

Ontology와 XML 기반의 제품 데이터 관리

- Product Data Management Based on Ontology and XML -

한 영 근 *

Han Young Geun

조 진 형 **

Joh Jin Hyung

Abstract

In this research, OIL(Ontology Inference Layer), one of the ontology language, is applied for classifying product data systematically, defining concepts, and establishing relationship between concepts. By transforming steel product data into XML documentation and managing them, knowledge management based on the logical structure of documents is possible.

Keyword : OIL, XML, knowledge management

1. 서론

인터넷 사용의 급속적인 증가로 조직이 분산되고, 정보 공유를 위해 정보의 구조적 관리가 필요로 하게 되었다. 최근 요구되는 새로운 전자문서의 형태는 워드프로세서나 사용하는 플랫폼에 대하여 독립적이며, 정보의 전송과 교환이 용이하고, 문서의 의미를 그대로 나타낼 수 있는 프레임워크이다. 차세대 인터넷 표준으로 채택된 XML(eXtensible Markup Language)은 위 조건을 만족시킨다. 이것은 기존의 HTML(HyperText Markup Language)과 SGML(Standard Generalized Markup Language)의 장점은 수용하고 단점은 보완하기 위해 등장한 표준이다[1]. HTML은 웹 상에서 간단히 문서 생성이 가능하고, 높은 이식성을 가지지만, DTD(Document Type Definition)가 고정되어 있어 정해진 태그만 사용가능하다는 단점이 있는 반면, SGML은 문서의 논리 구조 정의 및 사용자 정의 태그 사용이 가능하지만, 사용방법이 어렵고 웹에서 지원되는 응용프로그램이 부족한 단점을 가지고 있다. XML은 일반 문서와 달리 문서의 내용뿐 아니라 논리적 구조를 나타낼 수 있고, 다양한 형태의 미디어 역시 포함할 수 있으며, 문서의 논리적 구조를 통해 체계적으로 문서관리를 할 수 있다.

† 본 연구는 2002년 한국과학재단 산학협력연구과제의 지원으로 수행되었음.

* 명지대학교 산업시스템공학부

** 한국하우톤

이러한 장점을 통해서 XML은 여러 분야에서 활용 가능하지만, 지식 공유나 재활용을 위해서 XML 형태의 문서나 데이터의 의미를 정확히 알아야만 가능하다.

대개 XML이 사용자정의 마크업 언어이기 때문에 문서 자체만으로 그 의미를 알 수 있는 것처럼 과대평가 되는 경향이 있는데, XML은 문서 구조를 상세하게 설명하기 위한 표현 언어일 뿐이지 그 의미와 구조를 이해하기 위해서는 DTD나 XML 스키마를 이용해야 한다. 그러나 그 의미를 정확하게 이해하기 위해서는 개념을 정의하고 그 개념간의 관계를 명확히 규정할 수 있는 도구로서 온톨로지(Ontology) 개념을 이용하는 것이 바람직하다.

예를 들어, 온톨로지를 통하여 business rule과 같은 도메인의 용어를 명확하게 정의하고, 그들 간의 관계를 정의할 수 있기 때문에 도메인을 체계적으로 관리할 수 있다. 설계문서들은 제품의 구성요소들 각각에 대한 정보와 구성요소들 간의 상관관계에 따라 존재하기 때문에 이러한 문서들을 온톨로지를 통해서 구성요소 각각에 대한 정보와 상관관계를 정의함으로써 체계적으로 분류하고 관리할 수 있다.

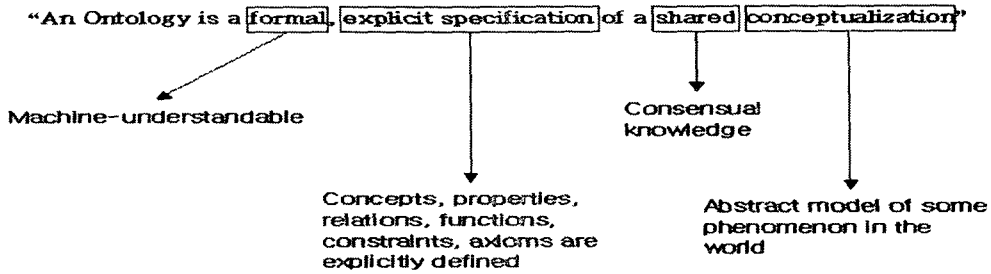
본 논문에서는 제품의 개발과정 및 제조과정 등에서 발생하는 설계지식을 XML 문서화하여 체계적으로 관리하는 방법을 제시한다. 설계지식을 XML 형태의 구조적 정보로 표현하기 위해서 도메인의 개념적 용어를 정의하는 방법으로 최근 인공지능 분야에서 지식표현 방법으로 활용되고 있는 방법인 온톨로지를 활용한다. 여러 가지 온톨로지 언어들 중에서 새로운 온톨로지의 웹 표준으로 부각되고 있는 OIL(Ontology Inference Layer)개념을 적용하여 설계 지식간의 상관관계를 도출하고, 지식 관리에 활용하여 설계지식의 변화에 대한 효율적인 대응과 기존의 설계지식을 토대로 새로운 부가가치를 지닌 지식의 도출에 활용하는 방안이 연구된다.

2. 온톨로지

온톨로지(Ontology)는 원래 철학적인 개념으로는 “존재에 관한 체계적인 이론(존재론)”이라는 의미로 사용되어져 왔다. 그러나, 최근에는 인공지능 분야에서 “개념화의 명시적인 기술” 즉, 관심을 가진 대상에 대한 명확한 개념과 그들 상호간의 관계를 명확하게 기술한다는 의미로서, 지식의 공유와 재사용을 용이하게 하기 위해 사용되어 왔다.

2.1 온톨로지의 개념

온톨로지의 개념은 위에서 언급한 것 이외에도 적용분야나 사용자에 따라 정의하는 개념의 차이는 있지만 일반적으로 대략 10가지 정도로 분류된다. 최근 컴퓨터과학 분야에서 이용되는 온톨로지의 본질을 가장 잘 나타낼 수 있는 개념으로는 <그림 1>과 같이 “An Ontology is a formal, explicit specification of a shared conceptualization”이다. “formal”의 의미는 “기계가 이해할 수 있는(machine-understandable)”, “explicit specification”은 “개념, 관계, 기능, 속성, 공리의 명확한 정의”이며, “shared”의 의미는 “협의된 지식”, “conceptualization”의 의미는 “실세계에 있는 어떤 현상의 추상적인 모델”을 나타낸다.



< 그림 1 > Ontology의 정의

2.2 응용분야

최근 온톨로지 개념이 많이 이용되고 있는 세 가지 주요 응용분야는 지식관리, 전자상거래, 전자 비즈니스 분야이다[2].

지식관리는 조직의 지식을 획득하고, 유지하고, 이용하는 것과 많은 관계가 있는데 지식관리의 목적은 더 큰 생산성과 새로운 가치, 그리고 증가된 경쟁력을 위한 조직의 지적 자산을 이용하는 것이다. 점차 현대 사회의 세계화와 인터넷의 영향으로 대부분의 조직이 점차 지역적으로 분산되고, 가상집단(virtual teams)으로 조직화되고 있어 수많은 온라인 문서의 관리가 필요하게 되었다. 이러한 온라인 문서의 관리를 위해서 많은 문서관리 시스템이 시장에 출시되었지만, 관리 시스템은 <표 1>에서와 같이 몇 가지 단점을 가지고 있다[2].

< 표 1 > 기존 문서관리 시스템의 단점

단 점	내 용
정보검색	기존의 키워드 기반 검색은 원하는 내용과는 다른 의미로 사용되는 관련 없는 정보도 역시 추출한다.
정보추출	자동화된 에이전트가 본문의 표현에서 정보를 추출하기 위해 필요한 지식의 부족으로 분산된 정보를 통합하지 못한다.
정보유지	완벽하게 구조화되지 않은 텍스트 자원의 유지는 어렵고, 자원이 많아지면 시간을 낭비할 수 있다.
자동문서 생성	구조화된 데이터를 구조화된 정보로 생성하기 위해서는 기계가 이해할 수 있는 표현이 필요하다.

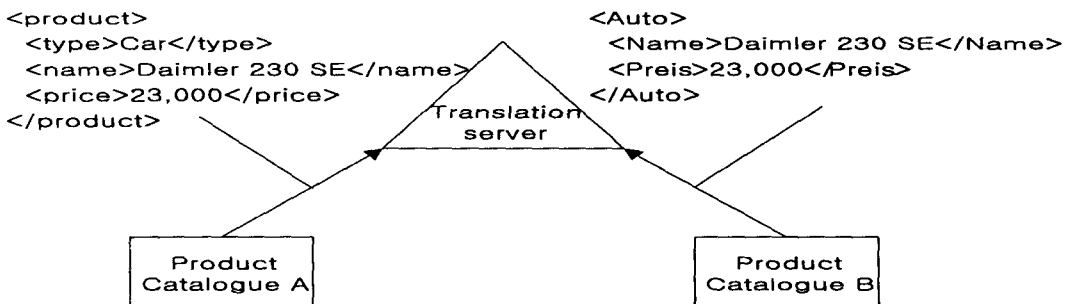
기존의 관리 시스템은 정보들이 일정한 형식이나 구조를 가지고 있지 않기 때문에 정보관리 측면에서 불필요한 정보검색이나 정보추출로 인해서 비효율적이다. 이러한 기존 문서관리 시스템의 문제점을 해결하기 위해서 온톨로지 개념을 이용하는데, 온톨로지는 문서의 구조적·의미적 정의를 명확히 하고, 이를 통해서 새로운 가능성 즉, 키

워드 매핑 대신에 기계가 처리할 수 있는 검색이 가능하고, 정보추출 대신에 query answering, 온톨로지 매핑을 통한 부서간의 문서 교환 등이 가능하도록 한다[2].

다음은 전자상거래 분야인데, 이것은 다음 두 가지 이유로 비즈니스 분야에서 중요시되며, 또한 성장하고 있다.

첫째로, 전자상거래는 기존의 비즈니스 모델을 확장한다. 이 모델은 비용을 줄이고, 기존의 유통 채널을 확장하고, 새로운 유통의 가능성을 제안하기도 한다. 둘째, 전자상거래는 새로운 형태의 비즈니스 모델을 가능하게 한다. 그 예로 사업 분야의 확장으로 온라인 상점이 있고, 새로운 비즈니스 영역은 쇼퍼에이전트와 온라인 marketplace 등이 있다. 제품 정보는 대부분 자연어로 제공되는데, 자연어의 자동적인 텍스트 인식은 아직까지 해결되지 못한 중요한 연구과제이다. 이를 보완하기 위해서 온톨로지는 소프트웨어 에이전트가 이해할 수 있는 제품정보를 생성하고, 필요한 정보에 대한 안내와 검색에 도움을 줄 것이다.

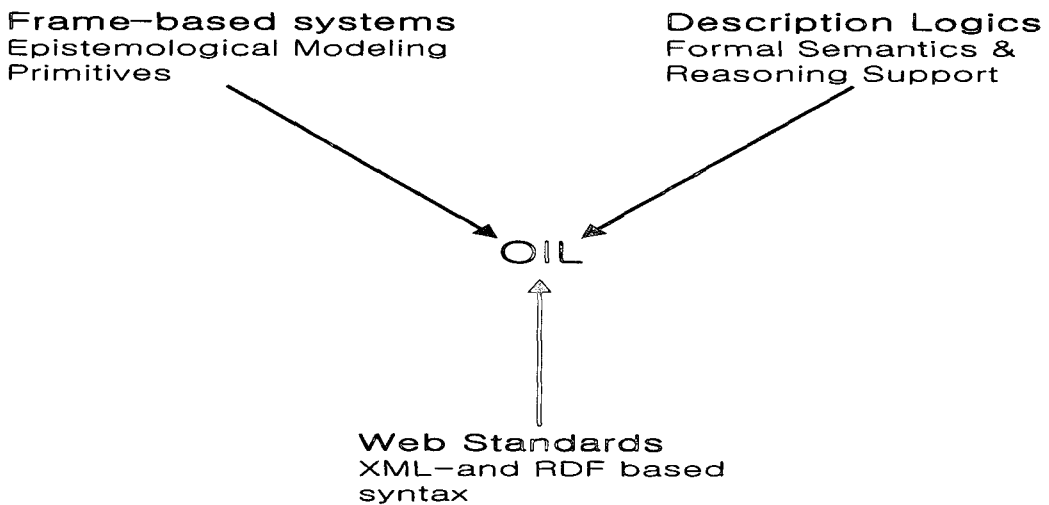
마지막으로 전자 비즈니스 분야이다. B2B분야에서 전자적 비즈니스 거래를 하기 위해서 송신자와 수신자는 거래내용을 설명하는 언어와 내용을 보낼 수 있는 프로토콜의 표준에 합의를 해야한다. 이런 목적을 위한 표준들이 많이 발표되었는데, 그들 중 하나가 바로 EDI의 표준인 UN/EDIFACT이다. 그러나 이들 EDI 표준들은 상호 독립적인 표준이기 때문에 서로 다른 표준을 이용하는 기업들간의 거래에서, 외부로의 비즈니스 문서나 정보들을 교환하고 통합하는 것은 사실상 불가능하다. 비즈니스 거래를 위해 인터넷을 이용함으로써 개선이 되었지만, HTML은 정보의 구조와 내용을 표현하지 못하는 한계점으로 풍부한 의미를 표현하지 못한다. 이를 해결하기 위해서 전자상거래에 XML을 기반한 온톨로지를 적용하면 <그림 2>와 같이 동일 차종에 대해 A와 B의 카탈로그가 다르게 정의되어 있어도, 온톨로지를 이용한 "Translation server"에 의해서 문서의 구조와 의미뿐만 아니라 언어의 차이점을 극복할 수 있다. 또한, 데이터 교환이 자동화될 수 있고, 서로 다른 구조의 데이터를 자신의 시스템에 적합한 형태로 변환할 수 있다.



< 그림 2 > 문서 구조, 의미, 언어의 변환

3. Ontology Inference Layer

OIL은 여러 분야의 단체에서 제공하는 다음의 세 가지 핵심 개념을 통합하고 있다[3, 4]. 세 가지 핵심 개념은 <그림 3>과 같이 첫 번째, 프레임 기반 시스템이다. 도메인 지식 표현을 위한 중요한 수단으로 프레임을 이용하는 지식 표현 시스템이다. 프레임은 지식을 소위 프레임이라 부르는 하나의 덩어리로 조직화하는 것이다.



< 그림 3 > OIL의 세 가지 핵심 개념

이들 프레임은 예를 들면 “living room”이나 “being in a living room”처럼 이러한 상황과 관련된 모든 정보를 통합함으로써 개념이나 실질적인 상황에 대한 정보를 얻기 위해 제안되었다. 프레임은 프레임을 어떻게 사용할 것인가에 대한 정보와 오류발생 가능성에 대한 정보 등을 포함하고 있는데, 이들 프레임의 집합이 상호 연결되어 프레임 시스템에서 조직화되는 것이다. 둘째, Description Logics(DL)은 개념(concepts)과 개념의 계층 구조에 관한 지식을 표현하기 위해 적합하도록 만들어진 지식표현 언어이다. 지식표현 연구의 주요 취지는 구조화된 지식을 표현하고, 정의된 방법으로 지식을 이용하고 추론하기 위한 이론과 시스템을 제공하는 것이다. 세 번째 핵심 개념은 Web standards이다. 현재 웹의 우수성과 중요성이 일반화되어 있기 때문에 지식표현을 위해서 웹 표준을 이용하여 온톨로지를 교환하기 위한 언어를 공식화해야 한다는 것을 의미하고 현재 W3C에서 제안하는 웹 표준으로는 XML과 RDF가 있다.

OIL은 위에서 언급한 세 가지 개념을 기반으로 온톨로지를 표현함으로써 사용자뿐 아니라 기계가 이해할 수 있는 매우 직관적인 언어를 이용하고, 완벽성·정확성·효율성을 위해 잘 정의된 의미를 보유하고, 상호 운용성을 위해서 XML과 RDF와 같은 웹 표준과 적절한 연계를 유지할 수 있다.

OIL은 크게 ontology-container와 ontology-definition의 두 부분의 엘리먼트로 구성되어 있다. ontology-container는 저자, 이름, 주제 등과 같은 온톨로지 특징을 설명하고,

ontology-definition은 온톨로지 정의를 위해 실제로 사용되는 용어를 정의하는 부분으로 명확히 정의된 의미를 가진 구조화된 어휘를 설명한다. 무엇보다 OIL 언어에서 가장 핵심이 되는 것은 바로 ontology-definition이다.

ontology-container는 Dublin Core Meta data Element Set 표준으로 15개의 엘리먼트 즉, title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coverage, rights로 구성되어 있다. <그림 4>는 ontology-container의 한 예로 "African animal"에 대한 엘리먼트 예이다.

```
ontology-container
title "Afriaca animals"
creator "Ian Horrocks"
subject "animal, food, vegetarians"
description "African animals"
description.release "1.01"
publisher "I. Horrocks"
type "ontology"
format "pseudo-xml"
```

< 그림 4 > ontology-container의 예

ontology-definition은 크게 class definition과 slot definition 두 부분으로 나눌 수 있는데, 우선 class definition을 먼저 살펴보겠다. class definition(class-def)은 클래스 이름과 그 설명을 연관시킨다. <그림 5>는 "elephant" ontology-definition의 한 예이다. class definition(class-def)은 클래스 이름과 그 설명을 연관시킨다.

```
class-def primitive elephant
subclass-of animal
slot-constraint skin-color
value-type grey
```

< 그림 5 > elephant의 ontology-definition의 예

slot-constraint는 slot에 적용되는 하나 이상의 제약조건 목록이다. 여기서 slot은 역할(role) 혹은 속성(attribute)이라고도 한다. 다음 <그림 6>은 animal에 대한 ontology-definition의 한 예이다.

```
class-def tree
subclass-of plant
class-def branch
slot-constraint is-part-of
has-value tree
class-def leaf
slot-constraint is-part-of
has-value branch
```

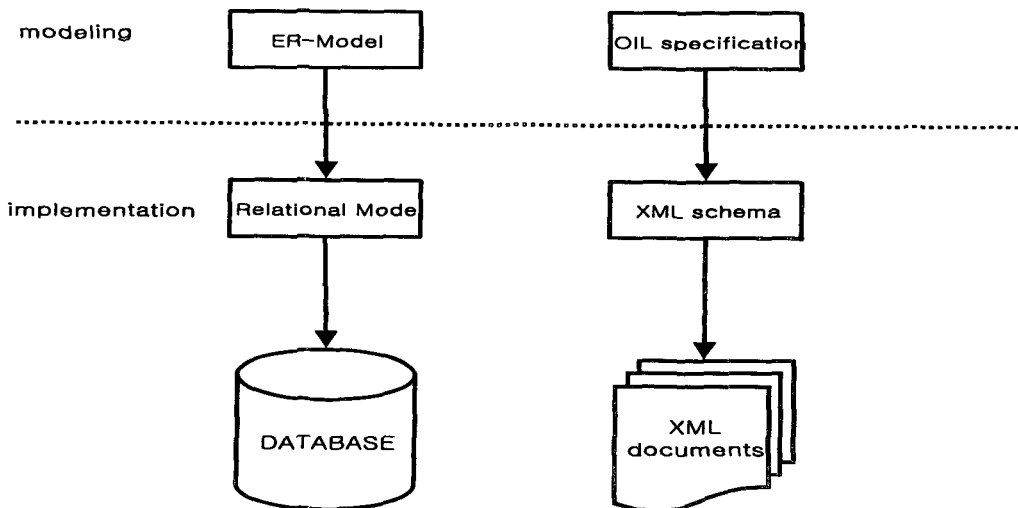
: trees는 plants의 한 종류이다.
: branches는 trees의 부분이다.
: leafs는 branches 의 부분이다.

< 그림 6 > ontology-definition의 예

4. OIL과 XML 스키마의 관계

온톨로지와 XML 스키마는 서로 다른 목적을 가지고 개발되었지만, 다른 한편으로 공통의 목적을 가지고 있다. 온톨로지 언어는 도메인 내용을 명확하게 설명하기 위한 수단이고, XML 스키마는 정보자원 즉, 문서나 반구조화된(semi-structured) 데이터에 대한 제한을 설명하는 수단이다. 또, OIL과 XML 스키마의 공통된 한 가지 주요 목적은 둘 다 정보 교환을 목적으로 하는 정보자원을 설명하기 위한 용어와 구조를 제공하는 것이다. 본 절에서는 온톨로지를 위한 웹기반 표현언어인 OIL과 웹 문서의 구조와 의미를 설명하기 위한 표준인 XML 스키마의 공통점과 차이점을 분석하고자 한다. 이 분석을 위해 Relational model과 Entity Relationship model(ER model)사이의 관계와 비교한다.

<그림 7>은 ER model과 Relational model, OIL과 XML 스키마의 관계를 모델링과 실행의 관점에서 설명하는 그림이다. Relational model은 실행 지향적인(implementation oriented) 데이터베이스의 형태를 제공하고 ER model은 응용을 위해서 필요한 모델링 정보자원을 위한 모델링 프레임워크를 제공한다. 시스템을 개발할 때 상위 단계인 ER model을 토대로 이 모델을 실행 지향적인 Relational model로 변환하여 사용하는 것과 같이 OIL은 모델링 프레임워크를 제공하고, XML 스키마는 실행 지향적인 형태를 제공하는 점에서 유사성을 가지고 있다. 아래의 <표 2>는 OIL과 XML 스키마의 비교결과이다. OIL과 XML 스키마의 가장 큰 차이점은 class와 slot을 구별하고, class(or slot) 정의는 계층구조를 이끌어 내기 위해 이용된다는 점에서 OIL이 더욱 풍부한 모델링의 기본을 제공하지만, datatype의 다양성과 엘리먼트의 내용을 구조화하기 위한 문법에 대해서는 XML 스키마가 더욱 풍부한 모델링의 기본을 제공한다. 사실상 XML 스키마는 도메인 모델을 설계할 때는 중요하지 않고, 문서의 구조를 정의할 때 중요하게 이용된다.



< 그림 7 > ER Model과 OIL Model과의 관계

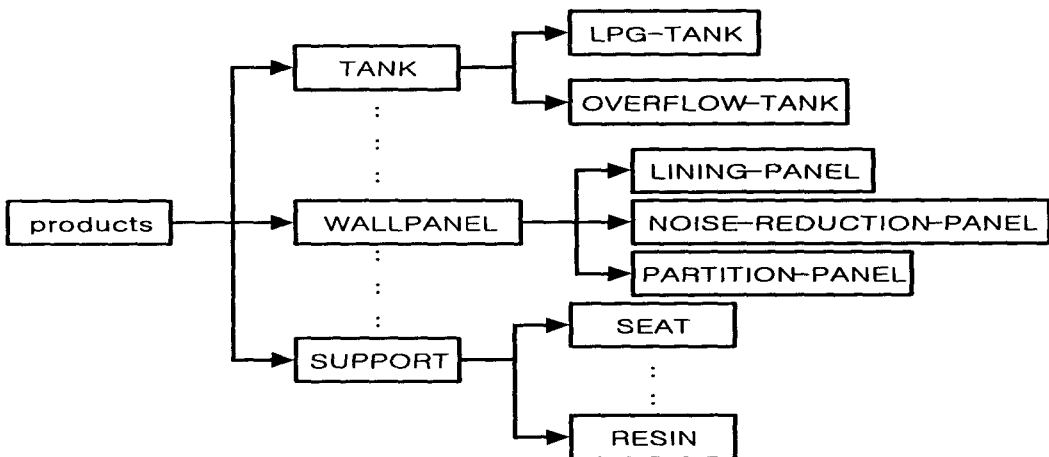
5. 설계지식의 OIL 및 스키마 생성

본 절에서 설명하는 설계지식은 제품정보 즉, 제품의 기능, 제품의 세부내용, 문제점, 개선사항 등을 말한다. 기존의 제품개발에 필요한 설계지식은 특정 설계 전문가에게 집중되어 있어서 새로운 설계자나 초심 설계자들에게 전달되는 것이 미비하고, 설계 노하우 등은 체계적으로 기업에 전산화되어 있지 않다. 또한 사용자가 원하는 정보를 찾기 위해서는 제품을 검색하고, 기능·제품에 대한 문제점과 개선사항들은 다시 검색해서 정보를 확인해야만 했다.

< 표 2 > OIL과 XML 스키마의 비교

구 분	내 용
공통점	<ul style="list-style-type: none"> • OIL과 XML 스키마는 둘 다 XML syntax를 가진다. • XML 스키마와 OIL은 모두 namespace를 제공한다.
차이점	<ul style="list-style-type: none"> • XML스키마는 풍부한 데이터 타입을 제공하지만, OIL은 그렇지 않다. • XML스키마의 모델링의 주요 기본 단위는 엘리먼트이고, OIL의 모델링의 주요 기본 단위는 concept와 slot이다. • XML스키마는 grammar를 제공하지만 OIL은 어떤 것도 제공하지 않는다.

<그림 8>은 본 절에서 예로 이용하는 한 기업의 제품(products)에 대한 계층구조를 나타낸 것이다. 이 그림을 통해서 “products”의 종류와 그 구성요소를 알 수 있다.



< 그림 8 > 사례제품의 구성

5.1 제품 설계지식에 대한 OIL의 정의

본 절에서는 설계지식을 OIL의 형식에 따라서 정의한다. 우선적으로 하는 것은 역시 OIL의 두 가지 요소 중에서 ontology-container 부분을 정의한다. <그림 9>는 container에 사용되는 15개의 엘리먼트 중에서 일부분을 이용해서 products에 대한 ontology-container를 정의한 부분이다.

```
ontology-container
  title "An Ontology of products"
  creator "cjhyong"
  description
  description.release "1.0"
  type "ontology"
  identifier "id"
  language "OIL"
  rights "cjhyong"
```

< 그림 9 > products의 ontology-container 정의

다음으로 ontology-definition 부분을 정의한다. <그림 10>과 같이 class가 사용하는 slot-def를 먼저 정의한 후, class-def를 정의한다. 앞에서 정의한 slot을 이용하여 "class-def"에서 class를 정의하고, "slot" 중에서 "slot- constraint"를 설정하여 하나의 class를 정의한다. <그림 11>은 products의 class-def의 예이다. 그 다음으로 정의된 OIL를 이용해서 사용자가 생성하려는 문서의 내용과 구조를 정의하는 XML 스키마 문서를 생성한다.

```
ontology-definition
  slot-def is-part-of
    inverse has-part
  slot-def product
  slot-def has-part
  class-def products
```

< 그림 10 > products의 slot 정의

```
class-def products
  slot-constraint product has-value string
class-def TANK
  subclass-of products
class-def SUPPORT
  subclass-of products
  slot-constraint has-part value-type SEAT
  slot-constraint has-part value-type RESIN
class-def LPG-TANK
  subclass-of TANK
  slot-constraint has-part value-type DECK
  slot-constraint has-part value-type T_BHD
class-def SEAT
  subclass-of products
  slot-constraint is-part-of has-value SUPPORT
```

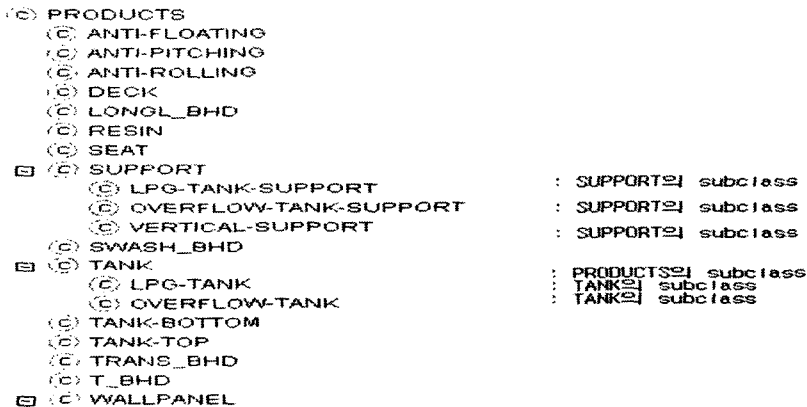
< 그림 11 > products에 대한 class-def 정의

5.2 OIL specification의 XML 스키마 변환

OIL의 XML 스키마 변환단계는 다음과 같이 크게 5가지이다.

- (1) 계층구조를 구체화한다.
- (2) OIL에서 각각의 slot을 정의한다.
- (3) OIL에서 각각의 class를 정의한다.
- (4) 각각의 slot과 class를 위한 엘리먼트 정의를 생성한다.
- (5) 엘리먼트의 세부사항을 정의한다.

위의 5단계에 따라서 OIL 중에서 products에 해당하는 부분을 XML 스키마로 변환하는 과정을 살펴보도록 하겠다. 첫 번째, 계층구조를 구체화한다는 것은 OIL로 정의된 class들을 <그림 12>와 같은 계층구조로 표현한다.



< 그림 12 > products의 OIL 구조화

두 번째, OIL에서 각각의 slot에 대한 타입을 정의한다. 앞에서 설명했듯이 class의 slot을 우선적으로 정의한 후에 class를 정의한다. <그림 13>에서 굵게 나타나는 글자가 products의 slot을 나타낸다. 이러한 slot type을 바탕으로 XML 문서의 인스턴스를 생성한다.

```

<xsd:complexType name="slotType" />
<xsd:complexType name="has-partType"
  base="slotType" />
  
```

< 그림 13 > OIL slot의 type 정의

세 번째, <그림 14>에서처럼 OIL에서 각각의 class에 대한 타입을 정의한다. 여기서는 두 번째와 같은 방법으로 정의한다.

```

<xsd:complex name="productType" />
  <xsd:element name="product">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element ref="TANK"/>
        <xsd:element ref="LPG-TANK"/>
        :
        <xsd:element ref="VERTICAL-SUPPORT"/>
        <xsd:element ref="SEAT"/>
      </xsd:choice>
      <xsd:attribute name="no" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:NMTOKEN">
            <xsd:enumeration value="001"/>
            :
            <xsd:enumeration value="013"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:element>
  </xsd:complexType>
  
```

< 그림 14 > OIL의 class type의 변환

```

<xsd:element name="TANK">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element ref="subclass" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element ref="has-part" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>

<xsd:element name="LPG-TANK">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element ref="subclass" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element ref="has-part" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>

<xsd:element name="has-part" type="xsd:string"/>
  
```

< 그림 15 > class와 slot의 정의

네 번째, 각각의 slot과 class를 위한 엘리먼트 정의를 생성한다. OIL에서 정의된 slot과 class를 XML 문서에서 사용될 엘리먼트로 생성한다. <그림 15>는 slot과 class의 엘리먼트 정의 예이다.

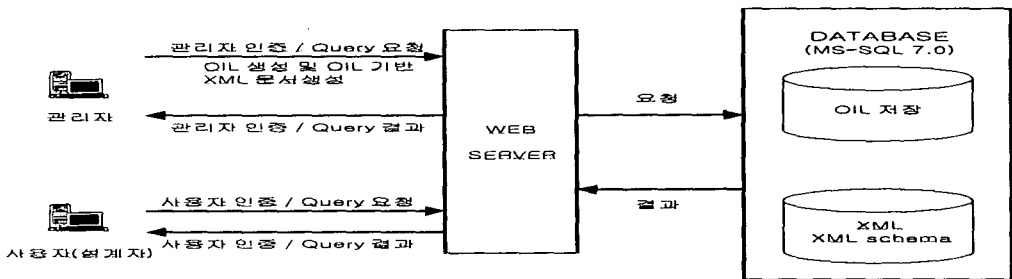
다섯 번째, 엘리먼트의 세부사항을 정의한다. 이것은 OIL에서 표현하지 못하는 문서 작성에 필요한 세부사항을 추가로 정의한다.

6. 설계지식 관리시스템 설계 및 구현

시스템의 구조는 우선 사용자가 검색하고자 하는 질의요구와 질의에 대한 결과를 제공하는 웹 클라이언트, 클라이언트의 요청을 받아들이는 웹 서버, 그리고 OIL 문서,

XML 문서, XML 스키마를 저장하고 있는 데이터베이스의 세 계층으로 구성되어 있다. 구현한 프로토타입 시스템의 구조는 <그림 16>과 같다.

사용자(설계자)가 원하는 부품에 대한 정보를 요청하면, 웹 서버에 질의요구를 넘겨주고, 웹 서버에서는 질의요구를 데이터베이스에 전달하게 된다. 검색되어진 결과는 다시 웹 서버로 전달되고, 웹 서버에서는 이 결과를 화면에 보여지게 된다. 사용자는 데이터베이스에서 검색한 결과를 가지고 이용하여 새로운 지식 생성 시 발생할 수 있는 동일한 오류의 반복을 막고 주의사항 등을 참조할 수 있다.



< 그림 16 > 시스템 구조

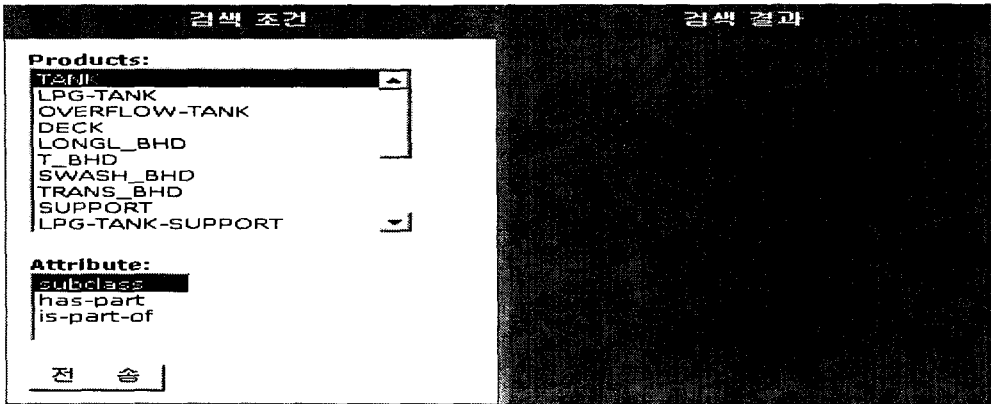
관리자는 사용자의 기능도 수행할 수 있고, 무엇보다도 개념에 대한 OIL을 생성/저장하고, 관리할 수 있으며, 이를 바탕으로 XML 스키마를 생성할 수 있고, 이를 가지고 설계 지식을 XML 문서화하여 보관할 수 있다. 관리자는 이렇게 생성된 지식을 공유하고 보존하기 위한 중요한 역할을 수행한다.

6.1 시스템의 수행 예

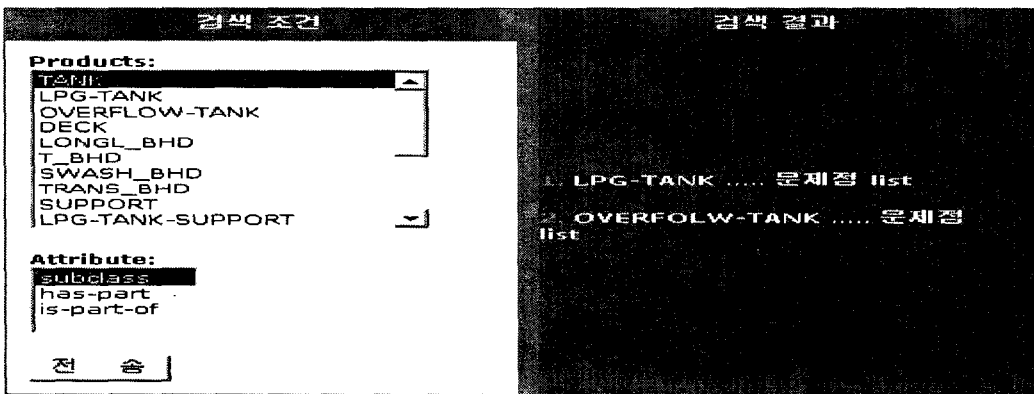
본 절에서는 앞에서 설계한 시스템의 프로토타입을 가지고 사용자와 관리자의 측면에서 수행 과정을 간단한 예제를 통해서 살펴보도록 하겠다.

(1) 사용자(설계자)

- ① 사용자는 자신의 아이디와 패스워드를 이용해 시스템에 로그인한다.
- ② 로그인 후 <그림 17>과 같은 검색조건 화면이 나타난다. 이 화면에서는 제품과 속성을 선택하여 질의를 요청하여 원하는 제품에 대한 정보를 얻을 수 있다. 이 화면에서 “Products” 에서 제품을 선택하고, “Attribute” 에서 원하는 속성을 선택한 후, “전송” 버튼을 누르면, 질의 결과가 “검색결과”란에 보여진다. 예를 들어 “Products”에서 “TANK”를 선택하고, “Attribute”에서 “subclass”를 선택한다.
- ③ <그림 17>과 같은 화면에서 질의에 대한 결과 화면은 <그림 18>와 같다. “검색결과” 부분에 “TANK의 subclass는 무엇이 있는가”에 대한 질의 요청에 대한 결과로 “1. LPG-TANK”, “2.OVERFLOW-TANK” 두 종류의 제품이 나타난다.



< 그림 17 > 검색화면



< 그림 18 > 검색결과 화면

④ <그림 18>에서 나온 검색결과에서 제품명을 선택하면, 예를 들어 여기서는 “1. LPG-TANK”를 선택하면, 우선 “LPG-TANK”의 기능을 설명하는 <그림 19>와 같은 문서를 볼 수 있다.

NO	ProductName	기능
1	LPG-TANK	LPG 가스를 저장한다.구획을 나뉘주고 화물을 보호한다. TANK의 온도를 유지시켜주며, TANK를 지지한다.구획을 나뉘주고 화물을 보호한다. TANK의 온도를 유지시켜주며, TANK를 지지한다.
2	LPG-TANK-VERTICAL-SUPPORT	하중을 선체에 전달시킨다. TANK를 지지하고, 보호한다. Access를 제공한다. TANK 거동을 구속시킨다.

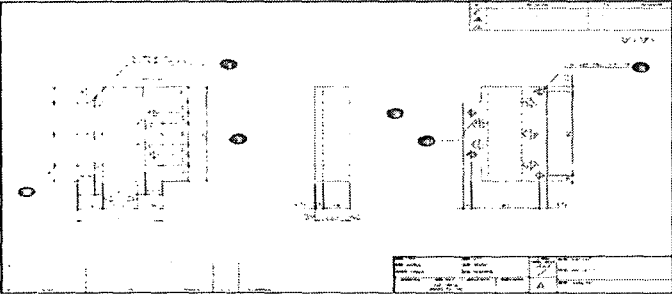
< 그림 19 > 검색결과와 “1. LPG-TANK” 선택 화면

⑤ “문제점 list”를 선택하면, 선택한 문제점에 대한 내용이 <그림 20>과 화면에 나타나게 된다.

No	구성요소	문제점	원인분석			
			설계	제작	현장	물류
1	LPG-TANK	1) TK, TOP BASE 작업시 C,L GIR에 작업용 HOLE이 없다		○		
		2) 추락위험 지역에 ACCESS HOLE이 있다. 추락위험 지역에 ACCESS HOLE이 있다.	○			
		3) SLIT 공법을 적용할 수 없다.			○	
		4) SLIT공법을 적용할 수 없다.				○

< 그림 20 > 문제점 list 선택 화면

⑥ 문제점 내용을 클릭하면 문제점에 대한 상세한 내용이 <그림 21>과 같이 볼 수 있다.



번호	사이즈	성질
1	50mm	물관라인
2	75mm	물관라인이뿔
3	175mm	중(轉)파이프
4	75mm	배수라인
5	25mm	오버플로우
6	50mm	환수라인

자속도면은 SVG파일입니다

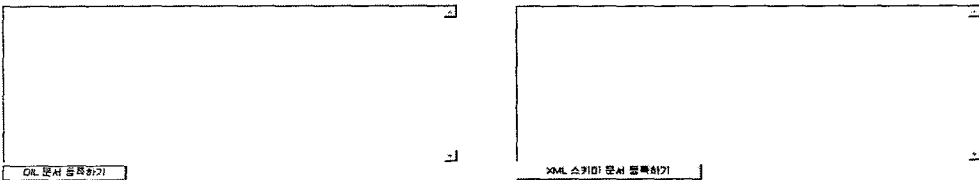
< 그림 21 > 문제점 list의 선택 후 상세 내용

(2) 관리자

- ① 관리자는 “사용자”가 이용하는 화면 역시 사용 가능하다.
- ② 관리자는 OIL을 정의하여 OIL 문서와 XML 스키마를 <그림 22>와 같은 양식을 이용해 작성하여 저장한다.

OIL 문서
OIL 문서 ID : 20011205121235

XML 스키마 문서
XML 스키마 문서 ID : 20011205121226



< 그림 22 > OIL 문서 및 XML 스키마 문서 생성화면

- ③ 관리자는 XML 스키마를 이용하여 온톨로지 기반이 문서를 생성하기 위해 <그림 23>과 같은 입력화면을 이용한다. 여기서는 여러 문서 중 하나인 “정보수집 계획서” 문서를 생성하는 화면을 예를 들어 설명하겠다.

정보 수집 계획서

항 목	2	<정보수집> <정보> <항목></항목> <정보수집내용>V/DATA 관리를 위한 분류기준 <정보자원>특) 설계 의장 3</정보자원> <담당자>황선웅</담당자> <일정>2002. 01. 01</일정> <비고> </정보> <정보> <항목>2</항목> <정보수집내용>실적 공사 V/DATA망 조사</정보수집내용> <정보자원>특) 설계 의장 3</정보자원> <담당자>심명보</담당자>
정보수집내용	실적 공사 V/DATA망 조사	
정보자원	특) 설계 의장 3	
담당자	심명보	
일정	2002. 01. 01	
비고		
입력 다시쓰기 미리보기		

< 그림 23 > 정보 수집 계획서의 입력 화면

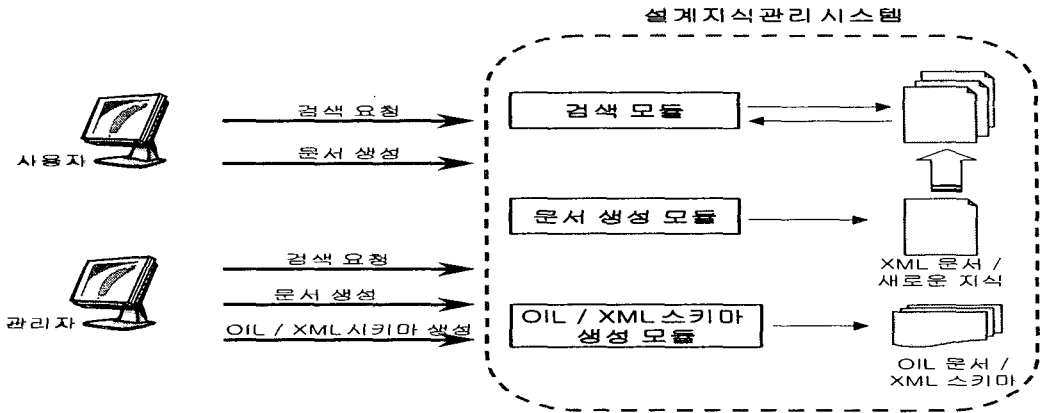
⑤ <그림 23>의 화면에서 “미리보기” 버튼을 선택하면, <그림 24>와 같이 정보수집 계획서 문서화면이 나타나고 확인 후 저장이 가능하다.

Record NO	항목	정보수집내용	정보자원	담당자	일정	비고
1	1	V/DATA 관리를 위한 분류기준	특) 설계 의장 3	황선웅	2002. 01. 01	
2	2	실적 공사 V/DATA망 조사	특) 설계 의장 3	심명보	2002. 01. 01	

저장

< 그림 24 > 정보 수집 계획서의 미리보기 화면

<그림 25>에서처럼 사용자(설계자)는 검색을 통해서 얻은 지식과 제품 설계 시 주의사항 혹은 제품조립 시 발생했던 문제점 등을 참고로 하여, <그림 23>과 같은 정형화된 입력화면을 통해서 문서를 생성할 수 있으며, 관리자는 정형화된 입력화면을 통해서 문서를 생성할 수 있고, 또한 설계자에게서 얻은 정보나 각 부서에서 보내 준 정보를 바탕으로 새로운 형태의 지식을 문서로 생성할 수 있다. 이러한 설계 문서들은 설계 도면이나 그림 정보들은 생성된 XML 문서를 브라우저에서 볼 수 있도록 HTML로 변환하는 XSLT(XML Stylesheet Language Transformation)를 정의함으로써 그림정보를 저장할 수 있고, 앞의 <그림 21>과 같이 SVG(Scalable Vector Graphics)를 이용해서 그림정보를 관리자가 생성한다.



< 그림 25 > 사용자와 관리자의 수행흐름도

7. 시스템 테스트 결과

기존의 키워드 검색에 의해 검색된 결과에서는 사용자가 원하는 정보를 얻기 위해 여러 번 검색해야 하는 번거로움이 있지만, 이 시스템을 적용할 경우 XML 문서의 논리구조를 이용하여 검색하게 되므로 원하는 결과를 쉽게 얻어 낼 수 있다. 또한, 검색된 결과와 관련되는 내용도 부가적으로 검색 할 수 있다. 앞에서 설명한 시스템의 수행 예에서처럼 "TANK"의 subclass에 대한 검색을 통해서 "LPG-TANK와 OVERFLOW-TANK"의 결과를 얻을 수 있고, 제품에 대한 문제점 list, 기능, 구성요소 등에 대한 내용을 볼 수 있다. 또한 제품 구성요소의 구성요소에 대한 검색과 문제점 list, 기능, 구성요소 등도 확인이 가능하다.

기존의 OIL은 앞에서 설명했듯이 사용자에게 매우 직관적이며, 잘 정의된 형식의 의미를 가지고 있고, 시스템간의 상호 운영성을 확보하기 위해 XML과 RDF 같은 기존 웹 언어와 적절한 연계가 가능하다. 하지만, 이러한 OIL도 몇 가지 단점을 가지고 있다. 우선, OIL은 비록 superclass로부터 값을 상속받는 메커니즘을 제공하지만, 그 값을 갱신할 수 없다. 그리고 OIL에서 정의하는 slot은 처음에 OIL 문서 생성 시에 정의되는 slot만 사용할 수 있다. OIL의 이러한 단점은 시스템의 변경이나 slot의 변경 시에 많은 작업이 필요하기 때문에 XML을 이용하는 확장성 및 재사용성에 있어서의 문제점이 발생한다.

8. 결론 및 추후 연구과제

본 연구에서는 설계지식에 온톨로지 개념을 도입하여 명확하게 개념과 그 관계들을 정의함으로써 기존의 문서나 정보들을 체계화하는 방법을 제시하였다. 여러 온톨로지 표현언어 중 semantic web 실현을 위해, 웹에서 정보자원을 표현하기 위한 기존의 온톨로지 언어보다 더욱 발전된 형태의 온톨로지 언어가 갖추어야 할 조건을 만족시

키는 OIL을 이용하여 용어와 관계를 정의하였다. 이를 바탕으로 XML 스키마를 생성함으로써 설계지식에 대한 내용정보를 XML 이용하여 구조화하여 데이터베이스에 저장하는 방법을 살펴보았다. 이러한 방법을 통해서 설계지식을 일관성 있게 유지·관리할 수 있으며, 이를 활용하여 새로운 설계지식 생성 시 활용할 수 있다. 설계 지식을 XML 문서화하여 관리함으로써 논리구조를 이용한 검색이 가능하기 때문에 검색결과가 기존의 키워드 검색과 다르게 원하는 결과를 쉽게 얻어 낼 수 있다. 또한 상관관계에 따른 검색 즉, "subclass" 혹은 "is-part-of"들의 속성을 통해서 제품과 그 구성요소들에 대한 검색이 가능하고, 이들 검색결과를 통해서 각종 정보를 확인할 수 있다. 향후, 앞에서 지적한 OIL의 문제점을 해결하기 위해 OIL 정의의 범용성 방안 연구가 필요하다.

9. 참 고 문 헌

- [1] 김민수, 정성원, 정준원, 신수정, 김정미, SGML, XML, EDI 통합 및 연계방안, 한국전산원, 1999.
- [2] Dieter Fensel, Frank van Harmelen, Ian Horrocks, Deborah L. McGuinness, and Peter F. Patel-Schneider "OIL: An Ontology Infrastructure for Semantic Web", IEEE Intelligent Systems, Vol 16, No 2, pp38-45, 2001.
- [3] Ian Horrocks, Dieter Fensel, J. Broekstra, Stefan Decker, Michael Erdmann, C. Goble, Frank van Harmelen, M. Klein, S. Staab, Studer, and E. Motta, "The Ontology Inference Layer OIL", On-To-Knowledge, <http://ontology.org>, 2000.
- [4] Sean Bechhofer, Jeen Broekstra, Stefan Decker, Michael Erdmann, Dieter Fensel, Carole Goble, Frank van Harmelen, Ian Horrocks, Michel Klein, Deborah McGuinness, Enrico Motta, Peter Patel-Schneider, Steffen Staab, and Rudi Studer, "An informal description of Standard OIL and Instance OIL", On-To-Knowledge, 2000.

저 자 소 개

한 영 근 : 서울대학교 기계설계학과 학사, 석사, 펜실베니아 주립대 산업 및 생산공학과 박사, 현 명지대학교 산업시스템공학부 교수, 공학박사
관심분야는 생산자동화, 가상생산, 협업엔지니어링 등이다.

조 진 형 : 명지대학교 산업공학과, 학사, 석사, 현재 한국하우톤 근무
관심분야는 생산관리, 정보시스템 등이다.