소프트웨어 제품 계열 공학의 온톨로지 기반 휘처 공통성 및 가변성 분석 기법

(Ontology-based Approach to Analyzing Commonality and Variability of Features in the Software Product Line Engineering)

이 순 복 † 김 진 우 † 송 치 양 †† 김 영 갑 †††
(Soon-Bok Lee) (Jin-Woo Kim) (Chee-Yang Song) (Young-Gab Kim)
권 주 흠 †††† 이 태 웅 † 김 현 석 † 백 두 권 †††††
(JuHum Kwon) (Tae-Woong Lee) (Hyun-Seok Kim) (Doo-Kwon Baik)

요 약 제품 계열 공학에서 제품의 공통성 및 가변성 분석을 결정짓게 하는 기준인 휘처(feature) 분석에 대한 기존 연구는 개발자의 직관이나 도메인 전문가의 경험에 근간으로 분석 기준이 객관적이지 못하며, 비정형적인 휘처 분석으로 인한 이해 당사자(stakeholder)의 공통된 휘처의 이해 부족 및 불명확한 휘처를 추출하는 문제점이 있었고, 기 개발된 소프트웨어에서 사용된 휘처의 재사용 개념이 부족했었다.

본 논문에서는 특정 도메인의 휘처 모델을 온톨로지로 변환하여 의미 기반 유사성 분석 기준에 의해 휘처의 공통성과 가변성을 추출하는 기법을 제시한다. 이를 위해, 먼저 공통된 휘처 중심의 메타 휘처 모델 기반으로 휘처의 속성을 정립하고, 메타 모델에 준거하여 휘처 모델을 생성하여 온톨로지로 변환 후, 휘처 온톨로지 리포지토리(Repository)에 저장한다. 이후, 동일 제품 계열 도메인의 휘처 모델 구축 시, 기존 생성 모델과 온톨로지의 의미 기반 유사성 비교 분석 기법을 통해 휘처의 공통성과 가변성을 추출하는 것이다. 또한 유사성 비교 알고리즘을 툴로 구현하였으며, 전자 결재 시스템 도메인의 실험 및 평가를 통해 효과성을 보인다.

본 기법을 통해 메타 휘처 모델의 구문적 정립으로 이해성과 정확성을 제고시켜 고품질의 휘처 모델을 구축할 수 있으며, 온톨로지의 의미 기반 매핑으로 휘처의 공통성 및 가변성 추출을 정형화할 수 있고, 재사용성을 향상시킬 수 있다.

키워드 : 제품 계열 공학, 휘처, 온톨로지, 공통성과 가변성

Abstract *In the Product Line Engineering (PLE), current studies about an analysis of the feature have uncertain and ad-hoc criteria of analysis based on developer's intuition or domain expert's heuristic approach and difficulty to extract explicit features from a product in a product line because the stakeholders lack comprehensive understanding of the features in feature modeling. Therefore, this paper proposes a model of the analyzing commonality and variability of the feature based on the Ontology. The proposed model in this paper suggests two approaches in order to solve the problems mentioned above: First, the model explicitly expresses the feature by making an individual feature attribute list based on the meta feature modeling to understand common feature. Second, the model projects an analysis model of commonality and variability using the semantic similarity between features based on the Ontology to the stakeholders. The main contribution of this paper is to improve*

· 이 연구에 참여한 연구자는 '2단계 BK 21 사업'의 지원을 받았음
† 학생회원 : 고려대학교 컴퓨터학과
lsb0510@korea.ac.kr
pkm311@korea.ac.kr
minote@korea.ac.kr
hskim@formal.korea.ac.kr
†† 정 회 원 : 상주대학교 소프트웨어공학과
cysong@sangju.ac.kr
††† 정 회 원 : 고려대학교 정보보호대학원 교수
always@korea.ac.kr
†††† 정 회 원 : 공군 중앙전산소
jkweon@gmail.com
††††† 종신회원 : 고려대학교 컴퓨터학과 교수
baikdk@korea.ac.kr
(Corresponding author임)
논문접수 : 2006년 8월 28일
심사완료 : 2006년 12월 12일

the reusability of distinguished features on developing products of same line henceforth.

Key words : Product Line Engineering, feature attribute list, feature-Ontology similarity graph (FOSG), meta feature repository (MFR), commonality, variability

1. 서 론

현재 제품 계열 공학은 유사한 기능을 지닌 소프트웨어 제품 개발에 있어 사용자의 요구 사항에 적합하고, 시장의 적시성(time to market), 생산성(productivity), 비용 감소(cost reduction) 측면에서 이슈화 됨에 따라 재사용성이 중요시 되고 있다[1].

오늘날 소프트웨어 재사용(reusability)의 초점은 소스 코드의 재사용에서 소프트웨어 설계(design)의 재사용으로, 설계 재사용에서 다시 도메인 공학(domain engineering)에 기반을 둔 재사용으로 그 패러다임이 옮겨져 왔다. 이러한 변화는 소스 코드의 재사용을 위해서는 설계의 재사용이 선행되어야 했고, 또한 설계를 재사용하기 위해서는 도메인 공학을 통해서 어떤 도메인에서 공통적으로 쓰이는 도메인 자산(domain asset)을 먼저 개발하여야 했기 때문이다. 따라서 소프트웨어 재사용 연구는 도메인 분석 및 공학에 중점을 두고 수행되어 왔다[2-5]. 제품 계열에서 핵심 자산의 하나인 도메인 요구 사항을 관리하는 방법에 대해서 많은 연구가 진행되었지만, 도메인 분석의 방법상 공통성(commonality)과 가변성(variability)을 추출하는 휘처에 대해 객관화된 근거 없이 개발자의 직관(intuition)이나 도메인 전문가(domain expert)의 경험에 의해 그 결정을 내리고 있어 분석하는 기준이 명확하지 않으며, 휘처 간의 상호 충돌(conflict)과 실제 적용 기준의 불명확함으로 이해 당사자간의 많은 이해의 오류가 발생하고 있다[6-9].

도메인 공학 프로세스와 소프트웨어 개발 프로세스 상에서 분석된 휘처는 체계적으로 재사용성이 보장되어야 하지만, 현재 도메인 요구 사항 분석 방법에서의 휘처는 비 정형적(ad-hoc)이고 그 실체가 명확하지 않아 적용이 어렵다. 지금까지 많은 연구가 제품 계열 공학과 관련하여 도메인 요구 사항을 분석하고 있지만, 그것은 특정 수식 및 매트릭스의 조합으로 결국 휘처에 대한 요구 사항 분석 적용이 어렵고[10,11], 도메인에서 사용되는 용어들이 각기 달라 서로의 의미간 구분이 명확하지 않으며, 각각 유사 도메인 안에서의 상이한 도메인 분석틀(framework)로 한번 분석된 도메인 휘처는 또다시 유사 도메인을 개발할 때 기존의 분석된 도메인의 재사용이 곤란하다. 더불어, 개발해야 될 어플리케이션이 복잡하고 대규모일 경우에는 단순히 도메인의 매트릭스나 사람의 직관에 의한 비교는 많은 시간과 비용을 수반하는 문제점을 내포하고 있다[8,9]. 왜냐하면, 정형화 되어 있지 않는 도메인 분석가의 경험적인 공통성 및 가변성 분석 방법은 규모가 크고 고비용인 소프트웨어를 개발할 시에는 재사용성이 보장되지 않으며, 추후 통합을 위해서 더 많은 비용의 미들웨어를 구축해야 하는 비용 소모를 내포하고 있기 때문이다.

따라서, 본 논문에서는 단순히 기능, 비 기능 중심으로 표현을 하는데 그치고 있는 휘처의 개념을 명확히 실체화하여 휘처 속성 기반의 메타 모델을 구축하고, 휘처에 대한 이해의 오류(the error of shared understanding) 및 휘처의 의미적 모호성(ambiguity)에 관한 문제점을 휘처 온톨로지(feature-Ontology)의 의미 유사성 매핑을 통해 해결할 수 있는 메커니즘을 제공하는 것이다. 이는 휘처 온톨로지 그래프 안에서 온톨로지에 기반한 의미 매핑 메커니즘을 사용하여 의미적 중심의 개념 유사성(semantic similarity)을 비교하여 공통성과 가변성 분석을 통해, 도메인 전문가들이 원하는 정확성 기반의 명시적 휘처를 추출함으로써 도메인의 재사용성을 향상시키는 것이다.

본 논문의 구성은 다음과 같다. 제 2장은 관련 연구로 기존 제품 계열 공학에서의 휘처 모델과 온톨로지와의 접목 관계를 살펴보고, 제 3장은 본 논문의 접근 연구 모형으로 기존 도메인 분석 상의 문제점에 대한 해결로서 온톨로지 기반 휘처의 공통성 및 가변성 추출 모델을 제시한다. 제 4장은 접근 모델의 상세 기법으로, 명시적 휘처 분석을 위해 메타 휘처 모델과 온톨로지 유사성 매핑 알고리즘을 제안한다. 제 5장은 Java와 OWL (Web Ontology Language)을 이용하여 앞서 제시한 휘처 분석 모델을 의미 유사성 매핑 툴로의 구현을 보여주며, 제 6장에서는 휘처의 공통성 및 가변성을 분석하는 실험 및 평가를 제시한다.

2. 관련 연구

S/W 제품 계열 공학에서의 휘처의 공통성과 가변성 분석에 관련된 휘처 분석과 모델링 방법, 온톨로지 명세 기법, 그리고 휘처와 온톨로지와의 관계에 대해 분석한다.

2.1 제품 계열 공학에서의 휘처

제품 계열 공학에서 휘처는 도메인 분석에 있어 중요한 요소가 되었고, 휘처를 명시적으로 표현하기 위해 휘처 모델링에 관한 연구가 진행되었다. 기존에 연구되어 온 휘처의 분석 방법과 휘처 모델링 방법에 대해 살펴본다.

제품 계열 공학에서 휘처의 대표적 분석 방법에는 FODA(Feature-Oriented Domain Analysis)가 있다.

FODA의 도메인 분석 방법은 휘처를 중심으로 도메인에서의 공통성과 가변성을 분석하고 있다[3,8,9,12-16]. 즉, 시스템의 집합 중에서 주도적이고 독특한 휘처를 식별하는 것을 바탕으로 도메인을 분석한다. 여기서 휘처는 "구현, 테스트되고 배포 및 유지되어야 하는 기능적/비기능적 추상화를 말하는 것으로 요구 사항이나 특징적인 기능"을 말한다[9]. 또한 FODA에서의 휘처 다이어그램(FD: feature diagram)은 공통성 및 가변성 분석에서 사용된 가장 범용적 접근 방법이며, 하나의 제품 계열에서 공통 휘처와 가변 휘처를 휘처들의 관계로 표현하는 방식이다. 휘처 모델의 모델링 언어인 FD는 노드와 에지 구조로 구성된 트리 구조이다. 각 노드는 하나의 휘처로 표현되고 각 에지는 에지에 의해 연결된 두 가지 휘처 사이의 관계로 나타내어 진다. 여기서 공통 휘처는 하나의 제품 계열에서 모든 제품 멤버에 포함될 필수적 휘처로 정의되고, 가변 휘처는 반드시 포함되지 않고 선택되어 구성될 수 있는 휘처로 정의된다[9].

휘처의 재사용에 초점을 맞춘 방법에는 FORM (Feature-Oriented Reuse Method)이 있다[13]. FORM의 핵심은 도메인의 휘처를 분석하고, 재사용 가능한 도메인 제품의 개발 시 해당 휘처를 사용하는 데 있다. 재사용을 위한 첫 단계로 FODA에서 정의한 4가지 계층인 능력(capabilities), 운용 환경(operating environments), 도메인 기술(domain technologies), 구현 기술(implementation techniques) 상에서 하나의 특별한 도메인 제품의 공통성을 분석하였다. 휘처 모델을 바탕으로 AND/OR 그래프를 이용하여 공통성을 추출하는데, AND 노드는 제품의 필수적인 휘처를 말하고, OR 노드는 다른 제품으로 선택될 수 있는 양자 택일의 휘처를 말한다[12].

휘처 모델링 방법에는 Reuse-Driven Software Engineering Business(RSEB)의 프로세스를 통합한 Featu-RSEB(the Featured FSEB)방법이 있다[17]. 이 방법은 도메인 공학 과정으로 휘처 모델을 생성하는 활동과 응용 공학 과정으로 유스케이스 모델을 생성하는 과정으로 이원화되고 병행 수행된다. 이후에 소프트웨어 제품 계열 개발에서 가변성을 표현하기 위한 연구에 휘처가 사용되었다[18]. [18]에서 Feature Description Language (FDL)는 휘처 그래프를 텍스트 형태로 표현하기 위해 개발된 언어로서 제품 계열 아키텍처의 가변성을 추출, 기술하기 위하여 휘처 그래프를 텍스트 형태로 바꾼 것이다. 다른 방법으로 시맨틱 웹(Semantic Web)을 이용한 접근 방법이 있다[19]. 이 방법은 휘처 다이어그램에서 휘처 사이 관계를 명확히 식별하기 위해 OWL DL을 이용하여 필수(mandatory), 선택(optional), 양자택일(alternative), 또는(or), 요구(requires), 배제(excludes)의 관계를 가지는 휘처 모델을 시각적으로 표현한다. 또 다른 방법으로 온톨로지 기반 휘처 모델링 방법이 있다[20]. 이 방법은 휘처 모델에서 행위(action)와 용어(term)는 OWL Class (owl:Class)로 정의하고, 패싯(facet)은 OWL property (owl:Property)로 정의하며, 행위와 용어의 관계는 subClassOf로 표현하고 있다.

2.2 의미적 유사성 측정을 위한 온톨로지

온톨로지는 "하나의 지식에 대해 공유할 수 있는 이해를 바탕으로 하는 명시적 명세(Formal specification of a shared understanding for a knowledge)"를 말한다[21]. 즉, 특정 분야의 지식을 표현하기 위한 기본 지식 체계를 말한다. 이러한 정의에서 알 수 있듯이, 온톨로지의 필요성은 사람과 사람 사이, 소프트웨어 에이전트 사이, 혹은 사람과 소프트웨어 에이전트 사이에서 설명이 가능한 정보 구조의 공통된 이해를 공유하는 것이다. 또한 상호 운용성(interoperability)을 위한 도메인 지식의 재사용을 가능하게 하는 것이다.

지식 도메인의 공통된 이해를 바탕으로 한 재사용에 관한 연구 중, 온톨로지 의미적 유사성 측정의 방법은 계층적 지식 표현 구조에 가장 알맞은 형태로 적용될 수 있다. 온톨로지의 개념(concept)과 개념 사이의 관계를 에지(edge)로 정의하고, 그 에지 수의 계산에 따라 의미 유사 정도를 측정한다[22-24]. 즉, 온톨로지는 개념 간의 의미적 유사 여부 판명에 유용한 기법이다. 또한 도메인 요구사항의 도출 및 분석에서 도메인의 공통성 및 가변성의 표현과 도메인 요구사항의 불완전성(incompleteness), 불일치(inconsistency)의 문제 식별 및 관리에 있어서도 온톨로지를 이용하고 있고, 소프트웨어 제품군에서 제품의 가변성을 관리하는데 있어 온톨로지를 이용하여, 지식 관리 시스템과 온톨로지의 통합을 통한 지식의 재사용을 높이고 있다[25,26].

2.3 휘처 분석과 온톨로지의 접목 관계

온톨로지에서의 개념은 도메인이 내포하고 있는 정보를 동일 의미의 용어로 표현한다는 측면에서 휘처와 유사하고, 구조는 특정 도메인 분야의 지식을 표현하는 체계를 제공한다는 측면에서 트리 분류법(taxonomy)과 유사하다. 따라서, 온톨로지로 정의된 휘처의 상호 연관 관계를 체계적으로 정의할 수 있다. 즉, 온톨로지는 개념과 관계의 두 가지 요소로 구성 되고, 개념은 온톨로지 내에 정의된 휘처를 말하며, 관계는 온톨로지 내에 정의된 개념 간의 연관 관계를 정의하는 이원화 관계로 적용된다[9,19-22,27,28]. 아래 표 1에서 제품 계열 공학의 FD와 온톨로지와의 관계성을 볼 수 있다.

표 1에서 FD와 온톨로지의 연관 관계를 기반하여 각각의 휘처 노드를 개념으로 접목하고, 휘처간의 관계를 개념과 개념간의 관계로 적용하면, 개발 도메인에 대하

표 1 제품 계열 공학의 FD와 온톨로지와의 관계성

구 분	FD(feature diagram)	온톨로지
개념적 측면	- 휘처 - 요구사항 또는 특징적 기능의 추상화	- 개념(concept) (= OWL의 class) - 개념적 유사 정도를 사용하여 단어의 정보 또는 의미적 개념을 표현
구조적 측면	- 계층적 구조를 나타내는 트리 분류법	- 지식 정보를 나타내는 체계적 구조
관계적 측면	- Composed-of - Generalization / Specification - Implemented-by	- Is-A / Has-A - Part-of / Has-part - Member-of / Has-member - Substance-of / Has-substance

여 온톨로지 구축을 통한 의미 유사성 그래프를 생성할 수 있으며[22-24,29,30], 이를 통해 휘처간 유사성을 분석할 수 있다.

2.4 기존 휘처 중심 도메인 분석 방법의 문제점

FODA에서 "휘처는 구현, 테스트되고 배포 및 유지되어야 하는 기능적/비기능적 추상화를 말하는 것으로 요구 사항이나 특징적인 기능"을 말한다[9]. 하지만 휘처는 일반적인 객체 지향 방법론에서 말하는 객체와는 다른 특징적인 것을 가지고 있다. 휘처 도메인 분석 방법의 문제점은 크게 아래와 같이 구분되며 그 문제점은 본 논문이 해결하고자 하는 지향점이 된다.

◦휘처의 공통성과 가변성 분리 기준의 불명확

FD는 도메인 전문가에 의해 직관이나 경험에 의해 그려진다. 따라서, 휘처의 관계는 다이어그램에서 표현되고 정의된 것들보다 훨씬 복잡하고 휘처 간의 의존성(dependency)이 커진다면 다이어그램의 복잡성(complexity)은 매우 증대된다. FD안에서 공통성과 가변성을 구분하는 준거인 휘처는 명확히 식별되지 않을 뿐만 아니라 휘처와 휘처간의 관계가 명시적으로 정의되어 있지 않다. 이는 중복 휘처의 포함으로 인한 정확한 휘처 모델의 정립과 재사용에 저해가 된다.

이처럼 도메인을 분석하는 분야에 휘처는 다방면에서 사용되었지만 휘처의 속성을 결정짓게 하는 논리적인 근거와 휘처의 실체를 정의하는 메타 모델에 대해서는 제시하고 있는 것이 없었고, 지금까지의 휘처 분류는 모두 도메인 전문가의 경험이나 직관에 의존한다. 또한 소프트웨어 제품 계열에서 핵심 자산으로서 공통성과 가변성을 분석하는 연구도 진행되어 왔지만, PR(Primitive Requirement)과 문맥(context)의 속성을 확률적으로 결정하는 매트릭스의 정렬로 공통성과 가변성을 결정하는 방법은 실제 적용이 어려운 추상적인 절차로서 그 한계를 내포하고 있다[31]. 즉, 휘처간의 의존성을 표현하는 방법으로 휘처를 순행과 역행의 관점으로 구분하여 매트릭스 기반 데이타 구조를 저장하여 표현하는 휘처 의존성 매트릭스(feature dependency matrix)는 개발 도메인이 대규모인 경우에는 복잡도가 증대되어 실제 적용하기가 용이치 않다.

• 비 명시적 휘처 추출

예로, 분산 미팅 스케줄러 시스템 도메인 상에서 발생 가능한 기존 휘처 모델의 문제점은 다음과 같다[3,11,12,15].

첫째, 휘처에 대한 이해의 오류이다. 그림 1의 (a)에서 보는 바와 같이 서로 다른 의미를 가진 두 휘처가

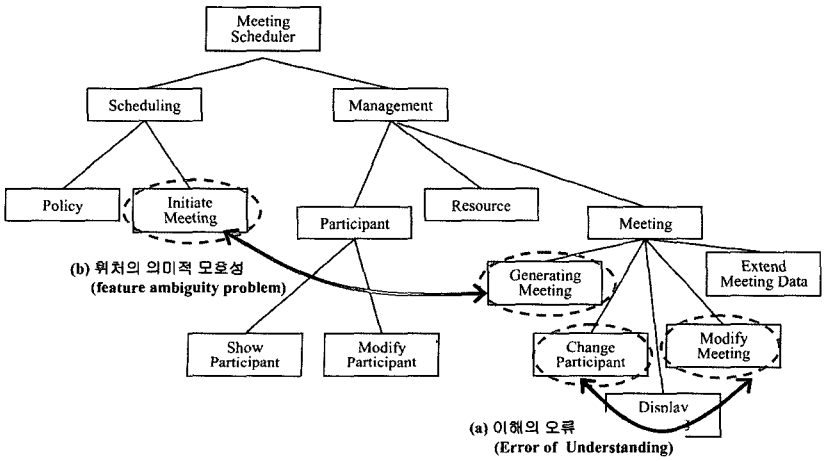

그림 1 기존 휘처 모델의 문제점 및 한계

비 정형적인 명세로 이해의 오류를 발생시켜 동일한 휘처로 이해될 수가 있다. 예를 들면, 미팅 참가자의 정보를 수정하기 위한 'ModifyParticipant'와 미팅 참가자들을 수정하기 위한 'ChangeParticipant' 휘처 사이에 이해의 오류가 발생하는 문제점이 있다.

둘째, 휘처의 의미적인 모호성에 관한 문제점이다. 그림1의 (b)에서 휘처 분석가에 따라서 동일한 의미를 가지는 휘처가 다른 표기법을 가지고 명세 되는 경우가 발생할 수 있다. 이 때 의미가 모호한 표기법을 사용한다면 동일한 의미의 휘처임에도 불구하고, 서로 다른 휘처로 인식되어 휘처 모델에 동시에 존재하는 중복성 문제를 야기 시킨다. 예를 들면, 분석가에 따라서 미팅을 생성하는 휘처에 대하여 'Initiate'와 'Generate'라는 서로 다른 술어를 사용할 경우 의미의 모호성이 발생하는 문제점이 있다.

휘처는 그 속성에 따라 명시적으로 정의 및 분해가 되어 도메인 분석가들의 공통적인 이해를 통한 명시적인 휘처를 추출하여야 한다. 하지만 기존의 휘처 분석은 명시적이고, 의미에 기반한 휘처 추출에 대하여 한계가 있다. 따라서 휘처의 실체를 정확하게 정의하는 메타 모델이 필요하다.

3. 접근 연구 모형

본 장에서는 기존의 휘처 중심 도메인 분석 방법의 문제점들을 해결하기 위한 온톨로지 기반의 휘처 공통성 및 가변성 추출 모델을 기술한다.

휘처를 정확하게 정의하고 휘처의 공통성과 가변성을 추출하기 위한 접근 과정은 예비 1 단계, 메타 휘처 모델링 2 단계, 온톨로지 의미 유사성 매핑 3 단계, 휘처 재사용 4 단계로 이루어지고 본 논문은 휘처의 공통성과 가변성 분석을 위한 2, 3 단계에 초점을 둔다.

단계 1은 예비 단계(preliminary stage)로서, 개발 해야 할 도메인을 선택하고 그 범위를 결정하며, 이해 당사자를 중심으로 휘처 도메인 전문가 분석팀을 구성하는 단계이다.

단계 2는 명시적으로 휘처를 실체화 및 정의하는 메타 휘처 모델링(meta feature modeling) 단계이다. FORM[12]에서 분류한 4가지 계층별 휘처 분류를 기반으로 휘처의 유형을 개념과 관계로 분석한 메타 휘처에 대한 속성별 목록을 정의하여, 기존 FORM의 휘처를 실체화하여 확장한다. 또한, 개별 휘처들의 관계를 온톨로지 언어인 OWL로 적용하기 위하여 part-of, has-part, sibling, self 관계로 분류한다.

단계 3은 온톨로지를 이용한 휘처의 의미 유사성 매핑[22-24,28-30] 단계로서, 초기 메타 휘처 리포지토리(MFR)에 저장될 제품 계열 도메인의 표준 휘처 온톨로지를 구축하고, 이후 동일 제품 계열 내의 개발해야 할 유사 도메인 휘처 온톨로지를 구성한 후, 이들을 의미 유

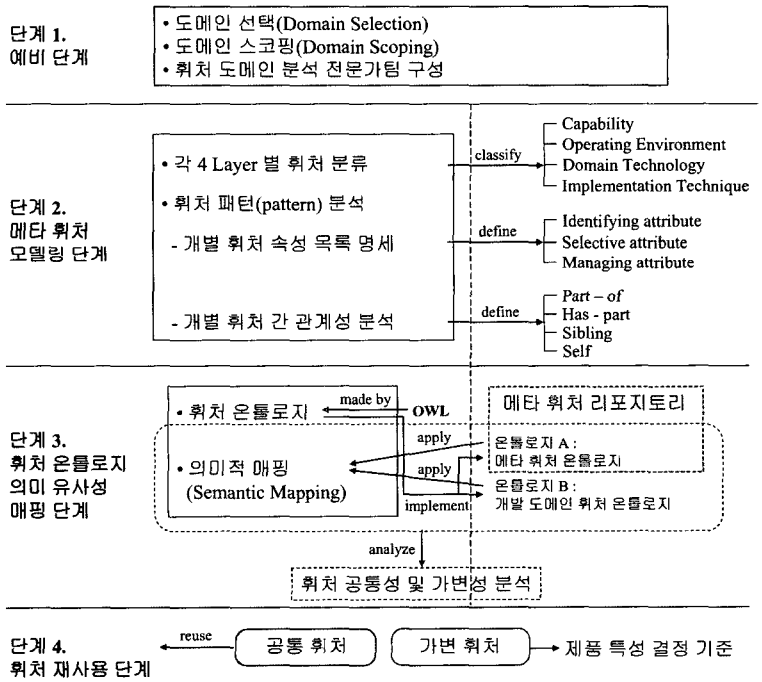

그림 2 휘처 온톨로지 기반 공통성/가변성 추출 모델

사성 매핑을 통해 휘처의 공통성과 가변성을 분석한다.

단계 4는 휘처의 재사용(feature reusing)단계로서, 분석된 휘처의 공통성과 가변성에 대해 공통적인 휘처로 분류된 것은 재사용하고, 가변적인 휘처로 분류된 것은 제품 계열의 다른 제품으로 고려할 수 있는 기준으로 판단한다.

본 기법의 접근 모델에 대한 각 단계별 세부 수행 내역을 그림 2에서 도식화하여 나타내고 있다.

그림 2의 접근 모델에 의거, 본 논문에서 제시하는 휘처의 공통성과 가변성 추출 과정은 다음과 같다.

(1) 휘처 모델에 대한 구문적 메타 모델과 그 요소의 속성을 정의
(2) 이 메타 모델에 준거하여 대상 도메인에 대한 초기 휘처 모델을 구축
(3) 구축된 모델을 온톨로지로 변환 하여 MFR에 저장
(4) 이후, 동일 제품 계열 도메인에 대한 휘처 모델 구축 및 온톨로지로 변환
(5) 온톨로지의 의미 기반 유사성 비교 분석 기준에 의거하여, 두 모델 내 휘처간의 공통성 및 가변성을 추출
(6) 공통 휘처로 식별된 휘처는 제품 개발 시 재사용하고, 가변 휘처로 식별된 휘처는 제품의 특성을 결정짓는 기준으로 해석

4. 온톨로지 기반 휘처 공통성 및 가변성 분석 기법

본 장에서는 제 3장의 그림 2에서 제시한 추출 모델에서 명시적으로 휘처를 실체화하는 메타 휘처 모델링 2단계와 온톨로지를 이용한 의미 유사성 매핑 3 단계에 대해 세부적인 명세, 기법과 알고리즘에 대해 설명한다.

4.1 메타 휘처 모델 및 속성 정의

제품의 휘처 상위 수준 카테고리는 능력(Capability), 운용 환경(Operating Environment), 도메인 기술(Domain Technology), 구현 기술(Implementation technique)에 의해 크게 분류된다[9].

- 능력 : 사용자 측면에서의 어플리케이션의 능력
- 운용 환경 : 어플리케이션이 운용되는 환경(하드웨어, 소프트웨어 플랫폼, 운영 체제, 다른 유형 장치와의 연결 장치)
- 도메인 기술 : 하나의 도메인에서 공통적으로 사용되는 기술(예, 항공 전자 도메인에서 navigation methods)
- 구현 기술 : 알고리즘과 데이타 구조에서 설계자가 결정하는 구현 기술

능력은 사용자의 관점에서 어플리케이션의 능력인 독특한 서비스(예, 전화기 도메인에서 전화 연결), 운용(예, 전화기 도메인에서 다이얼링), 비 기능적 속성(예, 성능)이며, 운용 환경(예, operating systems)은 어플리케이션이 사용되는 환경을 나타낸다. 도메인 기술(예, 항공 전자공학 도메인에서 운항 방법)은 서비스 또는 운용을 구현하기 위하여 방법을 제시하고, 구현 기술(예, 동기화 메커니즘)은 서비스, 운용, 도메인 기능을 구현하도록 사용되는 일반적 기능 또는 기술을 말한다[9].

그림 3은 휘처의 상위 수준 카테고리를 계층별 다이어그램으로 나타낸 메타 휘처 모델이다. 그림에서 범례로 나타낸 "●"는 각 상위 수준 카테고리에서 세부적으

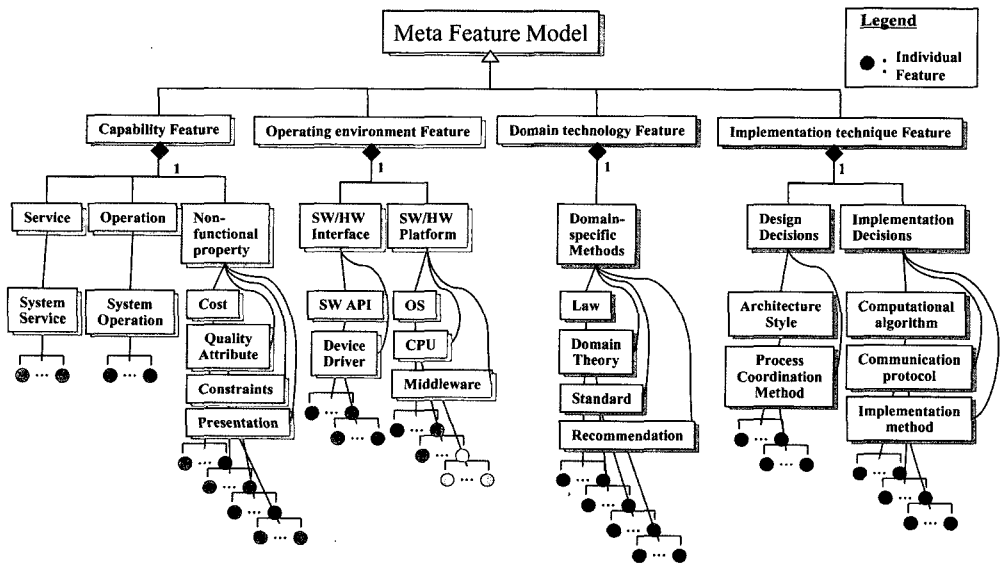

그림 3 계층별 메타 휘처 모델

로 분석될 수 있는 개별 휘처를 말한다.

그림 3에서 계층별 메타 휘처 모델의 휘처 요소에 대한 의미의 정확성을 표현하기 위해 이에 대한 속성 정의가 필요하다. 트리형의 메타 휘처 모델 상의 개념으로 간주되는 각 휘처는 개별적인 휘처로 각각의 서브 속성으로 명세화하여 실체화한다. 공통 휘처를 추출하기 위한 준거는 식별형 속성(Identifying attributes), 선택형 속성(selective attributes), 관리형 속성(managing attributes)으로 분류하였고, 그 기준으로 데이타의 명시적 표현 및 상호 교환에 대한 국제 표준인 ISO/IEC 11179의 MDR(Metadata Registries)은 메타 휘처의 속성을 정의하는 논리적 준거로서 적용한다[32]. 의미가 분명치 않는 휘처를 메타 휘처 수준으로 명시적으로 표현하여 상호 운용성을 고려한 데이타 관리와 같은 방법으로 적용할 수 있다.

메타 휘처란 '휘처의 휘처'로서 휘처를 표현, 기술 및 관리하기 위한 휘처를 말한다. 따라서 메타 휘처는 제품 개발 도메인에서 사용되는 휘처에 대한 공통된 이해를 가능하게 만들어 주는 유용한 정보이고, 휘처의 활용, 검색 및 공유를 돕기 위한 휘처를 말하며, FD에 내재된 휘처의 내용, 구조, 의미를 체계적으로 기술하는 데 사용한다. 다음 단계에서 온톨로지를 적용하여 유사 도메인에 대한 공통된 휘처를 식별해내는 기본 자원으로 활용된다. 세부적인 개별 메타 휘처의 속성은 다음과 같이 정의된다.

표 2는 속성별로 구분한 메타 휘처 속성 목록과 속성별 cardinality를 표현하고 있다. 여기서 cardinality는 각 유형별 속성의 세부 속성으로서 존재 가능한 개수를 나타낸다.

첫째, 식별형 속성은 휘처를 나타내는 형식적 이름을 나타내는 명칭(*feature name*), 기능적 분류에서 유사한 의미를 지닌 휘처들의 집합인 유사 휘처명(feature synset), 도메인 상에서 트리 구조의 수직적 관계상의 상위 휘처의 이름을 나타내는 분류(classification)로 나타낸다.

둘째, 선택형 속성은 같은 도메인의 휘처 트리 에서 sibling간의 반드시 존재해야 하는 하나의 휘처를 'mandatory'로 나타내며, 이것은 제품 계열 내에서 다른 제품들 간의 공통 휘처다. 'optional', 'alternative' 는 같은 휘처 트리 상에서 sibling간의 선택적 또는 양자택일적으로 존재하는 하나의 휘처로 이것은 다른 제품들간의 가변 휘처이다.

셋째, 관리형 속성은 MFR 에 표준 휘처로 등록하기 위하여 사용될 휘처를 구분하는 하나의 휘처에 대하여 명명한 유일한 식별자(identifier)와 MFR에 표준 휘처로 등록할 회사, 조직, 그룹의 이름인 등록 기관명으로 구성한다.

메타 휘처 모델은 휘처에 대하여 그래픽한 AND / OR의 계층적 트리 분류법으로 이루어지며 각각의 개별적 휘처는 에지 형태의 관계로 이루어져 있다.

- 상위 휘처의 관계(전체의 관계) : Part - of로 generalization 관계와 동치
- 하위 휘처의 관계(부분의 관계) : Has - part로 composed of 또는 specialization 관계와 동치
- 부모 노드가 같은 해당 휘처와 동일 계층의 관계 : Sibling 관계와 동치
- 해당 휘처가 자기 자신인 관계 : Self 관계와 동치

그림 4는 개별 휘처 속성과 개별 휘처들간의 관계를 다이어그램으로 표현 하였다. a)는 개별 휘처의 속성인 식별형, 선택형, 관리형 속성들과 해당하는 세부 속성들을 복합 및 집합의 연관 관계로 나타내었고, b)는 개별 휘처들의 4가지 관계인 *Has-part*, *Part-of*, *Sibling*, Self를 집합 관계로 표현하였다.

4.2 휘처 온톨로지를 이용한 의미 유사성 매핑

4.1절에서 명시적으로 실체화된 휘처 모델은 의미적 유사도를 측정하기 위해 휘처 온톨로지로 변환한다. 휘처 모델의 온톨로지로의 변환은 OWL을 이용하며, 휘처는 클래스(class)로, 각 세부 속성들은 ObjectProperty로

표 2 속성별로 구분한 메타 휘처 속성 목록

구 분	세부 속성	설 명	Cardinality
식별형 속성 (Identifying attribute)	명칭(name)	식별된 휘처의 명시적 이름	1
	유사 휘처명 (feature synset)	일반적 의미에서 휘처와 동일하거나 유사한 개념의 휘처 이름	1..*
	분류 (classification)	도메인 내에서 부모 휘처와 관계(상위 휘처의 이름)	1
선택형 속성 (Selective attribute)	필수 (mandatory)	도메인 내의 일련의 휘처들 중, 반드시 존재해야 하는 휘처	0 .. 1
	선택 (optional)	도메인 내에서 일련의 휘처들 중, 필수적이 아닌 선택 가능한 휘처	0 .. 1
	양자택일 (alternative)	도메인 내의 일련의 휘처들 중, 양자 택일적으로 선택이 가능한 휘처	0 .. 1
관리형 속성 (Managing attribute)	식별자 (identifier)	도메인 내에서의 휘처 고유한 ID number	1
	등록 기관 (registration authority)	휘처를 등록한 조직, 그룹 또는 회사명	1

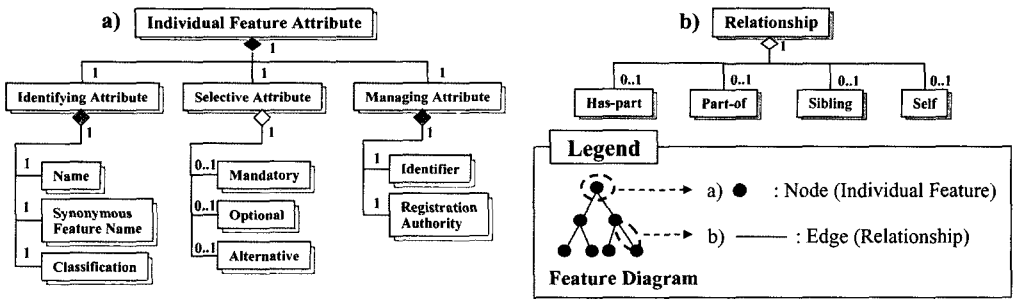

그림 4 개별 휘처 속성 다이어그램과 관계성 도시

변환한다. 본 절에서는 온톨로지로 변환된 휘처간의 공통성과 가변성 분석을 위한 의미 유사성 범위의 경계 조건 기준을 결정하고, 결정된 기준을 바탕으로 휘처의 공통성과 가변성의 범위를 정량적인 수치로 도출할 수 있는 휘처 온톨로지 유사성 매핑 알고리즘을 제시하였다.

그림 5에서 참조 모델(reference model)인 a) 메타 휘처 온톨로지와 동종 제품 계열 내의 유사 제품 도메인에서 개발하고자 하는 b) 개발 도메인 휘처 온톨로지를 의미적으로 매핑하여 각각의 유사성을 비교하는데 있어, 의미적 경계(SB: semantic boundary)를 적용하여 공통성과 가변성을 분석하는 것을 보여주고 있다. a) 메타 휘처 온톨로지는 4.1절의 메타 휘처 모델로부터 휘처를 실체화하고, OWL을 이용하여 온톨로지로 변환 구축하여 MFR에 저장하고, b) 개발 도메인 휘처 온톨로지는 동일 제품 계열의 새롭게 구축하는 휘처 모델로서, 메타 휘처 온톨로지와 동일한 방법으로 실체화 및 구축한다.

그림 5의 휘처 유사성 그래프를 이용한 휘처의 공통성과 가변성 분석을 위해서는 아래와 같은 정형적 정의가 선행되어야 한다.

정의 1. 휘처 온톨로지 유사성 그래프(FOSG: feature-Ontology similarity graph)

FOSG는 특정 휘처 도메인 온톨로지 트리 구조에서 상위 노드 U와 하위 노드 l_1, l_2, ..., l_n을 이용하여 U, I, L의 3개의 튜플(tuple)로 정의되며, 그 정형적 속성(well-formedness Property)은 다음과 같다.

1. $Onto(f)=\{U,I,L\}$ $where, f\in\{U\cup I\cup L\}, U=\{$상위노드$\}, I=\{$중간노드$\}$, $L=\{$하위노드 $l_1,l_2,...,l_n, n\geq 0\}$

일 때,

- f : feature
- U (upper level feature) : 휘처 온톨로지 상위 수준 개념 U이며, 온톨로지 Onto(f)상에서 상위 노드에 위치한 휘처이다.
- I (intermediate level feature) : 상위 수준 개념 U와 하위 수준 개념 L사이의 식별된 중간 계층 노드의 휘처이다.
- L (lower level feature) : 휘처 온톨로지 하위 수준 개념 L: $\{l_1, l_2, ..., l_n\}$이며, 온톨로지 Onto(f)상에서 하위 노드에 위치한 휘처이다.
- Onto(f)는 하나의 휘처 온톨로지에 대한 의미적 개

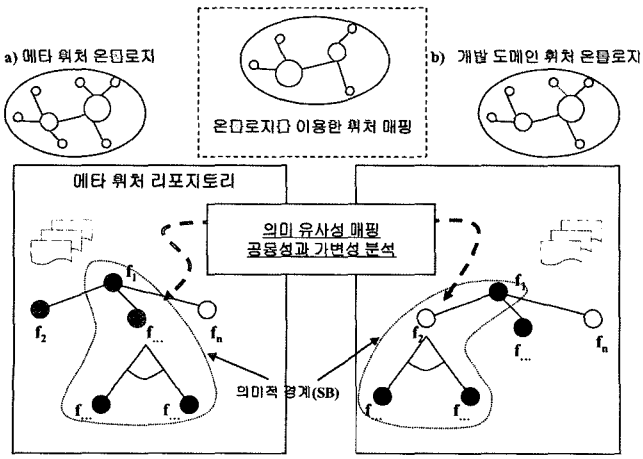

그림 5 휘처 유사성 그래프 이용 의미 사상

념의 휘처 집합이다.

- I는 온톨로지 의미 유사성 비교에 있어 중심이 되는 휘처이며, 휘처 다이어그램 상에서 I노드를 중심으로 상위 노드는 유일하게 존재하며, 하위 노드는 없을 경우부터 n개의 유한 개수가 있다.

2. $\forall f \in MFR, \forall f \Leftrightarrow \forall c$ 이고,
   - c : 온톨로지의 개념(concept)
   - MFR(meta feature repository) : 표준 휘처 속성들을 온톨로지 형태로 저장하는 저장소이며 유사 개발 도메인 개발 시 휘처 유사성을 비교하기 위한 참조 자원(reference resources)으로 활용된다.
   - 각 휘처는 MFR에 참조될 원소로 정의되고, Onto(f)의 원소가 된다.
   - Onto(f)에서의 모든 휘처는 온톨로지 내의 노드 상에서 개념(concept)과 동치이다.
3. $S = \{U, I, L \mid \bigcup (syn(L) \subset syn(I) \subset syn(U))\}$ 이다.
   - S : 의미(semantic)
   - Syn (synset) : 메타 휘처 모델링(meta feature modeling)에서 정의된 개별 속성 및 관계에서 상위 노드, 중간 노드, 하위 노드와 연결된 모든 개별적인 휘처에 대하여 동일하거나 유사한 개념에서의 휘처의 집합들을 말한다.
   - 개념 도메인 휘처 온톨로지 트리 구조상에서 유사 휘처 집합인 synset은 각 수준별로 계층적으로 하향적(top down)으로 개념을 상속받는다.

정의 2. 의미적 경계 (SB: semantic boundary)

두 휘처 모델 간의 의미적 유사성 비교를 위한 척도로 의미적 경계를 정의해야 한다. SB를 결정 하기 위해서는 $Onto(f) = \{U, I, L\}$ 은 아래 조건을 만족하여야 한다.

1. Onto(f) 상에서 F(c)는 하나의 개별 휘처이며 각각은 하나의 sub-property를 갖고 있는 개념의 집합이다.
2. F(c)는 OWL의 클래스이며, 4.1절에서 정의한 각각의 개별 휘처 속성인 식별형, 선택형, 관리형 속성을 가진다. 이 중 식별형 속성인 명칭, 유사 휘처명, 분류의 3가지와 선택형 속성(mandatory, optional, alternative) 중 1가지는 휘처의 유사성 비교를 위한 비교 요소로 사용된다.
3. Onto(f) 상에서 E는 F(c)간의 에지이며 F(c)간의 관계(relationship)로서 $E \subseteq F(c) \times F(c)$ 의 조건을 만족한다. 이 중 E는 Onto(f) 상에서 F(c)간의 Has part, Part - of, Sibling, Self의 4가지 형태의 관계로 연결된다.

SB는 그림 6과 같이 정의된다.

- F(c): 휘처의 개념적 의미의 집합
- r : 휘처 온톨로지 상에서 의미적 매핑 거리(radius)
- $d_{semantics}(\ )$: MFR 속성에 기준한 $F^n(c)$까지의 의미적 매핑의 적용 범위(거리)로서 하나의 중간 노드에 연결된 상위 노드와 하위 노드의 한 수준까지의 거리
- $SB(F(c), r) = \{F(c)\}$ such that $\forall\ nd_{semantics}(F(c), F^n(c)) \le r$

SB는 하나의 노드 F(c)에서부터 U, I, L의 계층적인 노드 $F^n(c)$까지의 의미적 거리로 계산된다[22].

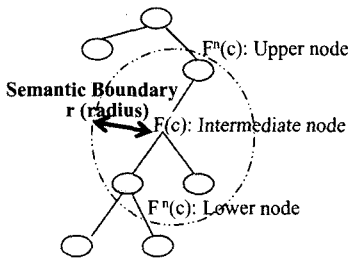

그림 6 휘처의 의미적 경계

정의 1, 2를 토대로 OWL을 이용한 두 휘처간 온톨로지 유사성 매핑 알고리즘을 그림 7과 같이 제시한다.

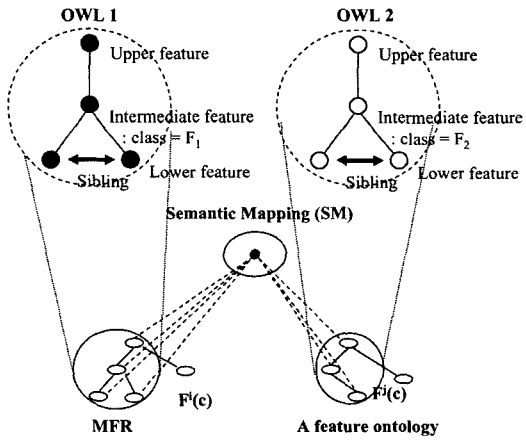

그림 7 유사성 매핑 알고리즘

초기에 사용자로부터 입력된 질의 문은 첫 번째 검색 조건으로부터 순차적으로 MFR 내 휘처 지식 베이스(knowledge base)와 단일화를 시도한다. 이때 매핑이 성공되면 다음 검색 조건으로 넘어가게 되고 매핑이 실패되었을 경우는 기존의 입력 질의문을 확장하여 의미적 질의문을 생성하고 다시 매핑을 시도한다. 매핑의 순서는 사용자의 입력 휘처 클래스에서부터 휘처 자신과 부모, 자식 노드의 차례로 매핑을 시도한다.

비교 요소로서 휘처의 4가지 속성인 OWL의 object-Property를 비교한 매핑 알고리즘을 아래와 같이 pseudo 코드로 나타내었다.

Q_f: Set of feature query condition
featureSim : Direct similarity match
featureSemanticMapping : Query inference by ontology
selfMatch : self node mapping
parentMatch : parent node mapping
childMatch: child node mapping

```
for ( f=0; f< feature Sim (Q_f+1) ; f ++) do begin
    If match( featureSim (Q_f)) := true  then continue
    else featureSemanticMapping(Q_f)  do begin
        for all Sim(Q_f) ∈ metaFeatureAttribute do begin
            If selfMatch() := true then continue
            else if parentMatch () := true then continue
            else if childMatch () := true then continue
        end
    end
end
return true
```

매핑 알고리즘에 따른 휘처의 공통성과 가변성 분석 툴 구현을 위한 공식은 다음 단계를 따른다.

단계 1: 중간 노드 간의 의미 유사성 매핑: 자신 노드와 매핑 후 동일 속성의 적중(hit) 수를 계산

단계 2: 상위 노드 간의 의미 유사성 매핑: 부모 노드와 매핑 후 동일 속성의 적중 수를 계산

단계 3: 하위 노드 간의 의미 유사성 매핑: 자식 노드와 매핑 후 동일 속성의 적중 수를 계산

단계 3의 경우는 자식 노드의 유무에 따라 분할 적용한다 (단위: %).

a) 자식 노드가 존재하지 않을 경우
1, 2의 총 8가지 속성들 중 동일 속성 적중 수를 count하여 적중률(hit ratio)을 구하여 공통성과 가변성을 계산한다.
⇒ **공통성(C: commonality) = n / 8 * 100, 가변성 (V: variability) = 100 - C**

b) 자식 노드가 1개 이상일 때
단계 1, 2의 총 8가지 속성들과 자식 노드의 각 휘처의 속성들의 전체 합계 대비 동일한 속성의 적중 수를 이용하여 적중률을 계산한다.
⇒ 자식 노드가 1개 일 때 (m = 비교된 각 휘처의 속성들 중 동일한 속성의 총 적중 수)
m/12 * 100 = C, 100 - C = V
⇒ 자식 노드가 2개 이상일 때 (n = 비교된 각 휘처 속성의 총 적중 수)
m/n * 100 = C, 100 - C = V

따라서 휘처의 공통성과 가변성을 분석은 아래와 같이 정형화하여 계산할 수 있다.

- F_1A: 메타 휘처 온톨로지의 각 휘처(=OWL의 클래스)의 개별 속성 (feature attribute of OWL 1)
- F_2A: 비교하는 제품 계열 도메인의 각 휘처의 개별 속성 (feature attribute of OWL 2)

$$C = \frac{\sum_{k=1}^{n} count(F_1A_k = F_2A_k)}{\sum_{k=1}^{n} count(FA_k)} \times 100 = \frac{m}{n} \times 100, (m \le n)$$

$$V = 100 - C$$

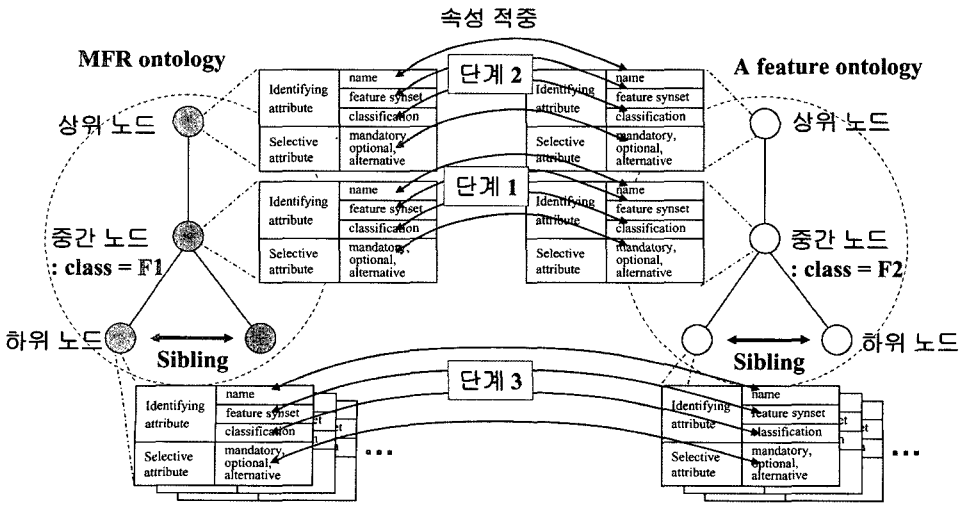

그림 8 의미 유사성 매핑 알고리즘에 따른 단계적 동일 속성 계산

위 산술식에 따른 휘처의 공통성과 가변성의 정량적인 결과는 공통 휘처와 가변 휘처를 식별하는 기준이 되어, 공통 휘처로 식별된 것은 재사용할 수 있도록 개발해야 하는 제품의 휘처 모델로 확정하고, 가변 휘처로 식별된 것은 제품의 특성을 결정짓는 기준으로 판단하여, 추후 재사용할 수 있도록 상위 카테고리로부터의 선택적인 휘처로서의 활용성을 충분히 검토하여, 필요시 MFR의 메타 휘처 온톨로 지에 수정 및 보완한다.

본 4장에서는 상기와 같이 휘처의 공통성과 가변성 분석을 위한 핵심 단계로서, 메타 휘처 모델링에 기반한 휘처의 명시적인 표현과 휘처 온톨로지 의미 유사성 매핑에 대한 알고리즘을 제시하였다.

5. 의미 유사성 매핑 툴 구현

제시한 휘처의 의미 유사성 비교 알고리즘에 대한 자동화 구축을 위해 지원 툴을 구현하였다. S/W 제품 계열 공학 도메인 분석 시 사용되는 비정형적 휘처에 대한 공통성 및 가변성 분석 기준을 온톨로지를 이용하여 적용하였고, OWL을 이용하여 온톨로지를 구성하고, Java를 이용하여 의미 유사성 매핑 툴을 구현하였으며, 개발 환경은 아래와 같다.

- 개발 플랫폼 : Window XP, CPU Pentium 1.2GHz
- 개발 언어(programming language) : JAVA, JAVA DOM
- OWL 툴 : Proteige 3.2
- 사용자 인터페이스(user interface) : 웹 기반 어플리케이션

그림 9는 휘처 모델의 부모 클래스 유사성을 비교하는 매핑 툴의 소스 코드 일부분을 보여주고 있다.

그림 10은 MFR의 메타 휘처 온톨로지와 공통성과 가변성 분석 대상인 새로 개발하고자 하는 제품의 도메인 온톨로지에 대한 것으로, 전자 결재 도메인에서 MFR의 온톨로지 내의 하나의 휘처인 'InteriorApproval'과 새롭게 개발할 제품 계열 도메인 내에서의 휘처인 'InternalApproval' 에 대하여 의미적으로 공통성과 가변성을 분석하기 위하여 온톨로지 언어인 OWL로 작성하여 구현하였다.

그림 11은 온톨로지 의미 유사성 매핑 툴을 이용하여 실제 전자 결제 시스템 도메인에서 'InteriorApproval'과 'InternalApproval'의 공통성과 가변성을 분석한 결과를 제시한 사용자 인터페이스를 보여준다. 화면 1은 의미 유사성 매핑을 위해 MFR에 저장된 메타 휘처 온톨로지와 새롭게 개발하는 휘처 온톨로지 파일을 불러오는 화면이고, 화면 2는 비교할 휘처의 이름, 즉 클래스 명을 각각 입력하는 화면이며, 화면 3은 매핑의 결과를 보여주는 화면이다. 예로서 두 개 휘처의 유사성 비교를 적용해본 결과 공통성이 91.67%, 가변성이 8.33%로 공통 휘처로 식별되어 'InternalApproval'이라는 휘처는 재사용 가능한 휘처로 간주한다.

6. 실험 및 평가

본 논문에서 제시하는 온톨로지 기반 휘처의 공통성/가변성 추출 기법의 효과성과 신뢰성을 입증하기 위해, 실험 및 평가를 제품 계열 상의 휘처 분석 도메인을 전

```
public String similarParentClass(ObjectGraph objectGraph){
        Hashtable ht;
        String k=null;
        String parentClassNames = objectGraph.getParentClassName();
        StringTokenizer stParentClass = new StringTokenizer(parentClassNames, "|", false);
        while(stParentClass.hasMoreTokens()){
                String parentClassName = (String)stParentClass.nextToken().trim();
                ht = (Hashtable)objectGraph.getHypernymSenseTable().clone();
                        for ( int i =0; i< ht.size() ; i++ ){
                        String senses =     (String)ht.get(String.valueOf(i+1));
                        StringTokenizer st = new StringTokenizer(senses, ",=>", false);
                        while(st.hasMoreTokens()){
                                String sense = (String)st.nextToken().trim();
                                if (parentClassName.equals(sense)){
                                        if(k==null)
                                                k = String.valueOf(i+1);
                                        else
                                                k = k +"|"+String.valueOf(i+1);
                                }
                        }
                }
        }
        return k;
}
```

그림 9 의미 유사성 매핑 툴 소스 코드

MFR 휘처 온톨로지

```
<owl:Class rdf:ID="DocumentForm">
  <rdfs:subClassOf rdf:resource="#Drafting"/>
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:ID="InteriorApproval"/>
    <owl:Class rdf:ID="ExteriorSending"/>
  </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="InteriorApproval">
  <rdfs:subClassOf rdf:resource="#DocumentForm"/>
    <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:ID="Informality"/>
      <owl:Class rdf:ID="WorkAttendance"/>
    </owl:unionOf>
  <owl:Class rdf:ID="InteriorApprovalDescriptor"/>
  <owl:Class rdf:ID="InteriorApprovalName">
    <rdfs:subClassOf rdf:resource
      <owl:Restriction>
        <owl:onProperty rdf:reso
        <owl:hasValue rdf:resour
      </owl:Restriction>
  </owl:Class>
    <owl:ObjectProperty rdf:ID="
      <rdfs:domain rdf:resource=
      <rdfs:range rdf:resource="#
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="
      <rdfs:subPropertyOf rdf:res
      <rdfs:range rdf:resource="#
    </owl:ObjectProperty>
```

개발 도메인 휘처 온톨로지

```
<owl:Class rdf:ID=" NewDocument ">
  <rdfs:subClassOf rdf:resource="#Request"/>
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:ID=" InternalApproval "/>
    <owl:Class rdf:ID=" ExternalDispatch "/>
  </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="InternalApproval">
  <rdfs:subClassOf rdf:resource="#NewDocument"/>
    <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:ID="Informality"/>
      <owl:Class rdf:ID="WorkAttendance"/>
    </owl:unionOf>
  <owl:Class rdf:ID="InternalApprovalDescriptor"/>
  <owl:Class rdf:ID="InternalApprovalName">
    <rdfs:subClassOf rdf:resource="#InternalApprovalDescriptor"/>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#Names"/>
        <owl:hasValue rdf:resource="#InternalApproval" />
      </owl:Restriction>
  </owl:Class>
    <owl:ObjectProperty rdf:ID="hasInternalApprovalDescriptor">
      <rdfs:domain rdf:resource="#InternalApproval"/>
      <rdfs:range rdf:resource="#InternalApprovalDescriptor"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="hasName">
      <rdfs:subPropertyOf rdf:resource="#hasInternalApprovalDescriptor">
      <rdfs:range rdf:resource="#InternalApprovalName">
    </owl:ObjectProperty>
```

그림 10 휘처 온톨로지의 OWL 소스 코드

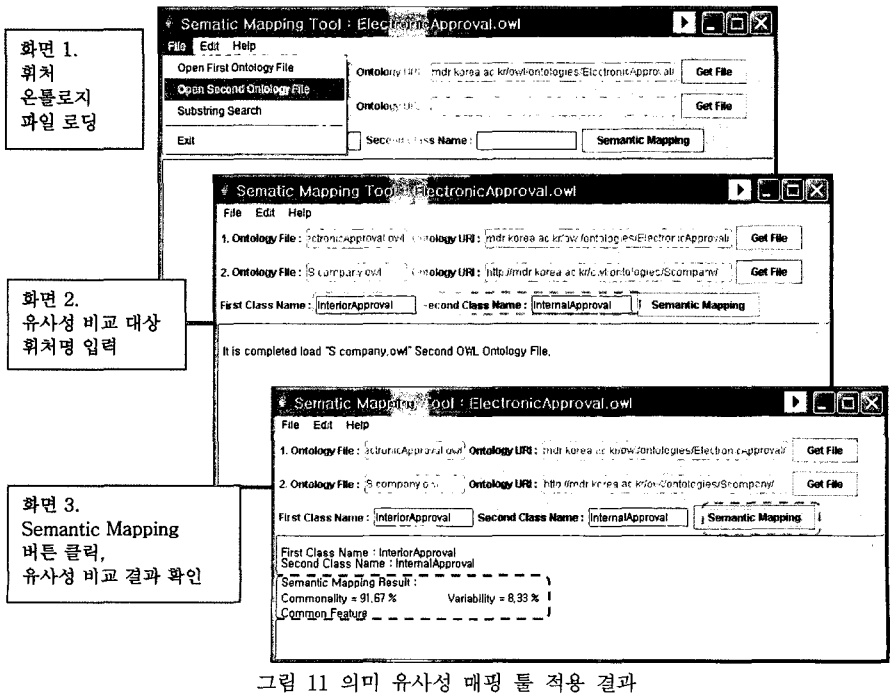

그림 11 의미 유사성 매핑 툴 적용 결과

자 결재 시스템을 대상으로 적용하였다. 적용 범위는 MFR에 저장되어 있는 S조직의 전자 결재 시스템 도메인 메타 휘처 온톨로지와 L조직의 전자 결재 시스템 휘처 온톨로지를 비교하여 의미 유사성 매핑 툴을 통해 공통성 및 가변성을 분석하였다.

S조직의 전자 결재 시스템을 메타 휘처 온톨로지로 선행 구성하는 단계로, 메타 휘처 모델링을 위해 4 Layer별 휘처를 분류하고, 휘처 패턴 분석을 실시한다. 본 실험에서는 제약 사항으로 메타 휘처 모델의 4개 계층 중 능력(capability) 계층에서 서비스 측면만을 고려하였다. 개별 휘처간 관계성 분석 및 개별 휘처 속성 목록 명세는 그림 12에서와 같이 System Service 측면에서의 휘처들을 산출하고, 그 휘처들간의 관계성을 분석하여 트리 Taxonomy 형태로 휘처 패턴을 분석한다. 그 후 개별 휘처들의 속성 목록을 명세한다. 그림 12에서는 개별 휘처인 'InteriorApproval'의 속성 목록 명세를 보여주고 있다.

다음 단계로 개별 휘처들의 속성 목록을 바탕으로 식별형 속성 3가지와 선택형 속성 1가지를 이용하여 그림 10과 같이 휘처 온톨로지를 OWL로 구성한다. 구성된 메타 휘처 온톨로지는 MFR에 저장하고, 새롭게 개발해야 할 L조직의 전자 결재시스템 휘처 온톨로지를 동일한 방법으로 구성한다. 다음으로 제안한 의미 유사성 매핑 툴을 이용하여 메타 휘처 온톨로지와 신규 휘처 온톨로지간 휘처의 의미 유사성 매핑을 실시한다.

제안한 의미 유사성 매핑 툴의 효과성을 평가하기 위하여, 기존의 분석 방법인 경험과 직관에 의한 공통성 및 가변성 분석과 본 논문에서 제안한 모델로 구현한 의미 유사성 매핑 툴에 의한 결과의 비교는 표 3과 같다.

표 3은 전자 결재 시스템 도메인에서 S 조직의 전자 결재 시스템의 분석된 휘처 온톨로지를 메타 휘처 온톨로지로 가정하고, L 조직의 분석된 휘처 온톨로지를 개발 예정인 제품의 온톨로지로 가정하여, 공통 휘처는 기존의 방법인 도메인 전문가의 직관 및 경험에 의한 분석을 통해 공통된 휘처로 식별된 Cf1~Cf20을 나타낸다. 가변 휘처는 동일한 분석 방법으로 두 개의 휘처 온톨로지를 비교하되, 동일한 부모 노드 아래에서 제품을 결정짓는 가변적인 휘처로 분석된 Vf1~Vf10 이다.

이에 대해 의미 유사성 매핑 툴에 의한 식별 결과는 그림 13, 14와 같은 그래프로 나타났다.

그림 13에서 도메인 전문가의 경험과 직관적인 방법으로 분석한 공통 휘처 20개를 의미 유사성 매핑 툴로 실험해본 결과 20개중 2개는 58.33%를 나타냈으며, 18개는 최소 75%의 적중률을 확인할 수 있었다. 그리고 그림 14에서 가변 휘처 10개의 실험 결과, 최소 8.33%, 최대 41.67%의 적중률을 확인할 수 있었다.

결론적으로 본 논문에서 제안한 방법에서 전체의 75% 이상의 결과를 가진 휘처는 차기 개발 시 재사용이 가능한 공통 휘처로 식별되고, 58%~74% 사이의 휘처는 임계 범위에 해당되어 공통 휘처의 가능성을 가지고 있어 재사용의 고려대상이 되어야 할 것으로 간주할 수 있다. 그 이외의 나머지 부분인 가변성은 공통 휘처가 아닌 가변적인 범위로 동종의 제품 계열에서 제품 고유의 특징적 요소로 활용되어야 한다.

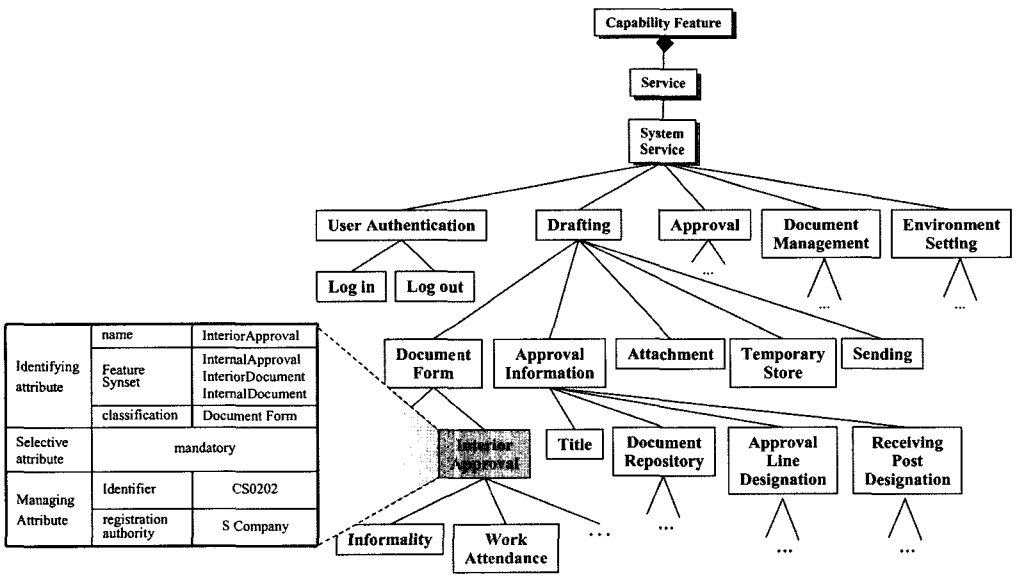

그림 12 휘처 패턴 분석

표 3 기존 분석 방법에 의한 공통 및 가변 휘처 분석 결과

구 분		S조직 휘처 온톨로지	L 조직 휘처 온톨로지
공통 휘처	Cf1	Authentication	User Authentication
	Cf2	Draft	Drafting
	Cf3	New Document	Document Form
	Cf4	Interior Approval	Internal Approval
	Cf5	Work Attendance	Work Attendance
	Cf6	Exterior Approval	External Approval
	...	...	...
	Cf19	Approval	Approval
	Cf20	Environment Setting	Environment Setting
가변 휘처	Vf1	Informality	Approval Line Designation
	Vf2	College Register	Request business trip
	...	...	...
	Vf9	Document Reservation	Document Return
	Vf10	Approval Form	Register Management

※ Cf: Common feature, Vf: Variable feature

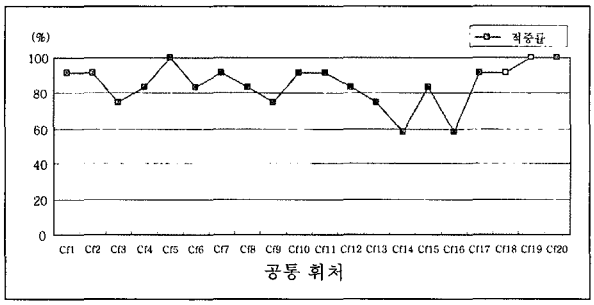

그림 13 공통 휘처 실험 결과 값

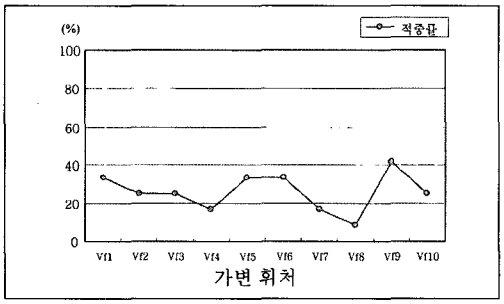

그림 14 가변 휘처 실험 결과 값

상기와 같이 동일 제품 계열의 도메인에서 휘처의 공통성과 가변성 분석을 위해 의미 유사성 매핑 툴을 이용한다면, 기존의 도메인 전문가에 의한 직관적인 분석 방법 보다, 정확성에 기반한 명시적이고 신속한 휘처 분석이 이루어 질 것이다. 또한 온톨로지의 의미 기반에 의거 휘처간 유사성 분석을 통해 휘처의 공통성과 가변성을 정확히 구분하여 휘처 모델의 재사용성을 극대화 할 수 있다. 특히, 도메인의 규모가 클수록 더욱 유용하고, 효과적일 것이다.

7. 결론 및 향후 연구

본 논문에서는 소프트웨어 제품 계열 도메인 개발 시 적용되는 휘처 분석 방법이 도메인 전문가의 직관적인 분석으로 비정형적임에 따라 실제 적용하기 불명확하고 휘처에 대한 이해의 오류로 인한 명시적이지 못한 휘처 추출의 문제점을 해결하고자 메타 휘처 모델에 기반한 개별 휘처 속성을 정의하고 온톨로지의 의미 기반 유사성 기법을 적용한 FOSG를 통해 휘처의 공통성과 가변성을 추출하기 위한 모델을 제시하였다.

기대 효과로서, 휘처 모델에 대한 메타 모델 정의로 휘처의 구문적 정확화 및 실체화를 통해, 개발 시 모델의 이해성 증진과 개발자간 동일한 해석을 제공한다. 또한 온톨로지를 이용한 휘처의 의미적 매핑으로, 공통 휘처의 정확하고 신속한 추출을 통한 휘처 모델의 구축으로 재사용성을 극대화함으로써 대규모 시스템 개발을 용이하게 할 수 있다[33,34]. 비록 현행 온톨로지적 접근 방법은 복잡하고 많은 시간과 노력이 소요된다는 적지 않은 문제점을 내포하고 있지만, 엔터프라이즈 급 이상의 국방 정보 시스템과 같은 고비용, 대규모의 정보 시스템을 개발할 경우, 트레이드 오프(trade off)를 고려한다면 본 논문에서 제시한 방법은 상당한 설득력을 가지고 있다고 판단되며, 추후 휘처의 재사용으로 인해서 개발 비용, 시간을 정량적으로 절약할 뿐 아니라, 이해 당사자의 요구사항을 명시적으로 반영할 수 있는 긍정적인 효과를 기대할 수 있을 것이다.

향후 연구 과제로 단일 휘처 온톨로지 내에서 휘처들 간의 중복 및 충돌 등의 상호작용 문제를 해결하기 위해 좀더 명시적인 메타 휘처 모델을 정제화하고, 이에 따른 의미 유사성 매핑 툴의 확장에 관한 연구와 아울러, FORM 기반의 개발 방법론에 본 기법의 적용을 통

한 실용화에 대한 연구를 진행하고자 한다.

참 고 문 헌

[1] Clements, P. and Northrop, L., Software Product Lines: Practices and Patterns, Addison - Wesley, Upper Saddle River, NJ, 2002.

[2] J. Neighbors, "The Draco Approach to Construction Software from Reusable Components," *IEEE Transactions on Software Engineering*, SE-10(5), pp.564-573, September 1984.

[3] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson, Feature-Oriented Domain Analysis(FODA) Feasibility Study, *Technical Report CMU/SEI-90-TR-21*, Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, November 1990.

[4] Bosch, J., Design & Use of Software Architectures, Addison-Wesley and ACM Press, 2000.

[5] Bosch, J., Florijn, G., Greefhorst, D., Kuusela, J., Obbink, J. H., Pohl, K., "Variability Issues in Software Product Lines," In: van der Linden, F. (eds.): Software Product Family Engineering. Lecture Notes in Computer Science, Vol. 2290, pp.13-21, Springer-Verlag, Berlin Heidelberg, 2002.

[6] Coplien, J., Hoffman, Daniel, Weiss, D., "Commonality and Variability in Software Engineering," IEEE Software, 15(6), pp.37-45, 1998.

[7] Czarnecki, K, Eisenecker, U., Generative Programming: Methods, Tools, and Applications, Reading, MA, Addison Wesley Longman, Inc., 2000.

[8] Kang, K., Lee, J., Donohoe, P., Feature-Oriented Product Line Engineering. IEEE Software, 19(4), pp.58-65, 2002.

[9] Lee, K., Kang, K., Lee, J., "Concepts and Guidelines of Feature Modeling for Product Line Software Engineering," In: Gacek, C. (eds.): Software Reuse: Methods, Techniques, and Tools. Lecture Notes in Computer Science, Vol. 2319, pp.62-77, Springer-Verlag, Berlin Heidelberg, 2002.

[10] R. Priteto-Diaz, Implementing Faceted Classification for Software Reuse, Communication of the ACM, 34(5), pp.88-97, May 1991.

[11] 송재승 외 2명, "프로덕트 라인 개발에서 피처 모델의 명세화 기법", 정보과학회 논문지 : 소프트웨어 및 응용, 제30권, 제11호, 2003.

[12] 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.

[13] Don Batory, "Feature models, Grammars, and Propositional Formulas," SPLC 3714, pp.7-20, 2005.

[14] Ferber, S., Haag, J., Savolainen, J., "Feature for Reengineering a Legacy Product Line," G. Gchastek (ed), Software Product lines, Springer verlag, Berlin, pp.235-256, 2002.

[15] Kang, K. C., Kim, S., Lee, J., et al., "Feature-Oriented Engineering of PBX software for Adaptability and Reusability," Software Practice & Experience, Vol. 29, No. 10, pp.875-896, 1999.

[16] Thomas von der Maben and Hort Lichter, "Determining the Variation Degree of Feature Models," SPLC2005, LNCS 3714, pp.82-88, 2005.

[17] Griss, M. L., Favaro, J., d'Alessando, M., "Integrating Feature Modeling with the RSEB," Proceedings of 5th International Conference on Software Reuse, Victoria Canada, IEEE, pp.76-85,1998.

[18] Van Gurp, J., Bosch, J., Svahnberg, M., "On the notion of variability in software product lines," Proceedings on Working IEEE/IFIP Conference on Software Architecture, pp.45-54, 2001.

[19] Hai Wang, Yuan Fang Li, et al., "A Semantic Web Approach to Feature Modeling and Verification," Semantic Web Enabled Software Engineering (SWESE'05), Galway, Ireland, 2005.

[20] Xin Peng, Wenyun Zhao, Yunjiao Xue, Yijian Wu, "Ontology-Based Feature Modeling and Application-Oriented Tailoring," 9th International Conference on Software Reuse, ICSR 2006, Turin, Italy, LNCS 4039, pp.87-100, 2006,

[21] Tim Berners-Lee, James Hendler, Ora Lassila, "The Semantic Web," Scientific American, 2001.

[22] Ju-hum Kwon, Doo-Kwon Baik, et al., "Measuring Semantic Similarity Based on Weighting attributes of Edge Counting," LNCS 3397, 2005.

[23] Ju-hum Kwon, Doo-Kwon Baik, et al, "Bridging Real World Semantics to Model World Semantics for Taxonomy Based Knowledge Representation System," Journal of Computer Science and Technology, pp.296-308, 2005.

[24] Ju-hum Kwon, Doo-Kwon Baik, et.al, "Intelligent Semantic Concept Mapping For Semantic Query Rewriting/Optimization In Ontology-based information System," International Journal of Software Engineering and Knowledge Engineering, Vol.14, 2004.

[25] Yuqin Lee and Wenyun Zhao, "Domain Requirements Elicitation and Analysis - An Ontology-Based Approach," ICCS2006, Part IV, LNCS 3994, pp.805-813, 2006.

[26] Kannan Mohan and Balasubramaniam Ramesh, "Ontology-based Support for Variability Management in Product and Service Families," proceedings of the 36th Hawaii International Conference on System Sciences(HICSS'03), 2003.

[27] Michael K. Smith, Chris Welty, Deborah L. McGuinness, "OWL Web Ontology Guide," W3C Recommendation 10, Feb. 2004.

[28] M. Andrea Rodriguez, et al, "Determining Semantic Similarity among Entity Classes from Different Ontologies," *IEEE Transaction on Knowledge and Data Engineering*, Vol. 15, No. 2, Mar. 2003.

[29] Yuhua Li, Zuhair A Bandar, David McLean, "An approach for measuring semantic similarity between words using multiple information sources," IEEE Transaction, Data and Knowledge Engineering, 15(4), pp.871-882, 2003.
[30] 하상범 외 1명, "온톨로지 기반 추론을 이용한 시맨틱 검색 시스템", 정보과학회 논문지 : 소프트웨어 및 응용, 제32권, 제3호, 2005.
[31] Mikyeong Moon, Keunhyuk Yeom, "An Approach to Developing Domain Requirements as a Core Asset Based on Commonality and Variability Analysis in a Product Line," *IEEE Transactions on Software Engineering*, Vol.31, No.7, pp.551-569, July 2005.
[32] ISO/IEC JTC1 SC32, ISO/IEC 11179: Specification and Standardization of data elements, Part 1-6.
[33] 이순복, 이태웅, 김진우, 백두권, "온톨로지를 이용한 S/W Product Line 도메인의 명시적 feature 분석 모델", 한국정보처리학회 춘계학술발표 논문집, 제13권, 제1호, pp.269-272, 5. 2006.
[34] 김진우, 이순복, 이태웅, 백두권, "Software 제품계열 공학에서 온톨로지에 기반한 feature의 공통성 및 가변성 분석 모델", 한국 컴퓨터 종합 학술대회 논문집 (c), pp.139-141, 한국정보과학회, 6. 2006.

이 순 복
2000년 해군사관학교 졸업. 2003년~2005년 해군 정보통신학교 전산학부 교관. 2006년~현재 고려대학교 컴퓨터학과 석사과정. 관심분야는 제품계열공학, 온톨로지, UML 모델링

김 진 우
1994년 해군사관학교 졸업. 2002년 국방대학교 전산학 석사. 2004년~현재 고려대학교 컴퓨터학과 박사과정. 관심분야는 EA, 소프트웨어 공학, 온톨로지 등

송 치 양
1985년 한남대학교 전산학과 학사. 1987년 중앙대학교 전산학과 석사. 2003년 고려대학교 컴퓨터학과 박사. 1990년~2005년 한국통신 중앙연구소 책임연구원 2005년~현재 상주대학교 소프트웨어 공학과 전임강사. 관심분야는 UML 모델링 기술, 소프트웨어 개발방법, 휘처 지향 도메인 분석 등

김 영 갑
2001년 고려대학교 식량자원학과 학사 (컴퓨터학과 부전공). 2003년 고려대학교 컴퓨터학과 석사. 2006년 고려대학교 컴퓨터학과 박사. 2006년~현재 고려대학교 정보보호 대학원 연구교수. 관심분야는 MDR, 컴포넌트, 제품계열 아키텍처, 위험분석, 데이타보안 등

권 주 홈
1996년 금오공대 전산학과 학사. 1999년 Wayne State Univ. 전산학 석사. 2005년 고려대학교 컴퓨터학과 박사. 2006년~현재 공군 중앙전산소 전산운영 과장. 관심분야는 EA, 온톨로지 통합, 컴퓨터 보안 등

이 태 웅
2002년 고려대학교 컴퓨터학과 학사. 2004년 고려대학교 컴퓨터학과 석사. 2005년~현재 한국과학기술정보연구원. 관심분야는 온톨로지, 메타데이타, 시맨틱 웹, 서치 엔진

김 현 석
2000년 육군사관학교 졸업. 2006년 고려대학교 컴퓨터학과 석사. 2006년~현재 고려대학교 컴퓨터학과 박사과정. 관심분야는 정형기법, 보안 프로토콜, RFID(무선주파수식별), 무선 네트워크 보안 등

백 두 권
1974년 고려대학교 수학과 학사. 1977년 고려대학교 산업공학과 석사. 1983년 Wayne State Univ. 전산학 석사. 1985년 Wayne State Univ. 전산학 박사 1986년~현재 고려대학교 컴퓨터학과 교수. 1989년~현재 한국정보과학회 이사/평의원/부회장. 1991년~현재 한국시뮬레이션학회 이사/부회장/감사. 1991년~현재 ISO/IEC JTC1/SC32 국내위원회 위원장. 2001년~현재 행자부 등록 (사)도산아카데미 원장 2002년~2004년 고려대학교 정보통신대학 초대학장. 2002년~2004년 한국시뮬레이션학회 회장. 2004년~현재 정통부 등록 (사)한국정보처리학회 부회장. 2005년~현재 정통부 등록 (사)한국정보과학회 부회장. 2005년~현재 교육부 등록 (사)홍사단 공의원. 2005년~현재 산자부 등록 (사)한국시뮬레이션학회 고문. 관심분야는 데이타베이스 공학, 시뮬레이션, 소프트웨어 공학, 메타데이타, 정보통합, 정보보호 등