Ontology 기반의 Feature - Class 변환 기법

김동리° 송치양, 백두권

고려대학교 컴퓨터학과

{c13635°,baik}@korea.ac.kr, cysong@sangu.ac.kr

A method of Feature - Class Transformation using Ontology

Dong-Ri Kim°, Chee-Yang Song , Doo-kwon Baik

Department of Computer Science and Engineering, Korea University

요 약

소프트웨어 개발을 위한 모델링 방법 중 대표적인 것으로 UML을 이용한 방법이 있으며, 제품계열 공학에서 소프트웨어의 재사용을 위한 모델링 방법으로 feature 모델링에 관한 연구가 진행 되고 있다. feature 모델링 방법은 잘 정의된 개발 기법을 제공하여 활용되고 있으나 다소 범용 적이지 않다. 또한 그 구조물이 UML과 상이하여 UML 사용자가 feature 모델을 재사용하는 데는 어려움을 가지고 있고, feature 모델에서 class모델로의 변환을 제시한 기존연구는 도메인 전문가에 의해 경험적으로 모델링을 하기 때문에 모호성과 이해의 오류, 그리고 잘못된 해석 등의 문제가 발생 된다. 그리고, feature 모델과 class모델의 모든 요소를 매핑하여 변환하지 않는다는 점에서 완전하지 못하다. 따라서 본 논문에서는 Ontology를 이용하여 의미 기반의 명확한 명세를 통한 feature모델의 class 모델로의 변환기법을 제시하고, 이를 위해 feature 모델과 class 모델의 구조물의 요소를 정의하고 이를 기반으로 feature 모델과 OWL, 그리고 class 모델 속성간의 매핑 규칙을 제시하고, 본 논문에서 제시한 변환 프로세스를 이용하여 사례연구를 하였다.

1. 서 론

소프트웨어 개발을 위한 모델링 방법 중 대표적인 것으로 UML을 이용한 방법이 있으며, 제품계열 공학에서 소프트웨어의 재사용을 위한 모델링 방법으로 feature 모델링에 관한 연구가 진행 되고 있다. 최근에는 서로 다른 언어로 구성된 모델 간의 상호 운용성을 높여, 모델의 재사용을 높이기 위한 연구들이 진행되고 있다. feature 모델링 방법은 체계적이고 잘 정의된 개발 기법을 제공하여 활용 되고 있으나 다소 범용 적이지 않고, 그 구조물이 UML 모델과 상이하여 UML 모델 사용자가 feature 모델을 재사용하는 데는 어려움을 가지고 있다. 이런 문제점들을 해결하기 위해 기존 연구에서는 feature 모델에서 class모델로의 변환을 제시하고 있으나 모델링 변환이 도메인 전문가에 의해 경험적으로 이루어져 이로 인해 모호성과 이해의 오류, 그리고 잘못된 해석 등의 문제가 발생되곤 한다. 또 feature 모델의 일부 속성만을 변환한다는 점에서 완전하지 못하다[1][2]. 그래서 본 논문에서는 Ontology 기반의 의미 명확한 명세를 통해 feature모델의 class 모델로의 변환기법을 제시한다. 이를 위해 feature 모델과 class 모델의 속성을 정의하고, 이를 기반으로 feature 모델과 OWL 속성간의 매핑 규칙을 정립하고 다음으로 OWL을 class 모델로의 변환을 정립하여 변환 규칙을 만든다.

이렇게 함으로써 기 구축된 feature 모델의 UML class 모델로의 변환을 통한 재사용을 증대 시킬 수 있고, feature 모델을 Ontology를 사용하여 의미를 명확히 정립하고, 이를 통해 feature 모델의 이해와 수정을 용이하게 할 수 있다. 또 본 기법을 역으로 적용하면 class 모델을 feature 모델로 변환하여 모델의 재사용을 할 수 있다.

본 논문 구성으로 2장에서는 feature 모델, OWL, class 모델에 관한 기존연구에 대해 간단히 살펴보고, 3장에서는 XMI를 이용한 feature 모델을 class 모델로의 변환 프로세스와 이 모델 변환에 필요한 속성들간의 매핑 규칙을 제시하고, 4장에서는 3장에서 정립된 매핑 규칙과 변환 방법을 이용한 간단한 적용사례를 보이고, 마지막 5장에서는 결론 및 향후 과제를 논한다.

2. 관련연구

2.1 feature 모델의 class 모델 변환

기존연구에서는 feature의 속성을 이용하여 UML class 모델로 직접 매핑을 실시 하였고, UML의 스트레오 타입을 사용하여 feature 모델의 속성을 표현을 하였다[3]. 그러나 대부분이 도메인 전문가에 의해 경험적으로 모델링이 이루어 지기 때문에 모호성과 이해의 오류, 잘못된 해석 등의 문제가 발생 할 수 있다[2]. 그리고 [3][4]에서는 Ontology를 이용한 변환이 아니라 직접 feature 모델을 class 모델로 변환하는 방법을 연구 하였다.

2.2 feature 모델의 Ontology 변환

feature 모델을 Ontology로 변환은 모델의 명세 방법 중 하나로 feature 모델의 속성을 OWL의 속성과 매핑하여 표현하는 방법이 있다[2]. feature 모델링을 할 때 도메인 전문가에 의해 경험적인 방법으로 모델링을 함으로써 생기는 문제점을 해결하기 위해

나온 방법이다. 하지만 이것 역시 feature 모델의 속성 모두를 OWL로 표현 하지 못한다는 문제점을 가지고 있다.

2.3 Ontology의 class 모델 변환

Ontology의 class 모델로의 변환은 Ontology를 좀더 쉽게 이해 하고 사용성을 높이기 위해 class 모델로 변환하는 연구 이다[5]. 하지만, 이 방법에서 Ontology의 일부 속성만을 class 모델 속성과 매핑을 하여 변환을 해주기에 일부 변환해 주지 못하는 부분이 있다.

표1은 관련 연구들의 특징을 비교한 표이다. 기존 연구에서는 각 속성들을 수동으로 매핑을 하여 시간이 과도하게 소요 되고, 오류의 발생 확률이 높았다. 그리고, 속성 표현에 있어서도 일부 제외된 부분이 있어서 제한이 되었다.

표 1. 기존연구 특징

구 분		변환기법		특징	지원툴
		매핑	변환		
Feature → UML 변환	DSPLU	수동	수동	모델링 전문가에 의해 변환 실시	UML툴
	GUDFD	수동	자동	모델 구현을 위해 완벽한 UML 모델이 필요, 과다시간 소요	feature-UML
Feature → OWL 변환	OAACVF	수동	수동	속성 표현이 제한됨	protégé
OWL → Class 변환	OUTM	자동	반 자동	속성 표현이 제한됨	XML파서 protégé UML툴

결국, 기존의 feature 모델의 class 모델로의 변환 연구는 그 의미가 명확하지 않으며 완전한 변환 방법을 제시 하고 있지 않다. 그래서, 본 논문은 표1 에서 보여주는 속성 매핑과 변환을 자동화 하였고, Ontology를 이용하여 feature 모델의 의미를 명확히 하고, 쉽게 이해하고 손쉽게 수정, 편집을 할 수 있도록 class 모델로의 변환 방법을 제시 한다.

3. feature – class 모델 변환

3.1 feature- class 모델 변환 프로세스

feature를 class모델로 변환하기에 앞서 의미를 명확히 표현하기 위해 Ontology를 이용하여 명세 한다. 그림1은 feature를 class diagram으로 변환하는 과정을 제시한다.

1단계에서는 feature를 매핑 규칙을 이용하여 OWL로 명세를 한다. 여기에서는 Ontology 변환을 위해 무료로 쉽게 사용할 수 있는 protégé 3.2 프로그램을 이용하였다.

2단계에서는 OWL로 만들어진 문서를 XMI문서 생성기를 이용하여 XMI(XML Metadata Interchange) 문서로 만든다. XMI 문서는 MOF와 호환성이 있는 모델들을 XML 문서 형태로 표현하기 위한 표준이다[6]. 즉, UML로 설계된 모델을 다른 포맷의 모델로 변환

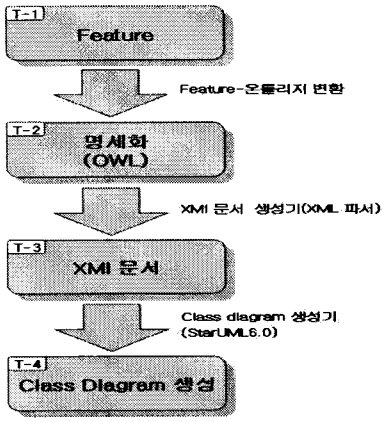

그림 1. feature- class 모델 변환 프로세스

하고 교환하기 위해 사용되는 표준이다. 그리고, OWL은 XML 기반의 언어 이므로 XML 파서를 이용하면 쉽게 XMI로 변환이 가능하다.

3단계에서는 만들어진 XMI 파일을 UML툴을 이용하여 UML class diagram으로 만든다. 그리고 역으로 UML class diagram을 XMI파일로 export하여 Ontology 파일을 만들 수 도 있다.

3.2 매핑 규칙

그림1의 feature-class 모델 변환 프로세스를 적용하기 위해서는 feature 모델과 OWL, 그리고 class모델 요소들간의 매핑 테이블이 필요하다.

그림2은 feature - class 모델 변환 규칙을 만드는 과정이다.

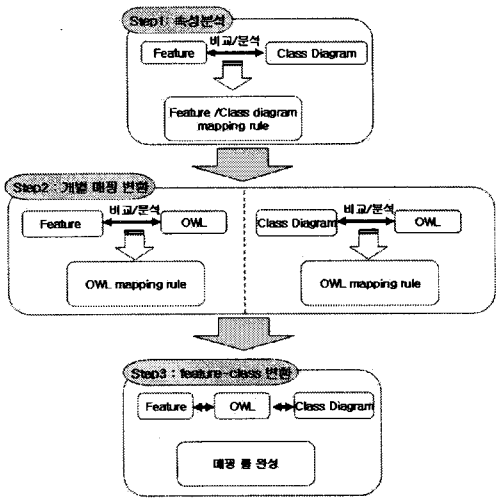

그림 2. 매핑 규칙 프로세스

1단계 속성분석에서는 feature 모델과 class 모델 간의 속성을 비교 분석하여 직접 feature 모델과 class 모델간의 매핑 규칙을 만들고, 2단계 개별 속성

변환에서는 feature 모델과 OWL 그리고 class 모델과 OWL과의 매핑 규칙을 만들어 OWL과의 관계를 정립하는 단계이다. 마지막으로 3단계 feature-class 모델 변환에서는 1단계와 2단계에서 만들어진 규칙을 이용하여 feature 모델, OWL, class 모델간의 변환 규칙을 완성한다.

3.3 속성 분석

3.3.1 feature 모델

본 논문에서는 그림 3에서 제시한 feature 모델을 근간으로 사용한다[1].

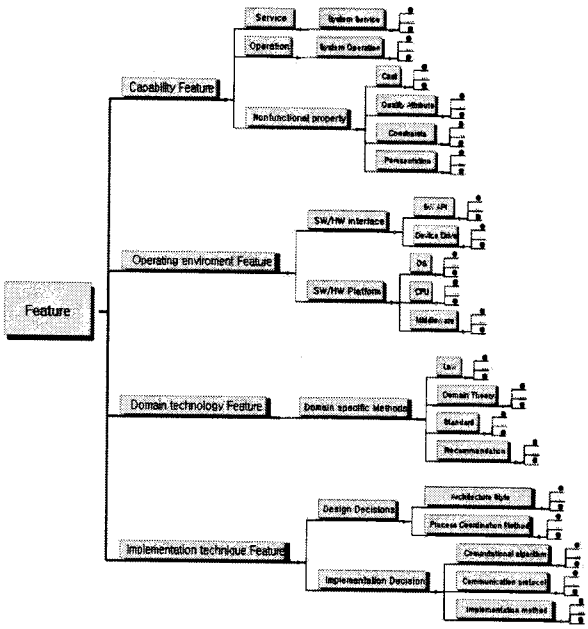

그림 3. feature 모델

feature 모델의 속성은 크게 concept 과 relationship의 이원화 형태로 표현 되는데 표2는 개별 feature 모델의 속성을 정의하는 feature 모델 속성 리스트를 보여주고 있다.

표 2. feature 모델 속성 리스트

구 분	세부속성	설 명
식별형 속성	명칭	식별된 feature의 명시적 이름
	유사 feature명	일반적 의미에서 feature와 동일하거나 유사한 개념의 feature 이름
	분류	도메인 내에서 부모 feature와 관계(상위 feature 이름)
선택형 속성	필수	도메인 내의 일련의 feature들 중, 반드시 존재해야 하는 feature
	선택	도메인 내에서 일련의 feature들 중, 필수적이지 않은 선택 가능한 feature
	양자택일	도메인 내의 일련의 feature들 중, 양자 택일적으로 선택이 가능한 feature
관리형 속성	식별자	도메인 내에서의 feature 고유한 ID number
	등록기관	feature를 등록한 조직, 그룹 또는 회사명

표3은 feature모델들간의 관계를 나타내는 관계형 속성 리스트를 나타낸다.

표 3. feature 모델 관계 리스트

구 분	세부속성	설 명
관계형 속성	Part-of	상위 feature의 관계(전체의 관계)로 generalization 과 동치
	Has-part	하위 feature의 관계(부분의 관계)로 composed of 또는 specialization관계와 동치

3.3.2 class 모델

class 모델은 UML 모델링의 한 방법으로 객체들의 타입 표현하고, 클래스 간의 정적인 관계를 나타낸다. 표4는 class 모델의 세부 속성을 나타낸 리스트이다.

표 4. class 모델 속성 리스트

세부속성	설 명
class	비슷한 속성과 공통적인 행동 수단을 지닌 것들의 어떠한 범주 혹은 그룹을 말한다
attribute	클래스에 속한 특성에 이름을 붙인 것으로 이것이 가질 수 있는 값의 범위를 설정한다
operation	객체에 요청 할 수 있는 행동을 말한다.
responsibility	특정한 타입 혹은 클래스가 해야 하는 일을 설명해 둔 것
constraints	중괄호{ } 안에 자유 형식의 텍스트가 들어 있는 형태로써, 클래스가 따라야 하는 규칙 기술
sterotype	<< >>을 사용하여 확장된 영역을 표시

표5는 class 모델의 관계형 속성 리스트를 나타낸 표이다.

표 5. class 모델 관계형 속성 리스트

세부속성	설 명
Association(연관)	두 클래스 간의 관계
Aggregation(집합)	전체와 부분의 관계
Composition	Aggregation의 특수한 경우, 강한 소유의 표시
Generalization(일반화)	일반화된 사물과 특수화된 사물
Realization(실체화)	정의와 구현관계
Dependency	의존형 관계

3.3.3 feature - class 모델 속성 매핑

3.3.1과 3.3.2에서 feature 모델과 class 모델의 속성을 분석한 자료 이용하여 표6과 같은 매핑 테이블을 만들 수 있다. 예로 feature 모델에서 식별된 feature synset은 class 모델의 attribute로 나타낼 수 있다.

표 6. feature - class diagram 속성 매핑 테이블

Feature		Class Diagram
식별형 속성	name	class
	feature synset	attribute
	classification	attribute
선택형 속성	mandatory	attribute
	optional	attribute
	alternative	attribute
관리형 속성	identifier	attribute
	registration authority	attribute
관계형 속성	part-of	Relationship: Generalization
	has-part	Relationship: Aggregation /Composition

3.4 개별 매핑 변환

3.1.1에서 정립된 요소들의 의미를 명확하게 하기 위해 feature 모델의 owl 변환을 한다.

3.4.1 feature 모델의 owl 변환

그림2의 2단계 개별 매핑 변환에서는 feature 모델과 와 OWL 그리고 OWL 과 class 모델과의 관계를 정의 한다.

표7은 feature 모델과 OWL과의 관계를 나타낸 표이다. 한 예로 feature 모델의 feature synset 은 feature의 유사어휘를 나타내는 속성으로 OWL의 owl:DatatypeProperty로 나타낼 수 있다.

표 7. feature - owl 모델 속성 매핑 테이블

Feature		OWL
식별형 속성	name	owl:class
	feature synset	owl:DatatypeProperty
	classification	owl:DatatypeProperty
선택형 속성	mandatory	owl:DatatypeProperty
	optional	owl:DatatypeProperty
	alternative	owl:DatatypeProperty
관리형 속성	identifier	owl:DatatypeProperty
	registration authority	owl:DatatypeProperty
관계형 속성	part-of	owl:subclassOf
	has-part	owl:UnionOf

표8은 OWL과 class 모델과의 관계를 나타낸 표이다. 하나의 예로서 OWL의 owl:DatatypeProperty는 데이터의 속성을 나타내는 요소로 class 모델의 attribute로 나타낼 수 있다.

표 8. owl - class 모델 속성 매핑 테이블

OWL	Class Diagram
owl:class	class
owl:DatatypeProperty	attribute
owl:objectProperty	Association
owl:onProperty	Association
owl:cardinality	Association
owl:restriction	Association
owl:subclassOf	Generalization
owl:UnionOf	Aggregation/Composition

3.5 feature - class 모델 변환

3.4에서 정의된 feature, OWL, class 모델과의 관계를 정리하여 도표화 하면 표9와 같다.

표 9. feature - owl - class 모델 속성 매핑 테이블

Feature		OWL	Class Diagram
식별형 속성	name	owl:class	class
	feature synset	owl:DatatypeProperty	attribute
	classification	owl:DatatypeProperty	attribute
선택형 속성	mandatory	owl:DatatypeProperty	attribute
	optional	owl:DatatypeProperty	attribute
	alternative	owl:DatatypeProperty	attribute
관리형 속성	identifier	owl:DatatypeProperty	attribute
	registration authority	owl:DatatypeProperty	attribute
관계형 속성	part-of	owl:subclassOf	Generalization
	has-part	owl:UnionOf	Aggregation/ Composition

4. 사례적용

본 논문에서는 그림3에서 제시한 프로세스를 전자결제체계에 적용하여 전자결재 체계를 OWL로 명세하여 XMI파일, UML class diagram으로 변환을 보여준다.

그림4는 그림2의 feature 모델을 수정하여 전자결재 체계를 나타낸 것이다.

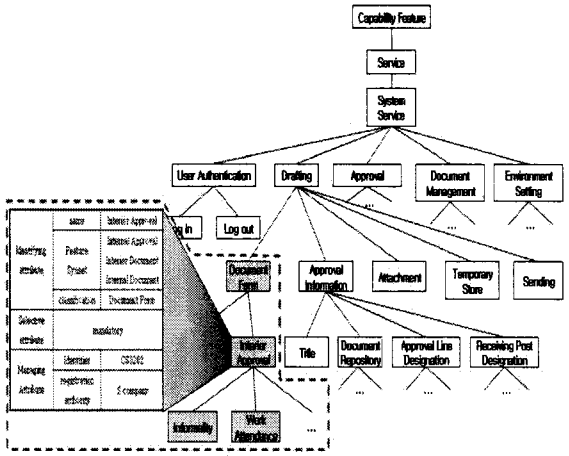

그림 4. 전자결재 feature model

그림4에서 점선으로 된 전자결재 체계 feature 모델을 Ontology로 표현하면, 그림 5와 같이 된다. 한 예로 전자결재 체계 feature 모델의 관계형 속성 part-of 는 OWL의 owl:subclassOf로 표현 되고, 속성은 owl:DatatypeProperty로 표현 한다.

```
...
<owl:Class rdf:ID="InteriorApproval">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >/</rdfs:comment>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="DocumentForm"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Informality">
  <rdfs:subClassOf rdf:resource="#InteriorApproval"/>
</owl:Class>
<owl:Class rdf:ID="WorkAttendance">
  <rdfs:subClassOf rdf:resource="#InteriorApproval"/>
</owl:Class>
<owl:Class rdf:ID="ExteriorApproval">
  <rdfs:subClassOf rdf:resource="#DocumentForm"/>
</owl:Class>
<owl:DatatypeProperty rdf:ID="selectiveValue">
  <rdfs:domain rdf:resource="#InteriorApproval"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf rdf:parseType="Resource">
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >mandatory</rdf:first>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:DatatypeProperty>
...
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3, Build 418)  http://protege.stanford.edu -->
```

그림 5. 전자결재 OWL 파일

이것을 XMI 변환 프로그램을 이용하여 그림6 과 같이 XMI 파일이 만들어 진다.

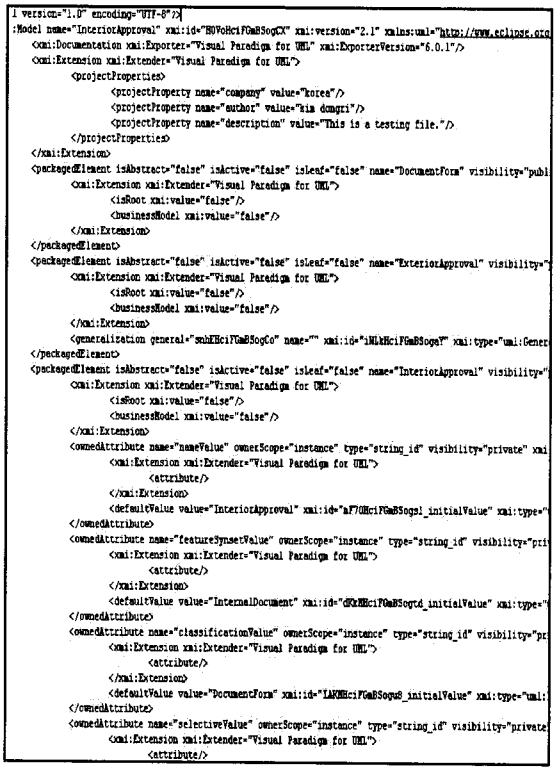

그림 6. 전자결재 XMI 파일

그리고 그림7은 위에서 만들어진 XMI 파일을 이용하여 전자 결재체계를 class diagram으로 나타낸 그림이다.

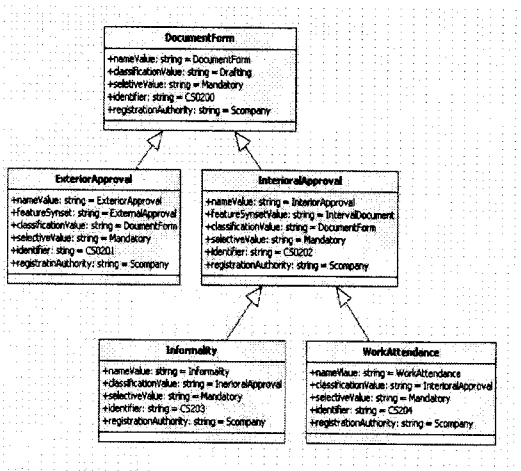

그림 7. 전자결재 class diagram

그림7에서 볼 수 있듯이 Ontology 기반의 feature 모델이 class 모델로 변화되면 한눈에 볼 수 있어 쉽게 이해 할 수 있게 되다. Ontology 기반의 feature 모델들은 시각적인 표현이 미흡하기 때문에 개별 feature 모델이나 feature 모델 전반에 걸친 이해를 하기 어렵다는 문제점이 있었다. 하지만, 이처럼 UML class diagram을 이용하면 그 문제점을 해결 할 수가 있다.

이렇게 만들어진 class 모델은 필요에 따라 확장을 하거나 수정할 수 있고, 역으로 class 모델에서 수정한 내용도 다시 XMI로 export 하여 수정된 feature Ontology의 생성이 가능하다.

5. 결론

본 논문에서는 Ontology 기반의 feature모델을 XMI를 이용하여 class 모델로 변환하는 방법을 제시하였다. 이렇게 함으로써 기 구축된 feature 모델을 UML class 모델로의 변환을 통해 재사용을 증대 시킬 수 있고, feature 모델과 class 모델을 Ontology를 사용하여 의미를 명확히 정립 할 수 있으며, 이를 통해 feature 모델의 이해와 수정을 용이하게 할 수 있다. 또한 역으로 본 기법을 적용하면 class 모델을 feature 모델로 변환 시킬 수 있어 자유로운 변환 및 공유가 가능하고, 또한 모델의 재사용에 기여 할 수 있다.

향후 연구로는 Ontology로 표현된 feature의 다른 UML 모델링으로 변환하는 기법과 Ontology 기반의 feature의 효율적인 재사용 방법에 대한 연구를 진행하고자 한다.

참고문헌

[1]Kang, K. C., Kim, S., Lee, J.,et al, " FORM: A Feature-Oriented Reuse Method With Domain Specific Reference Architecture," Annals of Software Engineering, Vol. 5, pp. 143-168, 1998.

[2]이순복, 김진우, 송치양, 김영갑, 권주홈, 이태웅, 김현석, 백두권, 소프트웨어 제품 계열공학의 Ontology 기반 휘처 공통성 및 가변석 분석 기법, 정보과학회논문지, Vol. 34, pp.196-211, 2007.

[3]Hassan Gomaa, Designing Software Product Lines with UML, Software Engineering Workshop Tutorial, April 2005.

[4] Xin Liu, Generating UML Diagrams using Feature Digagrams for Software Product Line, Master Thesis, pp. 1- 72, 2006.

[5] 이지홍, 양진혁, 손종수, 정인정, UML을 이용한 효율적인 Ontology 재사용에 관한 연구, 한국지능정보시스템학회, pp. 265-269, 2006.

[6] Dragan Durić, MDA-based Ontology Infrastructure, ComSIS, Vol.1, pp. 91-116, 2004.