한국 CAD/CAM 학회 논문집
제 11권 제 2호 2006년 4월 pp. 88-96

기술논문

제품개발을 위한 온톨로지 기반 지식 프레임워크

이재현*, 서효원**

Ontology-based Knowledge Framework for Product Development

Lee, J.H.* and Suh, H.W.**

ABSTRACT

This paper introduces an approach to ontology-based framework for knowledge management in a product development domain. The participants in a product life cycle want to share the product knowledge without any heterogeneity. However, previous knowledge management systems do not have any conceptual specifications of their knowledge. We suggest the three levels of knowledge framework. First level is an axiom, which specifies the semantics of concepts and relations. Second level is a product development knowledge map. It defines the common domain knowledge which domain experts agree with. Third level is a specialized knowledge for domain, which includes three knowledge types; expert knowledge, engineering function and data-analysis-based knowledge. We propose an ontology-based knowledge framework based on the three levels of knowledge. The framework has a uniform representation; first order logic to increase integrity of the framework. We implement the framework using prolog and test example queries to show the effectiveness of the framework.

Key words : Ontology, First order logic, Knowledge framework

1. 서 론

1990년대 초반 시장 진입 기간을 줄이기 위해 동시공학에 대한 많은 연구가 이루어졌다[1]. 제품 개발 기간을 단축하려면 제품 지식에 대한 체계적인 관리가 필요하다. 엔지니어들은 필요 지식을 찾기 위해 근무 시간의 70% 이상을 소비하고 그렇게 불필요하게 사용되는 시간은 엔지니어의 생산성을 떨어뜨린다. Stauffer *et al.*[2]는 엔지니어들이 과거 프로젝트의 지식을 활용하기 위해 많은 시간을 소비하는 이유에 대하여 연구하였다. 문제는 과거 지식이 잘 정리되어 있지 않기 때문이었다. 그 첫 번째 이유는 엔지니어들이 자신이 가진 정보와 지식을 정리할 시간이 충분하지 않다는 것이고 두 번째는 회사가 지식을 자신들의 재산으로 생각하지 않을 뿐 아니라 지식 관리를 위해 충분한 예산을 확보하지 못한다는 것이다. Court[3]의 실험적 연구 결과에 따르면 엔지니어는 제품을 개발하는 동안 평균적으로 자신의 개인적 지식의 약 30%를 사용한다. 특정 경우에는 50~70%까지 사용하기도 한다. 따라서 지식 관리를 통해 엔지니어의 개인적 지식의 활용도를 높이고 지식을 공유하는 것이 제품 개발의 중요한 수단이 된다. 그러므로 지식베이스 시스템(Knowledge Base System: KBS)을 기반으로 지식을 체계적으로 저장하고 활용할 수 있는 프레임워크가 필요하다.

1980년대 초 연구들은 KBS의 개발을 사람의 지식을 지식베이스로 옮기는 프로세스로 보았다. 이는 KBS에 필요한 지식은 이미 존재하고 이를 수집하여 구현하면 된다는 가정을 기반으로 한다. 하지만, 전문가의 문제해결 능력 같은 전략 지식이 도메인 지식과 혼합되어 있기 때문에 규모가 큰 지식베이스 시스템에서는 이 가정이 적절하지 않다. 최근 지식관리 프로세스는 한 번에 지식베이스를 구축하는 것이 아니라, 지식베이스를 구축하는 프로세스 자체를 지식을 모델링하는 활동으로 보고 계속적으로 전문가 지식을 모델링하며 개선해 나가는 방법을 취한다[4].

*교신저자, 학생회원, 한국과학기술원 산업공학과
**종신회원, 한국과학기술원 산업공학과
- 논문투고일: 2004. 12. 01
- 심사완료일: 2005. 10. 25

이러한 점진적인 개선 접근방법을 제품개발영역에 적용하기 위해서는 제품 개발 분야의 도메인 지식이 복잡하기 때문에, 이를 모델링하기 위한 프레임워크를 명확히 정의하여 지식을 모델링할 때 가이드를 해주어야 한다. 이전에도 몇몇 지식 관리를 위한 접근법들이 제안되었지만 모호함과 이질감없이 여러 유형의 포괄적 지식을 수용하는 데 한계가 있었다. 본 논문에서 제안하는 접근법은 새로운 기술인 온톨로지에 기반을 두고 있다. 자세한 내용은 다음 장에서 논의될 것이다.

2장에서는 이전의 관련 연구들을 소개하고 3장에서는 온톨로지 기반의 지식 프레임워크(Ontology-Based Knowledge Framework: OBKF)를 소개한다. 지식 프레임워크의 각 요소에 대한 설명이 4장에서 논의되고 5장에서는 OBKF 접근법에 대한 예가 소개된다. 마지막으로 6장에서는 OBKF의 예상 효과와 향후 연구를 논의한다.

2. 기존 연구

본 장에서는 지식관리 시스템의 지식 프레임워크에 대한 기존 연구들을 살펴보면서 본 연구의 접근방법과 비교하겠다.

초기 지식 프레임워크의 연구는 데이터 모델의 통합에 대한 문제에서 발생하였다. 데이터베이스 통합의 문제는 데이터 모델의 의미 이해 문제를 해결해야 했기에, 기존 데이터베이스 연구들[5]에서는 이를 해결하기 위하여 온톨로지를 활용하였다. 온톨로지는 Gruber[6]의 정의에 따르면 개념화에 대한 명확한 명세화로서, 도메인의 개념과 관계들을 컴퓨터가 이해 가능하도록 논리적, 정형적으로 기술한 것이다. Bozsak *et al.*[7]는 온톨로지가 5가지 구성요소로(개념, 관계, 개념 구조, 함수관계, 공리)로 이루어진다고 하였다.

제품 개발 분야에서 온톨로지는 도메인 전문가들 간에 협의된 지식으로 여러 분야가 협업을 할 때 발생하는 정보 의미에 대한 이해를 돕고[27] 정보의 공유를 가능하게 한다[28,29]. 이러한 기능 측면에서 온톨로지의 구성요소 중 개념, 관계, 개념 구조, 함수관계는 협의된 지식의 구조를 정의하는 반면, 공리는 이 요소들의 의미를 구체화 하는 역할을 맡는다. Yoshioka[8]는 공리를 제외한 나머지 요소들로 구성된 협의된 지식을 'Meta-model'이라 정의하고, 이를 제품설계 분야에서 서로 다른 시스템들간 정보 교환 시 일관성 유지에 활용하였다. 이 연구는 개념들의 의미를 개념 사전 형식으로 정리함으로써 공리정의의 복잡함을 피하고, 협업 체계의 일관성 유지를 꾀하였으나, 시스템 활용 과정에서 온톨로지의 수정/보완 시 발생하는 문제들에 대해서는 해결방법이 없다는 단점이 있다. 이와는 달리 온전한 온톨로지를 적용한 PACT[9], SHADE[10]와 같은 연구에서 온톨로지를 활용한 에이전트 협업 시스템을 개발하여 서로 다른 시스템간의 정보 교환사 온톨로지의 활용성을 보여주었다.

이후 온톨로지의 제품개발 분야 적용의 필요성이 커짐에 따라 많은 연구들이 있었는데, 그 중 대표적인 것으로 Lin *et al.*[11]은 고객의 요구사항들을 관리하기 위한 제품 온톨로지를 개발하였고, Kitamura *et al.*[12]은 제품의 기능을 표현하기 위한 기능 온톨로지를 개발하였다. Borst *et al.*[13]는 일반적인 시스템을 표현하는데 활용되는 PHYSYS 온톨로지를 기반으로 엔지니어링 온톨로지를 개발하였다. 또한, Gruber와 Olsen[14]은 엔지니어링 환경에서 공학적 계산을 위한 수학 온톨로지를 개발하였다. 이 기존연구들은 제품개발 분야에서 온톨로지의 역할을 정의하였고, 그 활용성을 보여주었다.

제품개발 분야에 지식베이스 시스템(Knowledge-Based System: KBS)을 적용하기 위해서는 온톨로지 기반의 지식베이스를 구축해야 할 뿐만 아니라, 온톨로지 기반의 문제해결 지식들을 수집하고 관리해야 한다. 온톨로지가 전문가들 간에 협의된 지식이라면, 실제 업무에서 발생하는 의사결정을 지원하는 지식은 문제를 해결하는 지식이다. 이러한 문제해결 지식들과 온톨로지는 KBS에서 통합된 모습으로 관리되어야 한다. CommonKADS[15], MIKE[16], Protégé 2000[17]과 같은 연구들은 이를 위하여 고유의 지식 모델링 프레임워크를 제안하였다. 하지만, 이 연구들은 문제해결 지식을 온톨로지 기반으로 정의하기 위한 명확한 관계와 표현방법에 대하여 논의하지 않았다.

한편, 온톨로지의 구성요소 중 공리는 지식을 명세화하는 역할을 하므로 매우 중요하지만 모델링이 어렵고, 많은 부분 온톨로지 구축자의 수작업(hand-coding)이 필요하다는 단점이 있다. 하지만, 그 필요성을 무시할 수 없기 때문에 공리를 독립적으로 모델링하기 위한 연구들도 있었다. Ontolingua[18] 연구에서는 온톨로지 정의시 어떤 부분이 공리인지 명시할 수 있도록 하여 공리를 부분적으로 분리하여 정의할 수 있도록 하였다. Staab와 Maedche[19]는 공리들을 분류하여 개념화하고 온톨로지에서 정의하는 공리들을 하나의 객체로 정의하는 접근방법을 제안하였다.

따라서, OBKF는 제품개발 지식을 모델링 하기 위해 문제해결 지식, 도메인의 일반적 지식과 지식의 명

세화로 구분하여 모델링하는 지식 프레임워크를 제안한다. 이와 같은 지식의 구분은 각 지식 레벨의 역할을 명확히 하고, 상호간의 관계를 이해하는데 도움을 준다. OBKF의 구조는 3장에서 간단히 서술되고, 4장에서 상세히 서술될 것이다.

3. 온톨로지 기반 지식 프레임워크

본 논문은 온톨로지 기반의 지식 프레임워크(OBKF)를 제안한다. OBKF 접근법은 명확한 지식관리를 위해 온톨로지 기반의 지식구조를 갖는다. 또한 포괄적인 지식관리를 위하여 함수, 규칙, 데이터분석기반 지식과 같은 전형적인 유형의 지식을 OBKF 안에 포함한다. 일관된 표현을 위해서 OBKF의 지식들은 일차 논리(First-Order-Logic: FOL)를 사용하여 표현된다.

지식 구조: 이전 연구들[20,21]이 다양한 유형의 제품 개발 지식의 분류체계를 제안하였지만 분류의 기준과 분류된 카테고리 간의 관계성이 명확하게 제시되지 못했다. 본 연구는 지식의 분류를 위해 지식의 역할(role)과 출처(source)라는 두 가지 기준을 정의한다. 우선 지식은 역할에 따라 '*문제 해결 지식*', '*도메인의 일반적인 지식*', '*지식의 명세화*' 세 가지로 분류된다. '문제 해결 지식'은 지식의 출처에 따라 '함수적 이론 지식', '데이터분석기반 지식', '규칙 기반 전문가 지식'의 타입으로 분류된다. 지식 구조에 대한 보다 자세한 설명은 4장에서 기술된다.

온톨로지 기반: KAON의 정의[7]는 온톨로지의 일반적인 구조에 대해서 정의하고 있다. 온톨로지의 구조를 이루는 요소인, 개념, 개념 간의 관계, 개념 간의 계층 구조, 관계 간의 계층 구조, 공리는 OBKF의 토대가 된다. 개념 및 개념 간의 관계는 '도메인의 일반적인 지식'의 기본 구조를 표현할 수 있다. 공리는 개념과 관계에 대한 의미를 구체화함으로써 '지식의 명세화' 역할을 한다. 또한 '특정 업무 지식'은 온톨로지 개념들의 인스턴스 간의 관계성을 나타내기 때문에 온톨로지 기반으로 정의된다고 할 수 있다.

일관된 지식 표현: 온톨로지의 개념과 그 관계를 상세하고 정확하며 모호성 없이 나타내기 위해서는 논리 기반의 표현으로 나타내는 것이 바람직하다[22]. OBKF에 사용될 논리 기반의 표현은 제품 개발 도메인의 복잡한 지식을 표현할 수 있어야 하며, 또한 적절한 추론 능력을 갖추어야 한다. Corcho와 Percz[23]은 여러 논리 기반의 언어들을 두고 표현력과 추론의 효율 면에서 비교해 보았다. 그 결과, 언어의 표현력과 추론의 효율 간에는 상충되는 관계가 있다고 하였다. 즉, 표현력이 강할수록 추론 능력은 떨어진다. 본 연구에서는 OBKF를 표현하는 수단으로써 FOL을 선택하였다. FOL 표현은 제품 개발 도메인의 공리는 물론, 개념과 그 관계를 표현하기에 충분하다. 또한 문제 해결 지식의 기반이 되는 추론 알고리즘[24]도 제공한다.

4. 세 가지 지식 레벨

본 논문에서는 앞서 언급한 온톨로지 기반의 제품 개발 지식인 '문제 해결 지식', '도메인의 일반적인 지식', '지식의 명세화'를 각각 도메인 특화 지식(Specialized Knowledge for Domain: SKD), 지식 맵(Knowledge Map: K-Map), 공리로 정의하고, SKD를 다시 세 가지 타입; 엔지니어링 함수, 전문가 지식, 데이터분석기반 지식으로 분류한다. 이와 같은 구조를 OBKF라 하며 Fig. 1과 같이 표현할 수 있다.

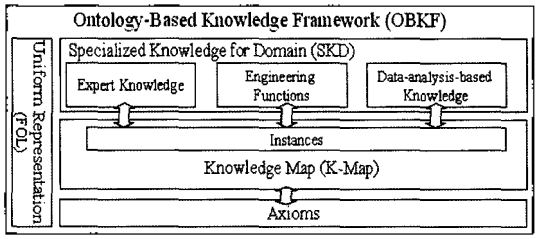

Fig. 1. 온톨로지 기반 지식 프레임워크의 아키텍처.

4.1 레벨 1: 공리

공리는 논리 표현을 통해 도메인에 존재하는 개념들과 관계들의 의미를 기술하여 사람과 컴퓨터가 각각의 의미를 명확하게 이해할 수 있게 한다. 공리를 통해, 개념들과 관계들의 기본적인 특성과 정의를 기술할 수 있게 된다. 예를 들어, 'subPartOf'라는 용어가 두 개의 부품 개념 간의 구조 관계를 표현한다고 가정하면 'subPartOf' 관계의 기본적인 특성들은 다음과 같다.

- 'subPartOf' 관계는 재귀적이지 않다; 한 부품은 그 자신의 하위 부품일 수 없다.
- 'subPartOf' 관계는 대칭적이지 않다; 부품 X가 부품 Y의 하위 부품이면, 부품 Y는 부품 X의 하위 부품일 수 없다.

- 'subPartOf' 관계는 추이적이다; 부품 X가 부품 Y의 하위 부품이고 부품 Y가 부품 Z의 하위 부품이면, 부품 X는 부품 Z의 하위부품이다.

개념들과 관계들의 정의 또한 논리 표현으로 기술될 수 있다. 간단한 예로, 'directSubPartOf'라는 용어가 두 개의 부품 개념간의 직접적 구조관계를 표현한다면, 두 부품 사이에 다른 부품이 구조관계로 연결되지 않는 것이라고 하자. 그러면 'directSubPartOf' 관계는 다음과 같이 논리적으로 정의할 수 있다.

- 부품 X가 부품 Y의 직접적 하위 부품이라는 것은, 부품 X가 부품 Y의 하위 부품이고, 부품 Z가 부품 Y의 하위 부품이고 부품 X가 부품 Z의 하위 부품인 부품 Z가 존재하지 않는다는 것이다.

공리	FOL 표현
1) '*subPartOf*' 관계는 재귀적이지않다.	$(\forall p)$ $\neg\, subPartOf(p, p)$
2) '*subPartOf*' 관계는 대칭적이지 않다.	$(\forall p1, p2)$ $subPartOf(p1, p2) \Rightarrow$ $\neg\, subPartOf(p2, p1)$
3) '*subPartOf*' 관계는 추이적이다.	$(\forall p1, p2, p3)$ $subPartOf(p1, p2) \wedge$ $subPartOf(p2, p3) \Rightarrow$ $subPartOf(p1, p3)$
4) '*directSubPartOf*' 관계의 정의	$(\forall p1, p2)$ $directSubPartOf(p1, p2) \Leftrightarrow$ $subPartOf(p1, p2) \wedge$ $\neg(\exists p3) \wedge subPartOf(p3, p2) \wedge subPartOf(p1, p3)$

Fig. 2. 공리 예.

우리는 이러한 공리들을 Fig. 2와 같은 FOL 표현으로 변환할 수 있다.

공리는 지식베이스의 일관성을 유지하는데 기여한다. 새로운 지식이 도입되거나 이전의 지식이 발전하여 수정 된다면 그 지식이 도메인에서 유효하기 위해서는 공리들을 위배해서는 안된다. 예를 들어, 부품 A가 부품 B의 직접적 하위 부품이라고 가정하면, 'directSubPartOf' 관계의 정의는 두 부품 사이에 다른 부품이 연결되는 것을 배제하게 된다.

4.2 레벨 2: 지식 맵(K-Map)

지식 맵은 의미 네트워크와 같은 구조를 통해 도메인의 공통된 지식을 표현한다. 지식 맵의 구조는 개념, 관계, 관계 함수, 개념 간의 계층 구조, 관계 간의 계층 구조 및 개념과 관계들의 인스턴스들로 이루어진다. 개념, 'Part'와 그 관계, 'subPartOf'는 실 세계의 개체와 그 관계를 표현한 말이다. 관계 함수 *Rel*은 개념과 그 관계 간의 관계성을 정의한다. 'subPartOf' 관계가 두 개의 'Part' 개념과 관련되어 있다면, 이것을 '*Rel*(subPartOf) = (Part, Part)'와 같은 논리 표현으로 나타낼 수 있다. 개념 간의 계층 구조는 용어집(taxonomy)이라고 불리기도 하는데 개념 간의 계층 구조 함수를 통해 표현할 수 있다. 개념 'Assembly'가 개념 'Part'의 하위 개념이라면 'H^C(Assembly, Part)'와 같이 표현될 수 있다. 또한 관계 간의 계층 구조는 함수 'H^R'에 의해 표현될 수 있다. 예를 들어, 'directSubPartOf' 관계가 'subPartOf' 관계의 하위 관계라면, 'H^R(directSubPartOf, subPartOf)'과 같은 논

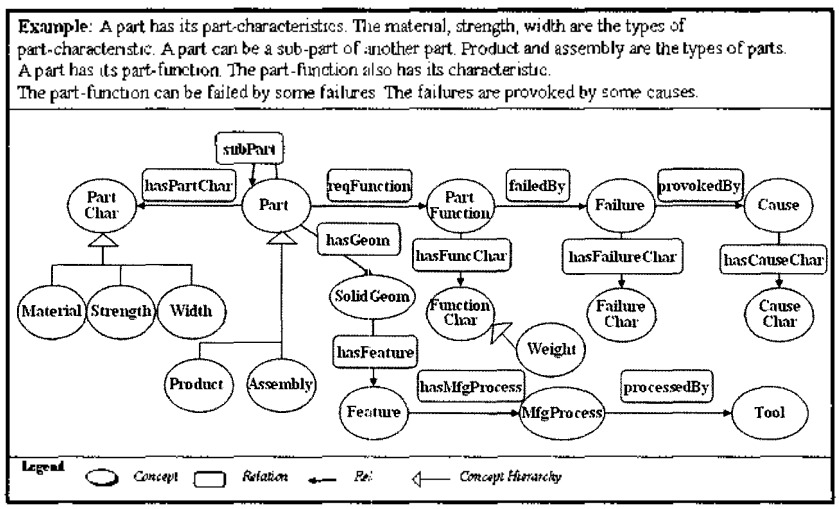

Fig. 3. 지식 맵 도식화 예.

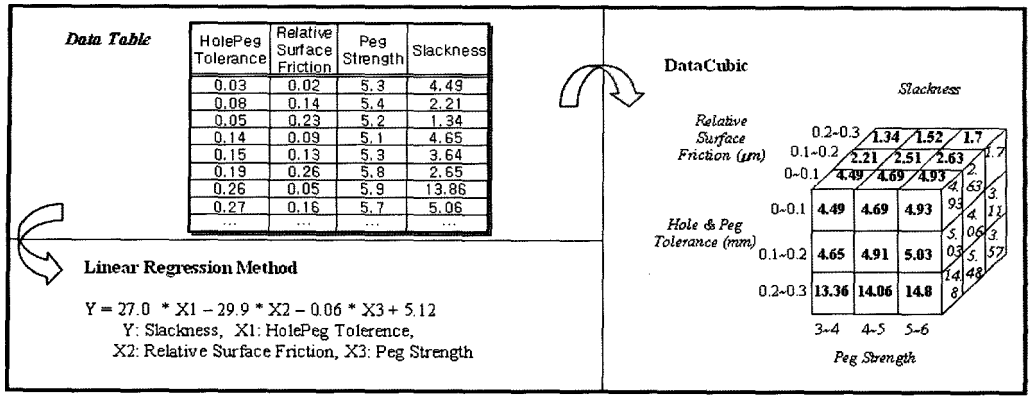

Fig. 4. SKD-3: 데이터분석기반 지식 예.

리 표현으로 나타낼 수 있다. 인스턴스는 앞서 정의한 개념들이 인스턴스화된 개념으로 'Part'에 대한 인스턴스는 Part(Chair), Part(Leg)와 같이 표현될 수 있다. Fig. 3은 제품 개발 도메인의 지식 맵 예제를 나타낸다. 그림에서 관계 간의 계층 구조와 인스턴스들은 생략되어 있다. 또한, 지식 맵은 FOL 형식으로 표현될 수 있는데 웹싸이트[25]에 FOL 표현을 설명해두었다. 지식 맵은 도메인 전문가가 동의해야 하며 동의 후에 SKD의 베이스가 될 수 있다. 동의된 지식 맵은 도메인의 참조 모델로써 사용될 수 있다.

4.3 레벨 3: 도메인 특화 지식(SKD)

SKD는 사용자의 문제를 해결하는 역할을 수행하며 의사 결정을 돕는다. SKD는 특정 업무나 문제에 관한 것이기 때문에 SKD의 논리적 표현은 지식 맵의 개념과 그 관계 그리고 인스턴스를 통해 이루어진다. 따라서, SKD는 도메인의 지식 맵에 기초로 하여 표현되고 지식 맵의 인스턴스 간의 관계에 대한 지식을 제공한다. SKD는 지식의 출처를 기준으로 엔지니어링 함수, 전문가 지식 그리고 데이터분석기반 지식의 세 가지 유형으로 분류될 수 있다.

첫 번째, 엔지니어링 함수는 과학적 이론으로부터 나오고 수학적인 표현으로 나타내어진다. 'Leg' 부품의 너비와 다른 요인들로 이루어진 다음의 방정식이 좋은 예이다. Ws, X, St 그리고 SF가 지식 맵의 '개념'이다. 엔지니어링 함수는 지식 맵의 개념들 간의 엔지니어링 관계를 수식으로 표현한 것이다.

(SKD-1): $X \geq (Ws / St * SF)^{1/2}$
X: width of a Leg part Ws: student's weight
St: material's tensile strength SF: Safety Factor

두 번째, 전문가 지식은 전문가의 경험이나 직관으로부터 나온 지식을 나타내며 일반적으로 정성적 지식이 된다. 전문가 지식은 일반적인 규칙처럼 'IF.. THEN..'의 형태를 가진다. 다음의 논리 표현이 전문가 지식의 예이다.

(SKD-2):
- IF feature.type = Closed_circular_hole
  THEN mfgProcess= Drilling.
- IF mfgProcess=Drilling AND feature.material = Wood THEN tool = HandDrill.

예에서, 'feature.type', 'part.material' 그리고 'tool'은 지식 맵의 '개념'이고 'Closed_circular_hole', 'Wood', 'Drilling', 'HandDrill'은 그 개념들의 인스턴스이다.

마지막으로 데이터분석기반 지식은, 데이터 분석을 통해 얻어지므로 수집된 데이터 및 데이터 분석 기법과 관련되어 있다. 예를 들어, 'FailureChar'의 인스턴스인 'Slackness'와 'CauseChar'의 인스턴스들인 'Hole & Peg Tolerance', 'Relative Surface Friction', 'Peg strength'에 대한 여러 데이터가 수집되어 데이터 테이블이 구성되었다면, 수집된 데이터를 바탕으로 각 속성들 간의 관계성을 분석하게 된다. 데이터 테이블에 대한 분석 결과로 'DataCubic'이 생성될 수 있고, 선형 회귀 분석외에 여러 데이터 마이닝 분석방법들을 통하여 명확한 지식을 정의할 수 있다. Fig. 4에서는 데이터분석기반 지식을 도출한 예를 보여주고 있다.

SKD의 예는 FOL 표현을 이용하여 표현될 수도 있는데 Fig. 5는 각 유형의 SKD에 대한 FOL 표현 예를 보여주고 있다.

Type of SKD	FOL formulae of examples
엔지니어링 함수 (SKD-1):	*(∀w1 w2 w3) Width(w1) ∧ Weight(w2) ∧ MaterialStrength(w3) ∧ (SF = 3) ∧ w1 ≥ squar((* (/ w2 SF) w3)).*
전문가 지식 (SKD-2):	*(∀x y z1 p) HasFeature(x y) ∧ Part(x) ∧ Feature(y) ∧ HasFeatureChar (y z1) ∧ FeatureType (z1) ∧ (= z1 ClosedCircularHole) ⇒ (= p Drilling) (∀y z2 p t)HasFeatureChar(y z2) ∧ Material(z2) ∧ (= z2 Wood) ∧ HasMfgProc(y p) ∧ MfgProc(p) ∧ ProcessedBy(p t) ∧ Tool(t) ∧ (= p Drilling) ⇒ (= t HandDrill)*
데이터 분석 기반 지식 (SKD-3):	*(∀y x1 x2 x3) Slackness(y) ∧ HolePegTolerence(x1) ∧ RelativeSurfaceFriction(x2) ∧ PegStrength(x3) ∧ (= y (+ 5.12 (+ (* (-0.06) x1) (+ (* (-29.9) x2) (* 27.0 x3)))))).*

Fig. 5. SKD의 FOL 표현 예.

5. 온톨로지 기반 지식 프레임워크의 프로토타입

5.1 OBKF 프로토타입

우리는 prolog를 이용하여 OBKF의 FOL 표현을 구현해보았다. 이번 구현에서 술어, 변수, 함수, 논리적 연계성 등이 prolog 형식으로 표현되는데 FOL에서의 표현과는 차이가 있다. 예제의 prolog 표현은 웹싸이트[26]에 전부 나타내어져 있다. 또한 프로토타입으로 어플리케이션을 개발하였는데 어플리케이션의 주요 화면이 Fig. 6에 나타나 있다. 어플리케이션은 사용자가 온톨로지를 수정할 수 있게 해주고 온톨로지를 시각적으로 볼 수 있게 해주며 쿼리를 수행할 수 있게 해준다. 우리는 prolog의 쿼리를 실행하기 위해 SICSTUS-prolog 프로그램과 Java를 사용하였고 윈도우 폼으로 온톨로지를 보여주기 위해 C#을 이용하였다.

쿼리를 수행함에 있어, OBKF의 각 레벨에 대한 상세한 역할은 다음 절에서 설명된다.

5.2 OBKF 프로토타입를 통한 쿼리 수행

구현된 OBKF는 엔지니어의 질문에 답하는 데에 사용된다. 우리는 몇 개의 쿼리 예제를 바탕으로 OBKF의 유용함을 보여주고자 한다. 쿼리가 수행되는 동안 공리, 지식 맵 그리고 SKD의 역할에 초점을 맞출 것이다. 예제는 'Chair' 제품에 대한 것으로써 기본적으로 입력되는 지식맵은 다음과 같다.

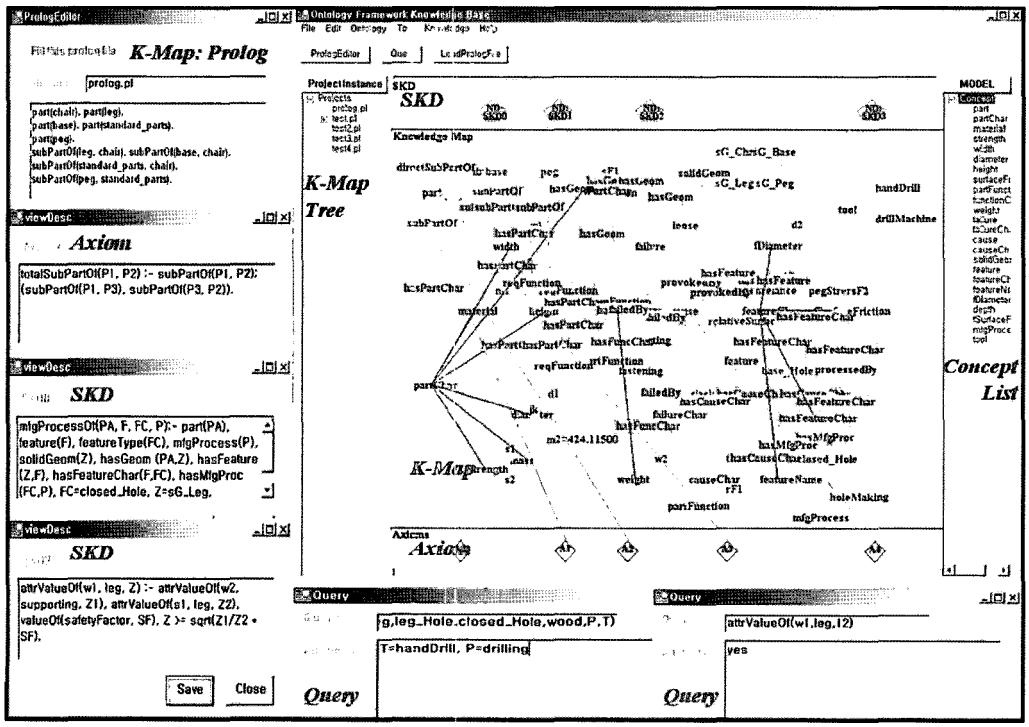

Fig. 6. OBKF 프로토타입 화면.

Knowledge Base

part(chair). part(leg). part(base).
part(standard_parts). part(peg).
material(m1). solidGeom(sG_Leg).
feature(leg_Hole). mfgProcess(drilling).
featureType(closed_Hole). tool(handDrill).
materialStrength(w3). width(w1).
partFunction(supporting).
failure(loose). failureChar(slackness).
cause(holePegTorelance).
cause(relativeSurfaceFriction).
cause(pegStrength). causeChar(t1).
causeChar(rF1). causeChar(pS1). weight(w2).
subPartOf(leg, chair). subPartOf(base, chair).
subPartOf(standard_parts, chair).
subPartOf(peg, standard_parts).
hasGeom(leg, sG_Leg).
hasFeature(sG_Leg, leg_Hole).
hasFeatureChar(leg_Hole, m1).
hasFeatureChar(leg_Hole, closed_Hole).
hasPartChar(leg, w1).
reqFunction(chair, supporing).
hasFuncChar(supporting, w2).
provokedBy(slackness, holePegTorelence).
provokedBy(slackness, relativeSurfaceFriction).
provokedBy(slackness, pegStrength).
hasCauseChar(holePegTorelence, t1).
hasCauseChar(relativeSurfaceFriction, rF1).
hasCauseChar(pegStrength, pS1).
valueOf(safetyFactor, 10). attrValueOf(w3, leg, 5).
attrValueOf(w2, supporting, 50).
attrValueOf(pS1, pegStrength, 5.3).
attrValueOf(t1, holePegTolerence, 0.03).
attrValueOf(rF1, relativeSurfaceFriction, 0.02).

Query 1: (지식 맵을 위한 예)
Query in English: What are the sub-parts of a 'chair' part?
Query in Prolog: ?- subPartOf(X, chair).
Outputs: X = leg; base; standard_parts; peg.
Related Axiom: 'subPartOf' 관계를 설명하는 추이적 공리는 시스템이 제품 구조를 고찰할 수 있게 한다.
Related K-Map: 관계 함수, subPartOf(P1, P2) : - part(P1), part(P2),는 'subPartOf' 관계의 각 변수가 part인지 검증한다.
Related SKD: 없음.

Query 2: (전문가 지식의 예)
Query in English: What method is appropriate for making a closed hole of a 'leg' part?
Query in Prolog: ?- mfgMethod (leg, P, T).
Outputs: P = handDrill, T = handDrill.
Axioms: 부품은 최소한 하나의 입체 형상을 가져야 한다는 공리는 지식 맵의 완전성을 보장한다.
part(X) :- solidGeom(Y), hasGeom(X, [Y|_]).
K-Map: 모든 입력값은 지식 맵으로부터 나온다. 모든 관계 함수는 변수가 옳은지 검증한다.
SKD: Fig. 5의 전문가 지식 SKD-2가 적용된다. 이를 Prolog로 표현하면 다음과 같다.
mfgMethod(PA, PR, T):- mfgProcessOf(PA, PR), mfgToolOf(PA, PR, T).

mfgProcessOf(PA, PR):- part(PA), solidGeom(S), hasGeom(PA,S), hasFeature(S,F), feature(F), featureType(FC), FC == closed_Circular_Hole, mfgProcess(PR), PR == drilling, hasMfgProcess(F, PR).

mfgToolOf(PA, PR, T):- part(PA), material(M), hasPartChar(PA, M), attrValueOf(M, wood), solidGeom(S), hasGeom(PA,S), hasFeature(S,F), hasMfgProcess(F,PR), PR == drilling, tool(T), T == handDrill, processedBy(PR, T).

Query 3: (엔지니어링 함수의 예)
Query in English: If the width of 'leg' part is 6, then the width satisfies constraints?
Query in Prolog: ?- check_attrValueOf(w1, 6).
Outputs: yes.
Axioms: 공리는 지식 맵의 완전성을 보장한다. Prolog 생략.
K-Map: query 2에서 지식 맵의 역할과 같다. Prolog 생략.
SKD: Fig. 5의 엔지니어링 함수 SKD-1이 적용된다. 이를 Prolog로 표현하면 다음과 같다.
check_attrValueOf(w1,Z) :- attrValueOf(w2, Z1), attrValueOf(w3, Z2), attrValueOf(safetyFactor, SF), Z >= sqrt(Z1/Z2 * SF).

Query 4: (데이터분석기반 지식의 예)

Query in English: What is the value of 'slackness' of the 'loose' failure?

Query in Prolog: ?- check_attrValueOf(slackness, X).

Outputs: X=5.014

Axioms: 원인은 자신이 초래한 고장과 관계를 갖는다. causes(X) :- failure(Y), provokedBy(Y, [X|_]). 지식맵은 공리를 통해 인증을 받는다.

K-Map: query 2에서 지식 맵의 역할과 같다.

Prolog 생략.

SKD: Fig. 5의 데이터분석기반 지식 SKD-3이 적용된다.

```
check_attrValueOf(slackness,Z) :-
attrValueOf(t1,X1), attrValueOf(rF1,X2),
attrValueOf(pS1,X3),
Z is (27.0*X3-29.9*X2-0.06*X1+5.12).
```

5. 결 론

우리는 제품 개발환경에서 엔지니어의 지식을 체계적으로 저장하고 활용하기 위한 온톨로지 기반 지식 프레임워크(OBKF)를 제안하였다. OBKF가 지식관리를 위한 프레임워크가 될 수 있는 이유는 엔지니어 지식에 대한 명시적인 구조를 제공하고, FOL 기반의 동일한 표현언어로 모델링 되기 때문이다.

비록 OBKF가 도메인의 근본적인 개념과 관계의 의미와 도메인의 지식들을 일관되게 표현할 수 있는 구조를 가지고 있더라도, 이를 상업적으로 실용성있게 활용하기 위해서는 여전히 더 많은 연구가 필요하다. 통합 제품 온톨로지에 대한 연구와 OBKF에 대한 성공적 적용사례들이 있다면 협업적 제품개발 환경을 위한 지식관리부분에 대한 다른 연구들도 더욱 발전할 수 있을 것이다.

감사의 글

본 연구는 산업자원부에서 주관하는 성장동력, 중기거점 신기술 개발 사업의 중간 결과물이며, 지원해 주신 여러 관계자 분들께 감사를 드립니다.

참고문헌

1. Syan, C. S. and Menon, U., "Concurrent Engineering: Concepts, Implementation and Practice", Chapman & Hall, 1994.
2. Kuffner, T. A. and Ullman, D. G., "The Information Requests of Mechanical Design Engineers", *Design Studies*, Vol. 12, No. 1, pp. 42-50, 1997.
3. Stauffer, L. A. and Ullman, D. G., "Fundamental Process of Mechanical Designers Based on Empirical Data", *Journal of Engineering Design*, Vol. 2, pp. 113-125, 1991.
4. Studer, R., Benjamins, V. R. and Fensel, D., "Knowledge Engineering: Principles and Methods", *Data & Knowledge Engineering*, Vol. 25, No. 1/2, pp. 161-197, 1998.
5. Wache, H., Vogele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H. and Hubner, S, "Ontology-based Integration Information: A Survey of Existing Approaches", *IJCAI Work-shop on Ontologies and Information Sharing* 2001.
6. Gruber, T. R., "A Translation Approach to Portable Ontology Specifications", *Knowledge Acquisition*, Vol. 5, pp. 199-220, 1993.
7. Bozsak, E., Ehrig, M., Handschuh, S., Hotho, A., Maedche, A., Motik, B., Oberle, D., Schmitz, C., Staab, S. and Stojanovic, L., "KAON - Towards a Large Scale Semantic Web", *Lecture Notes in Computer Science*, No. 2455, pp. 304-313, 2002.
8. Yoshioka, M. and Tomiyama, T., "Pluggable Metamodel Mechanism: A Framework of an Integrated Design Object Modelling Environment", *Computer Aided Conceptual Design '97, Proceedings of the 1997 Lancaster International Workshop on Engineering Design CACD'97*, pp. 57-70. Lancaster University, 1997.
9. Cutkosky, M. R., Engelmore, R. S., Fikes, R. E., Genesereth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M. and Weber, J. C., "PACT: An Experiment in Integrating Concurrent Engineering Systems", *Computer*, Vol. 26, No. 1, pp. 28-37, 1993.
10. Kuokka, D. R., McGuire, J. G., Pelavin, R. N. and Weber, J. C., "SHADE: Technology for Knowledge-Based Collaborative Engineering", *Artificial Intelligence in Collaborative Design*, pp. 245-262, 1994.
11. Lin, J., Fox, M. S. and Bilgic, T., "A Requirement Ontology for Engineering Design", *Concurrent Engineering, Research, and Applications*, Vol. 4, No. 3, pp. 279-292, 1996.
12. Kitamura, Y., Sano, T. and Mizoguchi, R., "Functional Understanding Based on an Ontology of Functional Concepts", *Lecture Notes in Computer Science*, No. 1886, pp. 723-733, 2000.
13. Borst, P., Akkermans, H. and Top, J., "Engineering Ontologies", *International Journal of Human-computer Studies*, Vol. 46, No. 2/3, pp. 365-406, 1997.
14. Gruber, T. R. and Olsen, G. R., "An Ontology for Engineering Mathematics", *Proceedings Fourth International Conference on Principles of Knowledge Representation and Reasoning*, pp. 258-269, 1994.
15. Kingston J. K. C., "Designing Knowledge Based

System: The CommonKADS Design Model", *Knowledge-Based System*, Vol. 11, pp. 311-319, 1998.
16. Angele J., Fensel D., Landes D. and Studer R., "Developing Knowledge-based Systems with MIKE", *Automated Software Engineering*, Vol. 5, pp. 389-418, 1998.
17. Noy, N. F., Fergerson, R. W. and Musen, M. A., "The Knowledge Model of Protégé-2000: Combining Interoperability and Flexibility", *Knowledge Engineering and Knowledge Management*, pp. 17-32, 2000.
18. Fikes, R., Farquhar, A. and Rice, J., "Tools for Assembling Modular Ontologies in Ontolingua", *Artificial Intelligence*, pp. 436-441, 1997.
19. Staab, S. and Maedche, A., "Axioms Are Objects, Too: Ontology Engineering Beyond the Modeling of Concepts and Relations", *Workshop on Ontologies and Problem-Solving Methods*, Berlin, 2000.
20. Ferguson, E. S., Engineering and the Mind's Eye, MIT Press, Cambridge, MA, 1992.
21. Vincenti, W. G, What Engineers Know and How they Know It: Analytical Studies from Aeronautical Engineering, John Hopkins University Press, 1990.
22. Heflin, J., OWL Web Ontology Language Use Cases and Requirements W3C Recom-mended, February, 2004, http://www.w3.org/TR/webont-req/.
23. Corcho, O. and Perez, A. G., "A Roadmap to Ontology Specification Languages", *Lecture Notes in Computer Science*, No. 1937, pp. 80-96, 2000.
24. Russel, S. and Norvig, P., "Artificial Intelligence, 2nd edition", Prentice Hall, 272-315, 1995.
25. Lee, J. H. (September 1, 2004), http://143.248.82.98/CADCAM/FOL.doc.
26. Lee, J. H. (September 1, 2004), http://143.248.82.98/CADCAM/PROLOG.doc.
27. 김경영, 서효원, "CPC 환경을 위한 Product 온톨로지 기반의 의미 공유 접근법", 한국캐드캠학회 논문집, 제9권, 제3호, pp. 192-202, 2004.
28. 김현, 김형선, 이주행, 정진미, 도남철, 이재열, "기업간 제품정보 공유를 위한 협업적 제품거래 프레임워크", 한국캐드캠학회논문집, 제8권, 제4호, pp. 201-211, 2003.
29. 이윤숙, 천상욱, 한순흥, "CAD 시스템 간의 상호 운용성을 위한 설계 특징형상의 온톨로지 구축", 한국캐드캠학회논문집, 제9권, 제2호, pp. 164-174, 2004.

서 효 원

1981년 연세대학교 기계공학과 학사
1983년 한국과학기술원 기계공학과 석사
1991년 West Virginia University 산업공학과 박사
1983년~1987년 대우중공업(주) 중앙연구소 주임연구원
1992년~1995년 생산기술연구원 생산시스템센터 수석연구원
1996년~현재 한국과학기술원 산업공학과 교수
관심분야: CE/PDM/CPC/PLM, Workflow Management/BPM, Ontology/Knowldge Based System

이 재 현

1999년 한국과학기술원 산업공학과 학사
2000년 한국과학기술원 산업공학과 석사
2001년~현재 한국과학기술원 산업공학과 박사과정 재학
관심분야: 제품 정보 관리, 온톨로지기반 지식관리시스템