeature(Unweighted), FaultyRequiredFeature(DFS))

그림11은 Protege 도구를 이용하여 GPL 특성 모델 온톨로지를 구축한 모습을 보여준다. GPL 특성 모델 중에서 Connected, Cycle, MST, Number, Shortest, StronglyConnected 특성이 OrFeature의 인스턴스로 생성되어 있음을 보여준다. 현재 선택된

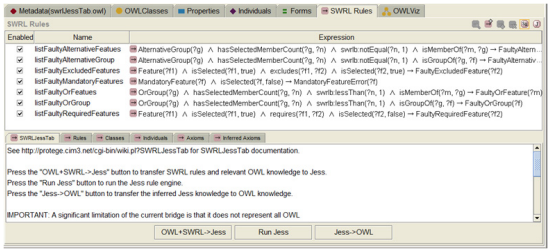
StronglyConnected 특성의 속성 중 requires 속성을 보면, DFS와 Unweighted 특성을 포함하고 있다.



(그림 11) 특성 모델 온톨로지와 StronglyConnected 특성

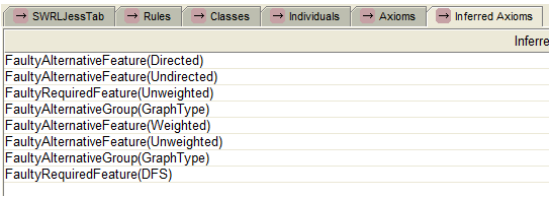
4.2 SWRLJessTab을 이용한 검증

Protege에서의 SWRL은 SWRLJessTab 플러그인을 통해서 실행할 수 있다. 그림12는 SWRLJessTab을 이용하여 특성 구성 검증 규칙을 정의한 화면이다. 아래 세 개의 버튼 중 "Run Jess" 버튼을 누르면 특성 구성 검증 규칙이 실행되고 현재 구축된 특성 모델 및 특성 구성 온톨로지에서 규칙에 해당하는 사실을 찾아내어 특성 구성 제한 조건을 벗어나는 특성들을 발견한다.



(그림 12) SWRLJessTab을 이용한 특성 구성 검증 규칙 정의

그림13은 특성 구성 검증 규칙을 현재 온톨로지에 적용한 결과를 보여준다. GraphType 특성이 오류가 있는 택일적 특성 그룹의 인스턴스이며, Unweighted, Weighted, Undirected, Directed가 택일적 특성 그룹의 오류를 유발하는 특성들임을 알 수 있다. 또한, Unweighted와 DFS 특성이 requires 관계와 관련된 오류를 유발함을 알 수 있다.



(그림 13) 특성 구성 검증 결과

4.3 평가

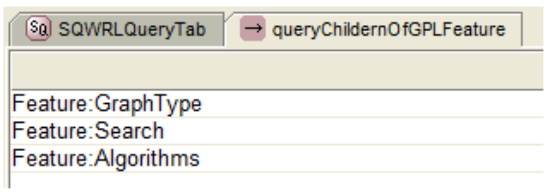
본 논문에서 제안한 시맨틱 웹 기술을 이용한 특성 구성 검증 기법은 다음과 같은 장점을 가진다.

• 다양한 시맨틱 웹 기술 적용 가능성

시맨틱 웹 기술은 최근 크게 관심을 받아서 다양한 관련 기술들이 연구되고 개발되고 있다. 이러한 시맨틱 웹 기술들을 특성 모델과 특성 구성 온톨로지에 적용할 수 있다. 예를 들어, SQWRL과 같은 쿼리 언어를 이용하면 특성 모델 자체에 대한 검색이 용이하다. GPL 특성 모델에서 GPL 특성의 하위 특성들을 모두 찾으려고 한다면 그림14와 같은 SQWRL 쿼리 언어를 작성할 수 있다. 그림15는 GPL의 하위 특성들을 검색한 결과이다.

```
hasParent(?f, GPL) ∧ hasName(?f, ?name) ∧
swrlb:stringConcat(?featureName, "Feature:", ?name)
→ sqwrl:select(?featureName)
```

(그림 14) SQWRL 쿼리 언어를 이용한 특성 모델 검색



(그림 15) GPL 하위 특성 검색 결과

• 다른 도구와의 결합성

Jena와 같은 Java Framework을 이용하면 최근의 시맨틱 웹 기술을 다양한 응용 프로그램 개발 시 쉽게 활용할 수 있다. 따라서, 기존의 소프트웨어 프러덕트 라인 관련 도구나 새로운 도구 개발 시 본 논문의 온톨로지 및 검증 기법을 쉽게 통합할 수 있다.

• 모듈성

본 논문의 검증 기법은 특성 모델의 메타모델, 특성 모델 및 특성 구성 인스턴스, 특성 구성 검증 규칙이 각각 분리되어 있어 각 부분을 수정하거나 확장하기가 용이하다. 특성 모델의 메타모델은 온톨로지의 클래스와 속성으로 정의되어 특성 모델 지식 베이스의 TBox를 이루며 크게 변화하지 않는다. 새로운 소프트웨어 프러덕트 라인 특성 모델과 특성 구성이 생성될 때마다 새로운 인스턴스가 생성되고 특성 모델 지식 베이스의 ABox 부분에 추가된다. 특성 구성 검증 규칙은 SWRL 표현되어 새로운 검증 규칙이 필요한 경우 쉽게 추가할 수 있다.

5. 관련 연구

본 논문의 기법과 관련하여 여러 가지 기존 연구가 존재한다. [10]은 특성 모델을 기능적 특성(functional feature) 뿐 만 아니라 비기능적 특성(extra-functional feature)을 모델링할 수 있도록 확장하였다. 확장된 특성 모델을 Contraint Satisfaction Problem(CSP)로 변환하고, constraint programming을 통하여 특성 모델에 대해서 여러 가지 정보를 추론하는 방법을 제안하였다. 추출 가능한 정보에는, 특성 모델로부터 만들어 질 수 있는 모든 제품의 개수, 비기능적 특성에 대한 제한 조건을 만족하는 모든 특성 구성의 종류 등이 포함된다. 이 기법은 각 특성들이 가지는 비기능적 요구 사항을 만족하는 제품 또는 특성 구성을 검색하는데 유용하지만, 특성들 사이의 requires와 excludes 관계를 고려하지 않는 단점이 있다.

특성 모델 및 특성 구성에 온톨로지와 시맨틱 웹 기술을 적용한 기존 연구가 존재한다. [11], [12]에서는 특성 모델과 특성 구성을 Protege 온톨로지 편집 도구를 이용하여 OWL[13] 언어로 표현하고 RACER[14]나 FaCT[15]와 같은 추론 엔진을 이용하여 특성 구성이 일관적인지를 검증한다. 이 기법은 특성 모델을 위한 메타 모델을 정의하지 않고, 특정 제품군의 특성 모델을 클래스들로 표현한다는 점에서 본 논문의 기법과 다르다. 또한, 본 논문의 기법과 달리 추론 엔진 자체 만으로는 불일치성을 발생시키는 원인을 명확히 파악하지 못하고, 오류 패턴을 이용한 별도의 OWL 디버깅 도구를 구현하여 특성 구성에 존재하는 불일치성의 원인에 대한 힌트를 제공한다. [16]는 Protege 도구 대신 JENA 프레임워크와 Pallet 추론 엔진을 이용하여 [11], [12]의 기법을 구현하였지만, 프로그래밍 적으로 문제를 해결했다는 점 이외에는 기존 연구와 크게 다르지 않다.

[17]는 특성 모델을 위한 온톨로지 프레임워크를 제안하고 하나의 특성 모델이 포함하거나 분산된 특성 모델이 통합된 후에 포함할 수 있는 논리적 오류를 발견하는 기법을 제안하였다. 이 기법은 특성 모델에 대한 메타모델을 온톨로지로 정의하고 특성 모델 검증 규칙을 SWRL로 정의한 점에서 본 논문의 기법과 유사하지만 특성 구성을 검증하는 기법은 제공하지 않는다. 또한, 제안된 검증 규칙은, 특성 A가 특성 B를 requires하고 특성 B는 특성 A를 excludes 하는 경우와 특성 A가 특성 B를 requires하고 특성B가 특성A를 requires 하는 경우 두 가지 경우의 오류만을 검증한다.

[18]은 온톨로지 기반의 특성 모델링 검증을 위해 특성 메타 모델을 제안하였다. 이 논문은 문제 영역을 비즈니스 관점에 중점을 두고 분석하며, 특성을 비즈니스 연산으로 간주해서 이를 표현하기 위한 클래스와 속성들을 정의하였다. 특성 모

델 인스턴스를 메타 모델에 정의된 클래스의 하위 클래스로 표현함으로써 ABox에 저장했다는 점에서 본 논문의 기법과 다르다. 또한, 주로 바인딩에 기초한 특성들 사이의 규칙을 정의하고 이를 기반으로 특성 모델의 유효성을 검증하였으며 검증 결과는 특성 모델이 유효한지 아닌지 여부만을 알려준다. 특성 구성 검증을 위한 구체적인 규칙과 검증 기법은 제안하지 않았다.

본 논문의 특성 구성 검증 기법을 특성 모델 및 특성 구성에 온톨로지와 시맨틱 웹 기술을 적용한 기존 연구들과 비교한 결과를 정리하면 표4와 같다.

6. 결론 및 향후 과제

소프트웨어 프러덕트 라인 패러다임에서는 소프트웨어 제품들 사이의 차이점과 공통점을 체계적으로 분석하여 소프트웨어 개발 생산성을 향상시키고자 한다. 이 때 사용되는 핵심 모델이 특성 모델이며, 특정 제품 생산 시 포함할 특성들을 선택한 결과가 특성 구성이다.

프로젝트의 규모가 커지면 특성 모델과 특성 구성이 복잡해지며 오류를 포함할 가능성 또한 커진다. 따라서, 자동화된 특성 모델 및 특성 구성 검증 기법이 필요하다. 본 논문에서는 최근 각광을 받고 있는 시맨틱 웹 기술을 적용하여 특성 모델을 위한 온톨로지를 정의하고 특성 구성을 검증하는 기법을 제안하였다.

먼저, 특성 모델 및 특성 구성에 대한 온톨로지를 구축하고 특성 구성에 존재할 오류를 검증하기 위한 SWRL 규칙을 정의한다. 특성 모델이 완성되고 특성 구성이 정해지면 이를 온톨로지로 표현한 후 SWRL로 정의된 특성 구성 검증 규칙을 적용하여 특성 구성에 포함된 오류를 발견한다.

(표 4) 기존 연구와의 비교

비교 항목	본 논문의 기법	[12] 제안 기법	[17] 제안 기법
목적	특성 구성 검증	특성 구성 검증	특성 모델 검증
적용한 시맨틱 웹 기술	OWL, SWRL, SQWRL, JessSWRLTab	OWL	OWL, SWRL
특성 모델을 위한 메타 모델	TBox(OWL 클래스와 속성들)	없음	OWL 클래스와 속성들
특성 모델 인스턴스	ABox(OWL 개체들)	TBox(OWL 클래스와 속성들)	ABox(OWL 개체들)
특성 구성 인스턴스	ABox(OWL 개체들)	TBox(OWL 클래스와 속성들)	없음
검증 규칙	SWRL (SWRL을 이용하여 검증 규칙을 독립적으로 정의하기 때문에 다양하고 복잡한 검증 규칙을 정의할 수 있을 뿐 아니라, 검증 규칙의 재사용성이 크고 확장성이 좋음)	OWL 클래스와 속성들 (검증 규칙이 클래스와 속성으로 표현되기 때문에 모든 특성 모델 인스턴스마다 검증 규칙을 추가로 만들어서 온톨로지에 삽입해야 함)	SWRL (특성 모델을 검증하기 위한 규칙을 정의하였으며,
추론 엔진	Jess (특성 구성에서 오류를 일으키는 원인을 나타내는 인스턴스가 생성되어 온톨로지에 추가됨)	FaCT++ (특성 구성이 모순을 포함하는지 여부만을 알려주며, 그 원인을 제공하지 못함. 오류 패턴을 이용한 특성 모델 디버거를 따로 구현함으로써 오류에 대한 힌트를 제공함)	Pellet (특성 모델에서 오류를 일으키는 두 특성을 발견하지만 그 오류의 원인에 대한 설명을 제공하지 못 함)
특성 모델 쿼리	SQWRL	없음	없음

온톨로지 개발 도구인 Protege와 Graph 프러덕트 라인 특성 모델을 사례 연구로 하여 본 논문의 특성 구성 검증 기법을 살펴보았다. 본 논문의 기법은 SQWRL 등 시맨틱 웹 관련 최신 기술을 적용할 수 있고, 다른 도구와의 결합성이 좋고, 규칙을 쉽게 추가할 수 있게 해주는 확장성이 뛰어나다. 본 논문의 기법은 소프트웨어 프러덕트 라인 지원 도구 개발에 활용될 수 있다.

향후 연구 과제로는, 확장된 특성 모델을 위한 온톨로지, 그래픽 특성 모델 작성 도구와의 통합, 보다 큰 예제 시스템으로의 적용, 컴포넌트 기반 소프트웨어 제품 라인 공학 지원 방법, 소프트웨어 제품 코드 생성 자동화 등에 대한 연구가 있다.

참 고 문 헌

[1] J.Bosch, "Design & Use of Software Architectures: Adopting and Evolving a Product-Line Approach", Addison-Wesley, 2000.

[2] http://www.w3.org/TR/owl-features/

[3] [Horrocks] I.Horrocks, P.F.Patel-Schneider, H.Boley, S.Tabet, B.Grosof and M.Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", http://www.w3.org/Submissions/SWRL.

[4] K.C.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, 5:143-168, 1998.

[5] R.E.Lopez-Herrejon and D.S.Batory. "A standard problem for evaluating product line methodologies", In Proceedings of the Third International Conference on Generative and Component-Based Software Engineering, pp.10-24, Erfurt, Germany, September 2001, Springer-Verlag.

[6] S. Thiel and A. Hein, "Systematic Integration of Variability into Product Line Architecture Design", SPLC2 2002, LNCS 2379, pp.130?153, 2002, Springer?Verlag.

[7] 노상규, 박진수 공저, "인터넷 진화의 열쇠, 온톨로지", Good's Toy business 사, 2007.

[8] http://protege.stanford.edu/

[9] http://www.jessrules.com/jess/index.shtml

[10] D.Benavides, P.Trinidad, and A.Ruiz-Cort´es, "Automated Reasoning on Feature Models", In: Proceedings of the 17th Conference on Advanced Information System Engineering (CAiSE'05), Porto, Portugal.

[11] Hai Wang, LI Yuan Fang, Jing Sun, Hongyu Zhang and Jeff Z. Pan. A Semantic Web Approach to Feature Modeling and Verification . In Proc. of the ISWC2005 Workshop on Semantic Web Enabled Software Engineering (SWESE). 2005.

[12] H. H. Wang, Y. F. Li, J. Sun, H. Zhang and J. Pan. "Verifying Feature Models Using OWL", In Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 5(2):117-129, June 2007.

[13] D.L. McGuinness, F. van Harmelen (Eds.), OWL Web Ontology Language Overview, 2003. http://www.w3.org/TR/2003/PR-owl-features-2003 1215/.

[14] V. Haarslev, R. M¨oller, RACER User''s Guide and Reference Manual: Version 1.7.6, 2002.

[15] I. Horrocks, Fact++ web site. http://owl.man.ac.uk/factplusplus/.

[16] V.B.Matcha et. al., "Software Reuse: Ontological Approach to Feature Modeling", IJCSNS(International Journal of Computer Science and Network Security), Vol.9, No.8, August 2009.

[17] L.A.Zaid, F.Kleinermann and O.D.Troyer, "Applying Semantic Web Technology to

Feature Modeling", ACM SAC(Symposium on Applied Computing) '09, March 2009.

[18] X.Peng, W.Zhao, Y.Xue and Y.Wu, "Ontology-Based Feature Modeling and Application-Oriented Tailoring", In: ICSR 2006: 87-100.

◐ 저 자 소 개 ◑

최 승 훈 (Seung Hoon Choi)
1990년 서울대학교 계산통계학과 졸업(학사)
1994년 서울대학교 대학원 계산통계학과 졸업(석사)
1999년 서울대학교 대학원 계산통계학과 졸업(박사)
2000년 ~ 현재 덕성여자대학교 컴퓨터학과 교수
2004년 ~ 2005년 George Mason University 방문 연구
관심분야 : 소프트웨어 프러덕트 라인, 온톨로지, 자동 생성 프로그래밍
E-mail : csh@duksung.ac.kr