韓國컴퓨터情報學會 論文誌
第15卷 第1號, 2010. 1.

2010-15-1-1-4

OWL DL을 사용한 GPM 핵심 모델의 구현

최 지 웅*, 박 호 병*, 김 형 진**, 김명호***

Implementation of GPM Core Model Using OWL DL

Ji Woong Choi*, Hobyung Park*, Hyung-Jean Kim**, Myung Ho Kim***

요 약

GPM(Generic Product Model)은 원자력 플랜트의 라이프 사이클 데이터를 통합, 공유하기 위하여 일본의 Hitachi에서 개발한 공통 데이터 모델이다. GPM은 추상 모델 성격의 GPM 핵심 모델과 핵심 모델의 기술을 위한 구현 언어 그리고 구현 언어로 작성된 참조 라이브러리로 구성되어 있다. GPM 핵심 모델은 객체들 사이의 의미가 부여된 관계 정의를 통하여 객체 의미 관계 네트워크 모델을 구성할 수 있는 특성이 있다. 초기의 GPM은 GPM 핵심 모델의 특성을 반영한 신택스의 GPML이라는 구현 언어를 개발하여 제공하였으나 원자력 플랜트 라이프 사이클 동안 다양한 목적으로 GPM 데이터 모델에 접근하는 이기종 애플리케이션들과의 상호운용성을 위하여 XML을 기반으로 하는 GPM-XML로 교체되었다. 그러나 현재 GPM-XML을 사용하여 구축한 GPM 데이터 모델이 객체 의미 관계 네트워크 모델로서 활용되기 위한 GPM-XML 기반의 연구가 미비한 상태이다. 따라서 본 논문에서는 객체 의미 관계 네트워크와 유사한 성격의 온톨로지를 기술할 수 있으며 이를 지원하는 기술 표준 및 도구들이 이미 활용 가능한 수준에 있는 OWL을 GPM 핵심 모델을 위한 구현 언어로서 제안한다. OWL은 XML 기반의 RDF/XML 형식으로 기술될 수 있으므로 상호운용성 또한 보장받을 수 있다. 본 논문은 OWL의 세 가지 하위 언어 사양 중 추론 기능을 완벽히 제공받을 수 있는 가운데 문법적 제약이 가장 덜 엄격한 OWL DL을 사용한다. 본 논문은 OWL DL을 GPM 핵심 모델의 구현 언어로서 사용하기 위하여 GPM과 OWL 두 모델 사이의 차이점을 도출한 후 이를 해소할 수 있는 방법을 제안하며 이 방법을 적용하여 GPML로 작성된 참조 라이브러리를 OWL DL 기반의 온톨로지로 변환하여 구축하는 방법을 기술한다.

Abstract

GPM(Generic Product Model) developed by Hitachi in Japan is a common data model to integrate and share life cycle data of nuclear power plants. GPM consists of GPM core model, an abstract model, implementation language for the model and reference library written in the language. GPM core model has a feature that it can construct a semantic network model consisting of relationships among objects. Initial GPM developed and provided GPML as an implementation language to support the feature of the core model, but afterwards the GPML was replaced by GPM-XML based on XML to achieve data interoperability with heterogeneous applications accessing a GPM data model. However, data models written in GPM-XML are insufficient to be used as a semantic network model for lack of studies which support GPM-XML and enable the models to be

• 제1저자 : 최지웅 교신저자 : 김명호
• 투고일 : 2009. 11. 05, 심사일 : 2009. 11. 20, 게재확정일 : 2010. 01. 26.
* 숭실대학교 IT대학 컴퓨터학과 학생 ** 고등기술연구권 연구원 *** 숭실대학교 IT대학 컴퓨터학부 교수

used as a semantic network model. This paper proposes OWL as the implementation language for GPM core model because OWL can describe ontologies similar to semantic network models and has an abundant supply of technical standards and supporting tools. Also, OWL which can be expressed in terms of RDF/XML based on XML guarantees data interoperability. This paper uses OWL DL, one of three sublanguages of OWL, because it can guarantee complete reasoning and the maximum expressiveness at the same time. The contents of this paper introduce the way how to overcome the difference between GPM and OWL DL, and, base on this way, describe how to convert the reference library written in GPML into ontologies based on OWL DL written in RDF/XML.

▸ Keyword : GPM(Generic Product Model), OWL(Web Ontology Language), 원자력 발전소(Nuclear Power Plant)

Ⅰ. 서 론

원자력 플랜트는 설계, 시공, 운영 및 유지보수, 해체의 라이프 사이클 동안 다양한 조직들이 참여하여 유/무형의 많은 자원이 소요되는 대형 프로젝트이다. 라이프 사이클의 각 단계는 다음 단계로 넘어갈 때 대량의 정보 이양(hand-over) 절차가 동반된다. 이양 데이터의 종류는 종이 문서는 물론 CAD, EXCEL 시트와 같은 전자 파일 형식도 포함된다. 이 때, CAD 파일의 경우 이기종 CAD 소프트웨어 사이의 상호 운용성이 보장되지 않기 때문에 정보 교환을 위한 많은 시간과 비용이 소비된다. 이러한 문제를 해결하기 위해 CAD 파일간의 상호 교환을 목적으로 하는 국제 표준으로서 ISO 10303(STEP: Standard for The Exchange of Product model data)이 있다. 그러나 STEP은 설계 단계에서 발생하는 CAD 데이터의 교환에 초점이 맞추어져 있기 때문에 이후 단계에서 발생하는 데이터까지 일관된 형식으로 관리 및 공유하기 위한 공통 데이터 모델로 사용하기에는 한계가 있다. 다른 국제표준으로서 오일 및 가스 시설을 포함하는 프로세스 플랜트를 대상으로 라이프 사이클 데이터를 일관되게 통합할 목적의 ISO 15926 규격이 있으나 이 또한 초기에 원자력 플랜트를 대상으로 하지 않았기 때문에 원자력 플랜트의 특성을 반영하지 못하는 한계가 있다[1]. 따라서, 일본에서는 IMS(Intelligent Manufacturing Systems) 프로그램을 통해 수행한 VIPNET(Virtual Production Enterprise Network) 프로젝트[2]를 통해 ISO 10303의 AP(Application Protocol) 221[3], 227[4] 및 ISO 15926[5,6] 등을 참조해 의미 네트워크(Semantic Network)[7,8] 기반의 데이터 모델인 GPM(Generic Product Model)[9]을 개발하였다. GPM은 원자력 플랜트의 라이프 사이클 데이터를 수용하여 일관되게 통합 및 공유할 수 있는 공통 데이터 모델이다. GPM은 GPM 핵심 모델(GPM Core Model)과 이의 구현을 위한 언어로서 GPML을 제공하였으며 차후에 XML 기반의 GPM-XML을 제공하였으며 두 언어별로 구현된 참조 라이브러리로 구성되어 있다.

GPM을 활용하기 위해서는 GPM을 사용하여 원자력 플랜트를 위한 공통 데이터 모델을 정의한 후 이를 운용할 공유 저장소 시스템이 필요하다. 이 저장소의 목적은 원자력 플랜트 라이프 사이클 동안 참여하는 다양한 조직들과 그들이 사용하는 다양한 애플리케이션이 발생시키고 필요로 하는 데이터를 통합된 형식으로 저장 및 제공함으로써 플랜트 전체 프로세스의 효율성을 확보하고 경쟁력을 향상시키고자 하는데 있다. 따라서 이러한 목적에 부합하는 저장소를 구축하는데 있어서 고려되어야 할 사항들 중 원자력 플랜트 라이프 사이클 데이터가 저장소 내부에 저장되는 형식이 GPM의 특성을 얼마나 잘 반영할 수 있는지와 저장소가 외부의 이기종 애플리케이션들과 정보 교환 시 얼마나 효율적인가 하는 문제가 있다. GPM은 초기에 GPM 모델의 특성 즉, 객체 사이의 의미 관계 네트워크 모델 구축이 가능하다는 점을 반영하는 독자적인 모델 기술 언어로서 GPML과 이를 바탕으로 구현한 참조 라이브러리를 개발하여 제공하였으나 이후 XML을 모델 구현 언어로서 사용하기 위하여 XML 태그명과 그 나열 규칙을 정의한 GPM-XML이라는 XML 스키마와 이를 준수하여 작성한 XML 형식의 참조 라이브러리를 제공하였다. 이러한 선택은 XML이 웹 기반의 문서 그리고 정보 교환을 위한 표준 형식으로서 그 응용 범위 또한 광범위하여 GPML과 비교하여 외부의 이기종 애플리케이션들과의 데이터 교환이 이루어질 때 상호 운용성 측면에서 우수하기 때문이다. 그러나 GPML은 GPM-XML에 비하여 객체간의 의미 관계를 보다 직관적으로 기술(description) 및 파악할 수 있는 구문 체계를 가지고 있다.

본 논문에서는 객체 사이의 의미 네트워크 구성이 가능한 GPM 핵심 모델의 특성을 반영하는 GPML의 장점과 저장소 외부의 이기종 애플리케이션들과의 상호 운영성 측면에서 우수한 XML의 장점을 모두 취할 수 있도록 GPM 핵심 모델 기술 언어로서 OWL[10]을 제안한다. OWL은 온톨로지[11,12]를 구축할 수 있는 W3C에서 제정한 표준 웹 온톨로지 언어이다. OWL이 GPM 핵심 모델의 특성을 반영할 수 있는 근거는

OWL을 사용해 구축한 데이터 모델 즉, 온톨로지의 구성 요소와 구조가 GPM 핵심 모델을 기반으로 구축한 객체 의미 관계 네트워크 모델과 유사한 특성을 가지고 있기 때문이다. 또한 OWL이 XML의 장점을 취할 수 있는 이유는 OWL은 RDF/XML[13] 형식의 파일로 표현될 수 있는데 RDF/XML은 XML 형식으로 OWL 온톨로지를 출력하기 위하여 정의한 XML 스키마 체계로서 XML의 응용이기 때문에 XML의 장점 또한 그대로 계승할 수 있기 때문이다. GPM 핵심 모델을 OWL을 사용하여 구현할 때 얻을 수 있는 추가적인 장점은 온톨로지의 장점이다. 즉, 온톨로지에 기술되어 있는 객체 관계 정의 사이의 의미적 충돌, 오류 등을 검출해 낼 수 있음은 물론 온톨로지에 명시적으로 기술된 내용을 기반으로 객체간의 숨겨진 더 많은 관계를 추론 기능을 통해 유추해 낼 수 있다. 또한 같은 맥락으로 동일 내용에 대한 중복 기술을 제거함으로써 모델 기술을 효율적으로 수행할 수 있다는 점이다. 예를 들면, 클래스 A가 클래스 B의 자식 클래스라면 이 내용을 GPML과 GPM-XML로 기술할 경우 클래스 A의 정의부에 클래스 B가 부모 클래스임을 명시하고 다시 클래스 B의 정의부에 클래스 A가 자식 클래스임을 명시해야 하지만 OWL의 경우 두 클래스 중 한 곳에서만 정의하면 된다. 마지막으로 OWL로 구축한 데이터 모델을 대상으로 하는 표준 질의 언어 또한 제공받을 수 있기 때문에 이를 기반으로 하는 공유 데이터 저장소의 구현을 용이하게 한다.

OWL은 OWL Lite, OWL DL, OWL Full의 세 가지 하위 언어 사양을 제공한다. OWL Lite와 OWL DL은 기술 논리(Description Logic)[14]를 기반으로 설계된 언어이며 OWL Lite는 OWL DL이 지원하는 서브 셋 언어 사양을 제공하므로 OWL Lite 규격에 맞춰 작성한 문서는 OWL DL 문서로 간주된다. OWL Full은 객체간의 관계 기술에 있어서 보다 다양한 표현 방법을 획득하기 위하여 기술 논리의 제약 사항을 파괴하였다. 따라서 OWL Full은 기술 논리에 기반한 추론 기능을 제공받을 수 없다는 단점이 있다[10]. GPM 핵심 모델 특성을 반영함과 동시에 구체적인 원자력 플랜트 데이터들을 위한 공유 데이터 모델을 기술하는 단계만을 고려했을 때는 OWL의 언어 사양 중 표현력에 있어서 가장 강력한 OWL Full이 모델 기술 언어로서 적합할 수 있으나 원자력 플랜트를 위한 공유 데이터 저장소의 구현 및 운영 단계까지 고려할 경우 저장소 내부의 온톨로지에 포함될 방대한 수의 객체들에 대한 수정, 삭제, 추가 등의 이벤트에 대한 관리를 DL 기반의 추론 기능을 활용할 경우 해당 이벤트 관련 객체들은 연쇄적으로 새로운 관계를 형성하며 그들 간의 관계에 대한 정합성 또한 유지 및 관리되기 때문에 본 논문에서는 GPM 핵심 모델을 위한 구현 언어로서 OWL DL을 선택하였다.

지금까지 OWL DL을 GPM 핵심 모델을 위한 구현 언어로서 뿐 아니라 원자력 플랜트 라이프 사이클 데이터를 위한 공유 저장소 내부의 데이터 모델로 선택하여 사용하고자 할 때 누릴 수 있는 장점들에 대하여 기술하였다. OWL DL을 기반으로 하여 GPM 핵심 모델을 구현하고자 할 때 OWL DL과 GPM 핵심 모델과의 두 모델 사이에 존재하는 객체 관계 기술에 있어서 존재하는 제약 사항들의 차이를 극복해야 하는 문제가 남는다. 따라서 본 논문에서는 두 모델 사이의 객체 관계 기술에 있어서의 차이점을 도출한 후 그 문제를 해결하기 위하여 사용한 방법을 기술하며 그 방법을 사용하여 구현한 변환기 소프트웨어를 통해 GPML로 작성된 참조 라이브러리를 OWL DL 제약 사항을 만족하는 OWL DL 온톨로지로 변환시킨 내용을 기술한다.

본 논문의 구성은 다음과 같다. 제2장에서는 본 논문의 모델 변환 대상인 GPM과 OWL에 대하여 각각 객체들의 관계를 기술하는 방법에 초점을 맞추어 기술한다. 제3장에서는 두 모델을 비교하여 변환 시 극복해야 할 차이점을 도출한 후 본 논문에서 이를 해결하기 위하여 제안하는 방법을 기술한다. 제4장에서는 이 변환 방법을 기반으로 하여 GPML로 기술한 참조 클래스 라이브러리와 참조 어소시에이션(association) 라이브러리를 OWL-DL 기반의 온톨로지로 변환하는 상세 내용을 기술한다. 마지막으로 제5장은 결론에 해당한다.

II. 관련 연구

이 장에서는 본 논문의 모델 변환 대상인 GPM과 OWL에 대하여 기술한다.

2.1 GPM

GPM은 원자력 플랜트의 라이프 사이클(설계, 운영, 유지보수)동안 발생하는 유/무형의 데이터를 정의하고 통합하기 위하여 일본의 Hitachi에서 개발한 데이터 모델이다. GPM은 PlantCALS, PlantEC, IMS VIPNET 프로젝트의 산출물이다. GPM은 ISO10303 STEP의 AP221, AP227, AP231과 ISO15926 Process Plants를 기반으로 하여 개발되었다. GPM은 범용 제품(product) 데이터를 대상으로 하여 CAD 데이터의 교환 목적으로 개발한 공통 데이터 모델 표준인 ISO10303 STEP과 해상 플랜트를 대상으로 라이프 사이클 동안 발생하는 데이터를 위한 중립 데이터 모델 표준인 ISO15926을 기반으로 하여 원자력 플랜트가 갖는 추가적인 특성을 반영 및 개선한 데이

터 모델이다.

GPM의 구성은 GPM의 논리적 근간인 GPM 핵심 모델(GPM Core Model)과 이를 기술할 수 있는 언어로서 GPML과 GPM-XML이 있다. 이 두 가지 기술 언어별로 각각 코어 모델을 구현한 참조 모델 성격의 클래스 라이브러리와 어소시에이션 라이브러리가 있다.

2.1.1 GPM 핵심 모델(GPM Core Model)

그림 1. GPM에서의 가장 간단한 문장 형태
Fig. 1. The simplest form for a sentence in GPM

GPM 코어 모델의 기본 요소는 객체(object)와 어소시에이션(association)이다. 객체는 클래스 또는 클래스를 구상화한 인스턴스(instance)를 포괄하는 개념이며 어소시에이션은 객체들을 연결하는 기능을 수행한다. 그림 1은 GPM의 최소 의미 단위를 보여주며 자연어의 문장 구조를 모방한다. GPM은 주어:서술어:목적어의 대응수(mapping cardinality)가 1:1:n이다. 이때, n>=1 이며 n은 어소시에이션에 의해서 결정된다. 임의의 object-association-object(s) 형태의 문장 구조에서 주격 객체는 다른 문장의 목적어가 될 수 있으며 목적격 객체(들)는 다른 문장의 주어 객체가 될 수 있다. 이러한 구조 확장 방식을 통해 GPM을 사용하여 특정 도메인을 위한 객체 관계 네트워크 모델을 구성할 수 있다.

2.1.2 GPM 참조 라이브러리

GPM 핵심 모델은 참조 라이브러리로 구현되어 있으며 참조 라이브러리는 클래스 라이브러리와 어소시에이션 라이브러리로 구성되어 있다. GPM 클래스 라이브러리는 2442개의 원자력 플랜트를 모델링하는데 필요한 형상/비형상 객체들에 대응하는 클래스를 정의하고 있다. GPM 클래스 라이브러리에 정의되어 있는 클래스는 각각의 의미 혹은 기능적 공통점에 근거해서 30개의 장르(genre)로 구분되어 있다. 장르는 클래스들의 단순한 그룹 이름일 뿐이며 GPM의 기술(description) 언어인 GPML에서도 이를 위한 키워드성격의 어휘는 존재하지 않는다. 어소시에이션 라이브러리는 19개의 어소시에이션을 정의하고 있다. 어소시에이션은 객체 사이의 연결 관계를 기술한다.

표 1은 GPM 어소시에이션 라이브러리에서 정의한 어소시에이션 목록과 각각의 어소시에이션에 대한 목적격 객체들의 출현규칙에 대한 명세를 보여준다. GPM은 목적격 객체들을 통칭하고자 어트리뷰트(attribute)라는 용어를 제공한다. GPM은 어소시에이션을 사용한 객체들 간의 관계 정의에서 복수의 목적격 객체들이 존재하는 것을 허용하기 때문에 이들을 구분하기 위하여 객체들이 롤(role)이라는 요소를 부여 받고 연결에 참여하도록 정의하고 있다. GPM의 어트리뷰트를 구성하는 롤의 종류와 롤의 출현 순서 그리고 동일한 롤을 담당하는 객체들의 개수 범위는 어소시에이션에 종속적으로 정의되어 있다.

표 1. GPM 어소시에이션 목록
Table 1. GPM association list

association	order	occurs	role
is_connected_to	1	+	side2
	2	*	via
involves	1	1	involved
	2	*	part
is_referenced_by	1	1	referring
	2	?	variable
boolean_operate	1	+	first_operand
	2	1	boolean_operator
	3	*	second_operand
is_placed_on	1	1	place
	2	1	by
	3	?	of
has_property_of	1	+	qualifier
	2	1	property
	3	*	data
	4	?	unit
possesses	1	+	possessed
is_represented_by			representer
is_a_version_of			predecessor
is_qualified_as			qualifier
is_presented_by			presenter
is_identified_with			identifier
is_described_as			describing
is_collection_of			part
is_assigned_to			assigning
is_assembled_from			part
is_an_instance_of			class
is_a_synonym_of			alternative
is_classified_as			classifier
			+: more than 1 *: more than 0 ?: 0 or 1

그림 2는 어소시에이션을 사용해 객체들 사이의 관계를 정의한 예이다. unit_segment 객체는 주어 역할을 담당하며 maximum, operation, pressure, data_value 객체는 어트리뷰트를 구성하는 멤버 객체이다. 목적격 객체들은 어소시에이션 'has_property_of'

명세에 의하여 qualifier, property, data, unit의 네 가지 롤 중 하나를 담당한다. 이 연결에서 unit 롤을 위한 객체는 정의되어있지 않다. unit 롤에 해당하는 객체의 출현회수가 0 혹은 1이므로 생략 가능하다.

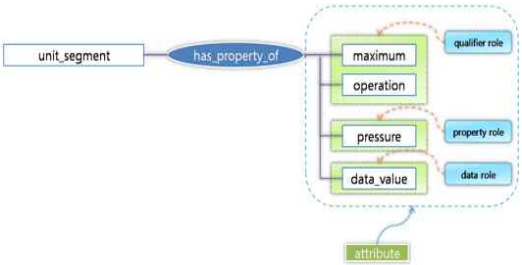


그림 2. has_property_of 어소시에이션을 사용한 객체들 사이의 관계 정의
Fig. 2. Relationship among objects using association has_property_of

2.1.3 GPML

GPM의 클래스 라이브러리와 어소시에이션 라이브러리는 GPML을 사용하여 구현되어 있다. GPML은 클래스를 다음과 같은 구문에 따라 정의한다.

```
CLASS class_name {
  ...
  association: attribute;
  ...
}
```

```
CLASS 5w1h {
 has_property_of: unqualified date_and_time data_value;
 has_property_of: unqualified place data_value;
 has_property_of: unqualified version_number data_value;
 is_assigned_to: person;
 is_assigned_to: organization;
 is_assigned_to: person_and_organization;
 is_presented_by: information_contents;
}
```

그림 3. 클래스 정의를 위한 GPML 구문 및 예
Fig 3. GPML syntax to define a class and the example

키워드 CLASS는 클래스 정의의 시작을 의미한다. class_name은 정의되는 클래스 이름이며 중괄호로 둘러싸인 클래스 정의부에서 사용된 모든 어소시에이션의 주격 클래스가 된다. 클래스 정의부는 association: attribute; 단위의 문장을 사용하여 다른 클래스들과의 연관 관계를 기술한다. 어트리뷰트에 포함되는 클래스들은 목적격 클래스이며 어트리뷰트의 형식은 어소시에이션에 종속적이다. 어트리뷰트에서 동일 롤에 해당하는 클래스가 복수로 나열될 경우 ‘,’로 구분하며 다른 롤과의 구분은 공백을 기준으로 한다. 어소시에이션에 따른 롤의 출현 순서는 표 1의 order를 따른다. 그림 2를 GPML로 표현할 경우 unit_segment has_property_of: maximum, operation pressure data_value; 이다.

2.2 OWL

OWL은 2004년 W3C에서 발표한 표준 웹 온톨로지 기술(description) 언어이다. OWL은 OWL Lite, OWL DL, OWL Full의 세 가지 하위 사양을 제공한다. OWL Lite와 OWL DL은 기술 논리(description Logic)를 기반으로 하였기 때문에 이 두 사양으로 구축한 온톨로지는 추론기(reasoner)를 통하여 명시적으로 기술된 관계보다 더 많은 객체들 간의 숨겨진 관계를 찾아낼 수 있다. 또한 OWL DL은 OWL Lite 사양을 완전히 포함한다. 즉, OWL Lite로 구축한 온톨로지는 OWL DL 온톨로지가 된다. OWL Full은 OWL Lite와 OWL DL에 비하여 개념들 간의 관계 기술에 있어서 보다 풍부한 표현을 허용하는 반면 이를 통해 구축한 온톨로지를 기반으로 한 완벽한 추론은 보장하지 않는 한계가 있다. 본 논문은 GPM을 OWL-DL로 변환하는 것을 목적으로 하기 때문에 이 절에서는 OWL-DL을 기준으로 OWL을 기술한다.

2.2.1 OWL의 구성 요소

OWL의 구성 요소는 클래스, 개체(individual), 프로퍼티(property)로 구성된다. OWL의 클래스는 개체들의 집합(set)이고 클래스의 정의는 자신들의 원소인 개체가 만족시켜야 하는 조건을 기술하는 것을 의미한다. 이 조건의 내용은 해당 클래스의 개체가 특정한 다른 클래스의 개체와 특정 프로퍼티로 관계를 맺는 상황에서 준수해야 하는 제약 사항에 해당한다.

2.2.2 OWL 클래스 정의

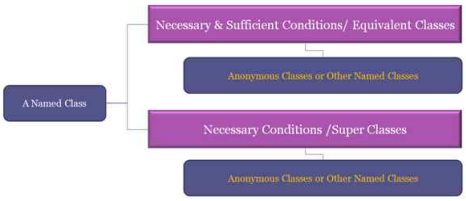


그림 4. 명명된 클래스와 익명 클래스의 관계
Fig. 4. Relationship between a named class and anonymous classes

OWL 클래스는 명명된(named) 클래스와 익명(anonymous) 클래스로 분류할 수 있다. 그림 4는 두 클래스의 관계를 보여준다. 익명 클래스의 정의는 자신의 멤버 개체에게 요구되는 조건을 제시하는 방법으로 이루어진다. 명명된 클래스의 정의는 다른 익명 혹은 명명된 클래스들을 자신의 동치 클래스 혹은 수퍼클래스로 설정함으로써 이루어진다. 동치 클래스로 편입

된 클래스들에 정의된 조건들은 정의하고자 하는 명명된 클래스를 위한 필요충분조건으로 편입되며 수퍼클래스로 편입된 클래스들에 정의된 조건들은 정의하고자 하는 명명된 클래스를 위한 필요조건으로 편입된다.

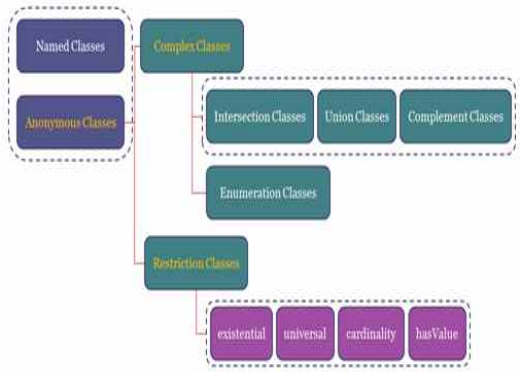


그림 5. 익명 클래스의 종류
Fig. 5. Kinds of anonymous classes

표 2. OWL 익명 클래스를 위한 생성자 목록
Table 2. Constructors for OWL anonymous classes

OWL Constructor	DL Syntax	OWL Constructor	DL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	allValuesFrom	$\forall P C$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	minCardinality	$\geq N P$
complementOf	$\neg C$	maxCardinality	$\leq N P$
oneOf	$\{x_1 \ldots x_n\}$	cardinality	$= N P$
someValuesFrom	$\exists P C$	hasValue	$\exists P \{x\}$
C, C_1, C_2: class P : property x, x_1, x_n: individual N : number			

그림 5는 명명된 클래스 정의의 기반이 되는 익명 클래스의 종류를 보여주며 표 2는 익명 클래스 종류에 따른 OWL의 클래스 생성자를 보여준다. 익명 클래스의 기본이 되는 클래스는 한정(restriction) 클래스이다. 존재 한정(existential restriction)은 정의하고자 하는 클래스의 임의의 개체가 클래스 C의 개체와 프로퍼티 P를 사용한 최소한 하나의 연결이 존재해야 한다는 조건을 의미한다. 전칭 한정(universal restriction)은 정의하고자 하는 클래스의 임의의 개체가 프로퍼티 P를 사용할 때는 반드시 클래스 C의 개체와만 관계를 맺어야 한다는 조건을 의미한다. 관계차수 한정(cardinality restriction)은 정의하고자 하는 클래스의 임의의 개체가 프로퍼티 P를 사용한 관계의 개수에 대한 제한이다. hasValue 한정은 정의하고자 하는 클래스의 임의의 개체가 특정 개체 a와 프로퍼티 P를 사용한 최소한 하나의 연결이 존재해야 한다는 조건을 의미한다. 열거형 클래스는 멤버 개체를 명시적으로 나열한 것이다. 교집합, 합집합, 여집합 클래스는 이미 정의된 클래스들을 기반으로 집합 연산을 적용한 클래스들이다.

III. 본론

3.1 GPM의 OWL로의 변환 방법

이 장에서는 GPM을 OWL로 변환하기 위해 변환 대상 모델의 구성 요소 사이의 사상 관계를 기술하며 객체간의 연결 관계를 기술하는 방식에 있어서 존재하는 두 모델 사이의 차이점을 극복하기 위하여 사용한 방법을 설명한다.

3.1.1 GPM의 OWL 변환을 위한 모델 구성 요소 매핑

GPM과 OWL 모두 객체간의 관계 네트워크 모델을 구성할 수 있다. 따라서 두 모델은 용어의 차이는 존재하지만 각 모델에서 동일한 기능을 수행하는 세 가지 모델 구성 요소를 가지고 있다. 객체의 원형 역할을 수행하는 요소는 두 모델에서 모두 클래스라는 동일한 용어를 사용하고 있다. 클래스를 구상화한 요소는 GPM은 객체(object), OWL은 개체(individual)라는 용어를 사용한다. 객체간의 관계 정의에 사용하는 요소는 GPM은 어소시에이션, OWL은 오브젝트 프로퍼티(object property)라는 용어를 사용한다. 따라서 GPM 클래스 라이브러리와 어소시에이션 라이브러리를 OWL-DL 온톨로지로 변환하기 위하여 모델의 구성 요소를 다음과 같이 변환한다.

- GPM의 클래스는 OWL의 클래스로 변환한다.
- GPM의 객체는 OWL의 개체로 변환한다.
- GPM의 어소시에이션은 OWL의 오브젝트 프로퍼티로 변환한다.

GPM 클래스 라이브러리에서 정의한 클래스들은 동일한 이름의 OWL 명명된(named) 클래스로 변환한다. GPM 어소시에이션 라이브러리에서 정의한 어소시에이션은 동일한 이름의 OWL 오브젝트 프로퍼티로 변환한다.

3.1.2 GPM의 OWL로의 객체 관계 기술 변환 방법

GPM을 OWL로 변환할 때 두 모델 사이의 객체 관계 정의에 있어서 극복해야 할 가장 큰 차이점은 주어:술어:목적어의 대응수다. GPM은 이들의 대응수가 1:1:n(n>=1)이 가능한 반면 OWL은 1:1:1이라는 점이다. 즉, OWL은 한 번에 하나의 술

어를 사용하여 n개의 목적격 객체와의 관계 정의를 동시에 할 수 없다. 따라서 본 논문에서는 이러한 두 모델 사이의 차이점을 극복하고 의미 손실 없는 모델 변환을 위하여 우선 주어:술어:목적어의 대응수가 1:1:n인 GPM 문장을 OWL로의 변환이 용이한 주어:술어:목적어의 대응수가 1:1:1인 다수의 문장들로 분해한 후 이들을 모두 하나의 연결된 형태로 재구성한다. 재구성의 이유은 분해 전 하나의 GPM 문장 내에서 1개의 주격 객체가 n개의 목적격들과 연결 정의가 존재한다는 의미단위를 보존하기 위함이다. 이러한 절차를 통해 객체 관계 기술에 있어서의 두 모델 사이의 구조적 차이점을 해소한 후 모델 구성 요소 간 변환을 3.1절의 내용에 따라 적용한 후 OWL 구문(syntax)을 사용하여 재구성된 GPM 문장을 OWL로 표현함으로써 변환을 완료한다.

주어:술어:목적어의 대응수가 1:1:n인 GPM 문장을 주어:술어:목적어의 대응수가 1:1:1인 다수의 문장들로 분해하는 방법은 GPM 어소시에이션 라이브러리에 정의되어있는 어소시에이션들을 자신의 어트리뷰트에 출현할 수 있는 롤을 2개 이상 갖는 어소시에이션 그룹(그룹 I)과 1개만을 갖는 어소시에이션 그룹(그룹 II)으로 나누어 그 변환방법을 달리한다.

3.1.2.1 그룹 I을 위한 GPM 문장의 변환

자신의 어트리뷰트에 출현할 수 있는 롤을 2개 이상 갖는 어소시에이션을 술어로 하는 대응수 1:1:n(n>=1)인 GPM 문장의 변환은 다음과 같은 방법을 사용한다.

1) 대응수 1:1:n인 GPM 문장에서 n에 해당하는 목적격 객체들, 즉 어트리뷰트를 하나의 익명 클래스로 치환한다.
2) 익명 클래스는 어트리뷰트의 n개의 멤버 객체를 각각 목적어로 하는 n개의 문장으로 구성되며 각각의 문장에 대한 술어는 각각의 멤버 객체가 담당하고 있었던 롤이 어소시에이션으로 변환되어 사용된다.
3) GPM의 OWL 변환을 위한 모델 구성 요소 매핑 정의에 따라 GPM 문장을 구성하는 주어, 목적어 위치의 클래스들을 OWL의 클래스로 변환하고 술어 위치의 어소시에이션들을 OWL의 오브젝트 프로퍼티로 변환한다.

이와 같은 방법으로 대응수가 1:1:n인 한 개의 GPM 문장은 대응수 1:1:1인 n+1개의 GPM 문장이 하나로 연결된 구조로 변형된다. n+1개의 문장은 대응수가 1:1:n인 GPM 문장의 주어, 술어를 그대로 주어, 술어로 취하고 새로 생성된 익명 클래스 객체를 목적어로 하는 문장 1개(마스터 문장)와 익명 클래스 객체를 주어로 취하고 어소시에이션으로 변환된 롤을 술어로 취하며 어트리뷰트를 구성하던 n개의 객체를 각각 목적어로 취하는 n개의 문장(슬레이브 문장)으로 구성된다. n+1개의 문장은 익명 클래스 객체를 통하여 하나의 문장으로 연결된다. 익명 클래스 객체는 1개의 마스터 문장에서는 목적어 역할을 담당하며 동시에 n개의 슬레이브 문장에서는 주어 역할을 담당하기 때문이다.

```
CLASS unit_segment {
 ...
 has_property_of: {
                    qualifier: maximum;
                    qualifier: operation;
                    property: pressure;
                    data: data_value;
                  }
 ...
}
```

그림 6. 그림 2의 어트리뷰트가 익명 클래스로 변형된 모습
Fig. 6. Trasformation of the attributes into the anonymous class for Fig. 2

그림 6은 그림 2에 해당하는 GPM 클래스 unit_segment의 정의부에 있는 문장 has_property_of: maximum, operation pressure data_value;의 1), 2) 단계 이후의 모습이며 그림 7은 단계 3) 이후의 모습을 RDF 그래프 형식으로 표현한 것이다. 그림 7의 원(circle)은 OWL 클래스를 의미하며 화살표는 오브젝트 프로퍼티를 의미한다. GPM은 익명 클래스를 지원하지 않으므로 그림 6의 어트리뷰트가 변환된 익명 클래스를 포함하는 구문은 GPML에서는 비문이다. 그러나 OWL에서는 익명 클래스의 사용이 가능하기 때문에 그림 6은 GPM의 OWL 변환 설명을 용이하게 하기 위하여 사용하였다.

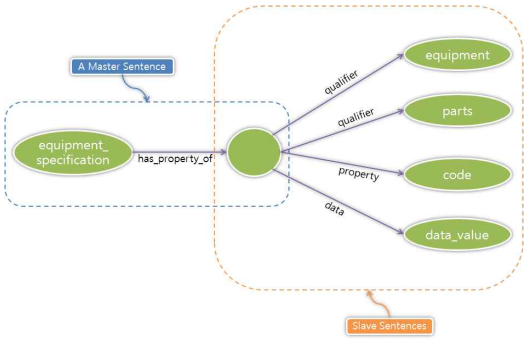


그림 7. 그림 6을 RDF 그래프로 표현한 모습
Fig. 7. An RDF graph for Fig. 6

3.1.2.2 그룹 II를 위한 GPM 문장의 변환

주어:술어:목적어의 대응수가 1:1:n인 그룹 II에 속하는 어소시에이션을 사용하는 GPM문장의 OWL 변환은 다음과 같은 방법을 사용한다.

1) GPM 문장의 주어와 술어는 그대로 유지하고 n개의 목

적격 객체 각각을 목적어로 취하는 n개의 문장으로 분해한다.

2) 그룹 I의 단계 3)을 수행한다.

그림 8과 같이 그룹 II에 포함되는 어소시에이션에 후속하는 어트리뷰트 내의 n개의 멤버 객체들은 모두 동일한 하나의 롤을 담당하며 단순 나열된 형태에 불과하다. 따라서 GPM도 그림 8 형식의 1개의 GPM 문장과 그룹 II를 위한 변환 단계 1) 이후의 n개의 GPM 문장을 동일한 의미로 해석한다.

그룹 II의 어소시에이션을 술어로 취하는 대응수 1:1:n의 GPM 문장들 역시 그룹 I의 방법으로 1개의 마스터 문장과 n개의 슬레이브 문장으로 구성한 후 OWL로 변환될 수 있다. 따라서 변환 방법의 일관성을 유지하기 위하여 그룹 II도 그룹 I의 변환 방법을 동일하게 적용시킬 수도 있었지만 본 논문에서는 롤로부터 변환된 어소시에이션들의 생성을 최소화 하고 익명 클래스의 사용을 억제하여 표현의 간결성을 획득하기 위하여 그룹 II의 경우 그룹 I과 다른 변환 방법을 선택하였다. 그림 9는 그림 8의 GPM 문장을 그룹 II의 변환 방법 단계 2)를 적용한 이후의 모습이다.

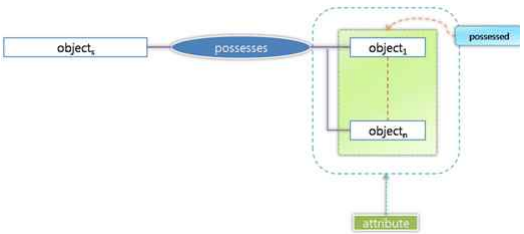


그림 8. 그룹 II의 대응수 1:n GPM 문장
Fig. 8. An 1:n GPM sentence in the group II

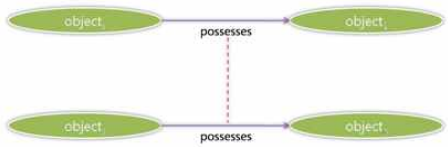


그림 9. 그림 8의 변환 과정 이후의 RDF 그래프
Fig. 9. Transformation result for Fig. 8

3.1.2.3 OWL 구문을 사용한 개체 관계 기술

OWL은 개체 관계를 기술하기 위해서 표 3의 OWL 생성자들을 사용해야 하며 이러한 OWL 생성자를 사용한 문장을 OWL에서는 조건문이라 한다. 이는 주격 클래스의 멤버 개체(individual)가 만족시켜야 하는 조건을 의미하기 때문이다. 그룹 I과 II를 위한 변환 과정을 통해 생성된 각각의 주어:술어:목적어의 대응수가 1:1:1인 개체 관계 정의들 즉, 하나의 주격 개체가 해당 술어로 지정된 특정 오브젝트 프로퍼티를 사용하여 하나의 목적격 개체와 관계를 갖고 있음을 someValuesFrom과 cardinality 생성자를 상호 보완적으로 사용함으로써 표현할 수 있다.

(형식 1) object_property SOME filler

(형식 2) object_property EXACTLY N

형식 1과 형식 2의 조건문은 Manchester OWL 구문[15]을 사용하여 표현한 것이다. SOME은 생성자 someValuesFrom을 의미하며 EXACTLY는 생성자 cardinality를 의미한다. object_property는 오브젝트 프로퍼티를 의미하며 filler는 목적격 개체를 그리고 N은 0이상의 정수로 치환될 수 있다. 형식 1의 조건문은 주격 클래스의 하나의 멤버 개체가 object_property를 술어로 하여 filler 클래스의 멤버 개체들과 최소한 하나 이상의 관계를 가지고 있어야 함을 의미한다. 형식 2의 조건문은 주격 클래스의 하나의 멤버 개체가 목적격 개체의 타입과 상관없이 object_property를 정확히 N번 사용해야 함을 의미한다. 따라서 형식 1과 2의 조건문을 각각 한 번씩 사용하고 이들을 AND로 연결할 경우 하나의 주격 클래스 개체가 object_property를 술어로 하여 특정 목적격 클래스의 멤버 개체와 정확히 1개의 연관이 있음을 의미하게 된다.

형식 1과 2의 조건문을 사용하여 association: attribute; 형식의 문장이 m개 존재하는 GPM 클래스 라이브러리에 정의되어 있는 임의의 하나의 GPM 클래스를 OWL 명명된 클래스를 위한 조건문들으로 변환할 경우 형식 1의 조건문은 GPM 클래스 정의부 내의 각각의 문장과 1:1로 대응하여 m개 생성된다. 그리고 형식 2의 조건문은 새로운 어소시에이션이 포함된 문장이 출현할 때마다 N을 1로 하여 하나씩 생성된다. 즉, 형식 2의 조건문은 어소시에이션의 종류별로 하나씩 생성된다. 이후 이미 출현한 어소시에션이 포함된 문장을 위한 처리는 해당 어소시에이션을 위한 기 생성된 형식 2의 조건문의 N을 1씩 증가시킨다. 따라서 하나의 OWL 명명된 클래스를 위한 형식 2의 조건문들의 N의 합은 m이 된다.

어트리뷰트가 변환된 OWL 익명 클래스는 형식 1의 조건문의 filler 위치로 치환되며 OWL 명명된 클래스를 위한 조건문을 생성하는 규칙이 동일하게 적용된다. 어트리뷰트 내에 p개의 롤이 출현할 수 있으며 k개의 목적격 객체가 존재할 경우 형식 1의 조건문은 k개 생성되며 형식 2의 조건문은 p개 생성된다. p개의 형식 2의 조건문들의 N의 합은 k가 된다.

그림 10은 GPM 클래스 plant_item_shape을 Manchester OWL 구문을 사용하여 변환한 것이다. 어소시에이션 is_assembled_from

은 그룹 I의 규칙에 따라 변환되었으며 어소시에이션 has_property_of는 그룹 II의 규칙에 따라 변환되었다. GPM의 어트리뷰트가 변환된 익명 클래스 정의 역시 어트리뷰트 내에 출현한 객체의 개수만큼 형식 1의 조건문이 생성되며 출현 가능한 롤의 개수만큼 형식 2의 조건문이 생성된 후 생성된 형식 2의 조건문들 내의 N의 합은 익명 클래스 내부에 생성된 형식 1의 조건문들의 합과 같음을 볼 수 있다.

```
CLASS plant_item_shape {
 is_assembled_from: shape_representation_item;
 has_property_of: unqualified name data_value_by_string;
}
```

```
plant_item_shape THAT
   has_property_of SOME (
      (qualifier SOME unqualified) AND
      (property SOME name) AND
      (data SOME data_value_by_string) AND
      (qualifier EXACTLY 1) AND
      (property EXACTLY 1) AND
      (data EXACTLY 1) AND
      (unit EXACTLY 0)
   ) AND
   has_property_of EXACTLY 1 AND
   is_assembled_from SOME  shape_representation_item AND
   is_assembled_from EXACTLY 1
```

그림 10. 맨체스터 OWL 구문으로 기술한 GPM 클래스 정의의 변환
Fig. 10. Translation of the GPM class definition using GPML into the Manchester OWL syntax

OWL은 그림 10을 다음과 같이 해석한다. 어떤 개체(individual)가 plant_item_shape의 멤버라면 다음 조건을 만족한다. 그 개체는 오브젝트 프로퍼티 has_property_of를 사용하여 1개의 익명 클래스 멤버와 연결 관계를 갖는다. 그리고 그 익명 클래스 멤버 개체는 1개의 unqualified 개체와는 qualifier로, 1개의 name 개체와는 property로 그리고 1개의 data_value_by_string 개체와는 data로 연결 관계를 맺고 있으며 unit 오브젝트 프로퍼티는 사용하지 않는다는 조건을 만족한다. 그리고 그 plant_item_shape 개체는 오브젝트 프로퍼티 is_assembled_from을 사용하여 1개의 shape_representation_item 클래스 멤버 개체와연결 관계를 맺는다는 조건을 만족한다.

3.2 GPM 온톨로지 구축

이 장에서는 3장에서 기술한 GPM의 OWL DL 로의 변환 방법을 적용하여 GPML로 기술한 GPM 클래스 라이브러리와 어소시에이션 라이브러리를 RDF/XML 형식으로 기술된 OWL DL 온톨로지로 변환 시 수행한 내용을 기술한다. RDF/XML 은 OWL 온톨로지를 XML 형식으로 기술할 수 있도록 하는 표준 스키마이다. OWL 사양을 준수하여 온톨로지를 구축할 때 RDF/XML 신택스를 사용하여 파일로 저장하는 방식이 가장 일반적인데 그 이유는 인터넷 환경에서의 활용 목적에서 가장 용이하기 때문이다. 본 논문에서는 이러한 방식으로 구축한 온톨로지를 GPM 온톨로지라고 명명한다.

3.2.1 GPM 온톨로지를 위한 네임스페이스

OWL은 RDF/XML 형식으로 구현될 때 XML 네임스페이스[16]를 사용하여 OWL 온톨로지에서 정의하는 개체 이름의 유일성을 보장한다. XML 네임스페이스는 URI (Unified Resource Identificaion) 형식으로 표현한다. GPM은 그림 11과 같이 30개의 장르를 정의하고 있다.

```
5w1h, Activity, Area, Common_part, Common_unit, ESLib,
Electrical_component, Equipment, Geometry, HVAC_component,
Information_contents, Instrumentation_and_control_component,
Location_base_definition, Others, Person_and_organization,
Piping_component, Plant_and_system, Plant_item, Port_and_connection,
Representation, Schematic_item, Specialty_item, Structural_component,
Support_unit, Topology, Valve, cable_tray_component, property, qualifier,
unit
```

그림 11. GPM 장르 목록
Fig. 11. GPM genre list

GPM 온톨로지는 31개의 네임스페이스로 구성되어 있다. 30개의 네임스페이스는 GPM의 장르(genre)를 변환한 것이며 이를 GPM 장르 네임스페이스라고 명명한다. 따라서, GPM 장르 네임스페이스에는 GPM 클래스 라이브러리에서 정의한 2442개의 클래스들이 자신의 장르에 해당하는 네임스페이스 안에서 OWL 클래스로 변환되어 정의되어 있다. 나머지 하나의 네임스페이스는 GPM 기본 네임스페이스라고 명명하며 GPM의 어소시에이션과 롤이 OWL의 오브젝트 프로퍼티로 변환되어 정의되어 있으며 GPM 클래스 라이브러리에서 정의한 2442개의 클래스들의 루트 클래스 역할을 수행할 하나의 클래스가 추가로 정의되어 있다.

GPM 기본 네임스페이스는 'http://cosmos.ssu.ac.kr/GPM#'로 정의하며 이 네임스페이스의 qualified name은 'gpm'으로 사용한다. 30개의 GPM 장르 네임스페이스는 'http://cosmos.ssu.ac.kr/GPM/장르명#'의 작명 규칙에 따라 정의되며 qualified name은 'gpm_장르명'의 규칙을 따른다. 예를 들어, GPM의 '5w1h' 장르에 해당하는 네임스페이스는 'http://cosmos.ssu.ac.kr/GPM/5w1h#'이며 qualified name은 'gpm_5w1h'이다.

하나의 네임스페이스는 하나의 RDF/XML 형식으로 기술된 OWL 파일과 대응하므로 각각의 네임스페이스를 위하여 독립된 31개의 OWL 파일을 생성하였다. GPM 온톨로지를 위

한 31개의 OWL파일의 URL은 다음과 같다.

기본 온톨로지 URL:

http://cosmos.ssu.ac.kr/GPM/gpm.owl

장르 온톨로지 URL:

http://cosmos.ssu.ac.kr/GPM/gpm_genre.owl

genre는 그림 11의 개별 장르명으로 치환한 후 접근해야 한다.

하나의 OWL 파일은 하나의 온톨로지에 해당한다. 이러한 상황에서 다른 네임스페이스에서 정의한 오브젝트를 참조할 수 있는 방법이 필요하다. OWL은 이러한 문제를 해결할 수 있는 메커니즘을 제공하며 이를 위해서 <owl:imports> 엘리먼트의 속성값으로서 참조할 OWL 파일의 URI를 할당하면 된다. <owl:imports> 엘리먼트는 전이적 특성(transitive)이 있다.[1] <owl:imports> 엘리먼트의 전이적 특성은 온톨로지 A에서 온톨로지 B를 임포트하고 온톨로지 B에서 온톨로지 C를 임포트 하고 있다면, 결과적으로 온톨로지 A가 온톨로지 B와 C를 임포트 하는 결과를 낳는다는 것을 의미한다. 이러한 메커니즘을 사용하여 참조하는 오브젝트가 다른 네임스페이스에 존재할 경우 해당 네임스페이스의 OWL 파일을 임포트 하였다.

그림 12. 기본 온톨로지에서의 장르 온톨로지 임포트
Fig. 12. Import of genre ontologies in the default ontology

3.2.2 GPM 온톨로지를 위한 오브젝트 프로퍼티

GPM의 어소시에이션은 OWL의 오브젝트 프로퍼티로 변환한다. 상속 관계를 표현할 때 사용하는 'is_classified_as'를 제외한 18개의 어소시에이션은 OWL의 오브젝트 프로퍼티로 변환될 때 그 이름이 그대로 유지된다. 어소시에이션 'is_classified_as'가 오브젝트 프로퍼티로의 변환에서 제외되는 이유는 RDF/XML이 수퍼클래스의 정의를 위하여 클래스 공리(class axiom) 'rdfs:subClassOf'를 제공하기 때문이다.

GPM에서 어트리뷰트에 출현할 수 있는 롤이 2개 이상 갖는 6개 어소시에이션의 16개 롤 만이 OWL의 오브젝트 프로퍼티로의 변환 대상이다. 본 논문에서는 GPM의 롤을 OWL의 오브젝트 프로퍼티로 변환시킬 때 OWL이 오브젝트 프로퍼티의 이름을 영어의 동사로 시작하는 관례를 따르기 위하여 다음과 같은 규칙에 따라 오브젝트 프로퍼티의 이름을 명명하였다.

- GPM의 롤 이름이 'role_name'이라면 OWL의 오브젝트 프로퍼티 'assigns_role_name_role_to'로 변환된다.

예를 들어, 어소시에이션 'boolean_operate'의 롤 중 하나인 'boolean_operator'는 OWL의 오브젝트 프로퍼티'assigns_boolean_operator_role_to'로 변환된다.

GPM의 어소시에이션은 별도의 정의역과 공역 개념을 가지고 있지 않기 때문에 어소시에이션으로부터 변환한 OWL의 오브젝트 프로퍼티는 'gpm:Thing' 클래스를 정의역과 공역으로 하였다. 'gpm:Thing' 클래스는 GPM 클래스 라이브러리에 정의되어 있는 클래스로부터 변환한 OWL 클래스들의 최상위 수퍼클래스이다.

3.2.3 GPM 온톨로지를 위한 클래스

3.2.3.1 클래스의 상속 구조

GPM의 클래스는 OWL의 클래스로 변환한다. GPM과 OWL 두 모델 모두 클래스 상속관계를 표현할 수 있는 문법을 지원한다. 따라서 GPM의 클래스 상속 계층 구조는 OWL에서 표현 가능하다. GPM 클래스 라이브러리에서 정의하고 있는 클래스들의 상속 계층 구조는 다수의 루트 클래스로 구성되어 있다. 즉, 다수의 상속 트리를 생성한다. 본 논문에서는 이들 트리의 루트 클래스들이 유일한 부모 클래스를 갖도록 디자인 하였다. 따라서, GPM의 OWL 변환 과정에서 생성한 모든 클래스들은 'gpm:Thing' 클래스를 최상위 부모 클래스로 하며 이 클래스는 기본적으로 'owl:Thing' 클래스를 상속하며 기본 네임스페이스 하에 정의하였다. GPM에서 정의한 2442개의 클래스는 각각의 장르에 대응하는 네임스페이스에서 정의하였다. 그림 13은 OWL 온톨로지 가시화 도구인 Jambalaya[17]를 사용해 GPM 온톨로지의 클래스 계층구조를 출력한 것이다.

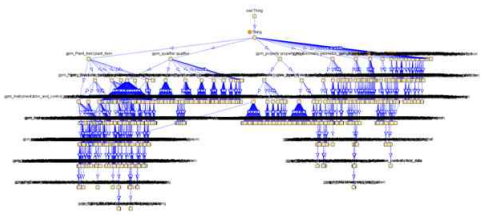


그림 13. GPM 온톨로지의 클래스 계층구조
Fig. 13. GPM ontology class hierarchy

3.2.3.2 클래스의 이름

GPM 클래스 라이브러리에서 정의하고 있는 클래스들은 이름의 첫 문자가 숫자로 시작하는 클래스들을 제외하면 OWL로의 변환과정에서 이름이 그대로 유지된다.

RDF/XML 문서는 XML 문서이므로 XML의 엘리먼트 네이밍 규칙을 준수해야 한다. XML은 엘리먼트 이름이 숫자로 시작할 수 없다.[2] RDF/XML 신택스는 클래스의 인스턴스인 개체(individual)를 선언할 때 클래스 이름을 엘리먼트의 이름으로 사용하기 때문에 이 경우 엘리먼트의 이름이 숫자로 시작하게 되는 문제가 발생한다. 본 논문에서는 이러한 문제를 방지하기 위해서 GPM의 클래스 이름이 숫자로 시작하는 경우 그 앞에 '_'를 첨가하였다. 예를 들어, '5w1h' 클래스는 '_5w1h'로 변환하였다.

```
5w1h, 3d_model, 3d_plant_model, 3d_equipment_model, 3d_piping_model,
2d_shape, 3d_shape, 90_degree_elbow, 45_degree_elbow, 90_degree_bend,
45_degree_bend, 22_half_degree_bend, 11_quarter_degree_bend,
5_5_over_8_degree_bend, 1st, 2nd, 3rd, 4th, 3d_layout, 11th, 12th, 10th,
5th, 6th, 7th, 8th, 9th
```

그림 14. 클래스명이 숫자로 시작하는 GPM 클래스 목록
Fig. 14. GPM classes of which the first character of the name is number

IV. 평가

본 논문은 OWL과 GPM 핵심 모델 사이의 객체 관계 정의에 있어서 존재하는 차이를 극복하는 방법을 제안함으로써 OWL을 GPM 핵심 모델을 위한 구현 도구로 사용할 수 있도록 하였다. 이 장은 기존의 GPM 핵심 모델의 구현을 위하여 사용된 언어들과 본 논문에서 사용한 OWL을 비교 및 평가한다.

표 3. RDF/XML과 기존 GPM 핵심 모델 구현 언어의 비교
Table 3. The comparison of RDF/XML and existing GPM core model implementation languages

	GPML	GPM-XML	RDF/XML
의미 관계 기술	가능	가능	가능
상호운용성	어려움	용이함	용이함
표준 기술 및 도구의 지원	미비함	미비함	풍부함

표 3은 기존의 GPM 핵심 모델을 위한 구현 언어들과 본 논문에서 구현 언어로 사용한 OWL DL, 정확히 표현하면 RDF/XML로 표현된 OWL DL과의 비교이다. GPML과 GPM-XML은 GPM 핵심 모델을 기반으로 하며 RDF/XML은 RDF와 OWL을 기반으로 하는 차이점이 존재하나 세 언어 모두 모델링 대상 객체 간의 관계 정의 시 의미를 부여할 수 있는 신택스를 제공하는 공통점이 있다. 그러나 GPM-XML과 RDF/XML은 둘 다 XML을 기반으로 하기 때문에 이들을 사용하여 구축한 GPM 데이터 모델은 GPML을 사용하여 구축한 것에 비하여 이기종 애플리케이션들과의 메시지 교환을 고려할 때 상호운용성을 확보하기에 용이하다. 그리고 GPM-XML과 RDF/XML을 비교하면 OWL이 OWL 온톨로지 대상 질의 언어 표준과 파서(parser)를 포함한 OWL로 작성된 파일을 핸들링 할 수 있는 API 및 OWL 모델의 특성을 활용 가능하게 하는 추론기(reasoner)와 같은 도구 지원이 풍부한 반면 GPM-XML은 GPM-XML을 지원하는 기술 표준 및 도구를 위한 연구가 미비한 상태이다. 그러므로 GPM-XML을 사용한 GPM 데이터 모델의 구축과 운용은 OWL을 사용하는 것에 비해 더 많은 노력 및 비용을 요구한다.

V. 결론

본 논문은 원자력 플랜트의 라이프 사이클 데이터를 통합 및 공유할 수 있는 공통 데이터 모델인 GPM을 구현하기 위한 언어로서 표준 웹 온톨로지 언어인 OWL을 사용하기 위하여 GPM과 OWL 두 모델 사이의 객체 관계 정의를 위한 제약 사항들의 차이를 분석한 후 이를 극복하기 위한 방법을 제안하였다. 또한 이 방법에 따라 기존의 GPML 형식으로 기술된 원자력 플랜트를 구성하는 2442개의 자원들을 모델링한 GPM 참조 라이브러리를 31개의 RDF/XML 형식의 파일로 구성된 OWL DL 기반의 온톨로지로 변환한 내용을 기술하였다. 이 온톨로지는 원자력 플랜트의 생애 주기 동안 발생하는 데이터를 통합 관리하기 위한 공유 데이터 저장소의 내부 데이터 모델로 활용될 수 있다.

참고문헌

[1] Sharon J. Kemmerer, "STEP the Grand Experience," NIST SP939, July 1999.

[2] VIPNET project, *http://www.openknow.com/vipnet/*

[3] ISO. Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 221: Application protocol: Functional Data and Their Schematic Representation for Process Plant. ISO/DIS 10303-221:2007.

[4] ISO. Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 227: Application protocol: Plant spatial Configuration. ISO

10303-227:2001.
[5] ISO. Industrial Automation Systems and Integration - Integration of Lifecycle Data for Process Plants - Part 1: Overview and Fundamental Principles. ISO 15926-1:2004.
[6] ISO. Industrial Automation Systems and Integration - Integration of Lifecycle Data for Process Plants - Part 2: Data Model. ISO 15926-2:2003.
[7] M. R. Quillian, "Semantic memory," in Semantic Information Processing, M. L. Minsky, Ed. Cambridge, MA: MIT Press, pp 227-270, 1968.
[8] S.Russell and P.Norvig, "Artificial Intelligence: A Modern Approach," Prentice-Hall, pp. 316-323, 1995.
[9] T. Yoon, Y.Oota, Y. Naka, T. Yoshinaga, K. Shibao, M. Igoshi, K. Matsushima, T. Suzuki, "Knowledge Fusion among the Virtual Production Enterprises within the Technology Information Infrastructure Environment", IEEE International Engineering Management Conference, pp. 35-40, Cambridge, UK, Aug. 2002.
[10] Michael K. Smith, "OWL Web Ontology Language Guide," W3C Recommendation, 2004, *http://www.w3.org/TR/2004/REC-owl-guide-20040210/*
[11] 조대웅, 최지웅, 김명호, "OWL 온톨로지 사용을 위한 SPARQL 쿼리 툴," 한국컴퓨터정보학회논문지, 제 14권, 제 11호, 21-30쪽, 2009년 11월.
[12] 박지현, 양재군, 배재학, "Protege를 이용한 기업 온톨로지 기반 구축 및 활용 -조선 건조공정 표현과 분석- ", 한국컴퓨터정보학회논문지, 제 14권, 제 3호, 27-39쪽, 2009년 3월.
[13] Dave Beckett, "RDF/XML syntax specification (revised)," W3C Recommendation, 2004, *http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/*
[14] Franz Baader, Ian Horrocks, Ulrike Sattler, "Description Logics as Ontology Languages for the Semantic Web," Mechanizing Mathematical Reasoning, Vol. 2605, pp. 228-248, 2005.
[15] Matthew Horridge, Nick Drummond, John Goodwin, Alan L. Rector, Robert Stevens, Hai Wang, "The Manchester OWL Syntax," OWLED, 2006.
[16] Tim Bray, "Namespaces in XML 1.0 (Second Edition)," W3C Recommendation, 2006, *http://www.w3.org/TR/2006/REC-xml-names-20060816/*
[17] M. A. D. Storey, M. A. Musen, J. Silva, C. Best, N. Ernst, R. Fergerson, and N. F. Noy, "Jambalaya: Interactive visualization to enhance ontology authoring and knowledge acquisition in protege," Workshop on Interactive Tools for Knowledge Capture, 2001.

저 자 소 개

최 지 웅

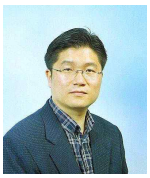
2001 : 숭실대학교 컴퓨터학부 학사
2003 : 숭실대학교 컴퓨터학과 석사
2007 - 2008 : 고등기술연구원 연구원
2003 - 현재 : 숭실대학교 컴퓨터학과 박사과정
관심분야 : 시맨틱 웹, BI, 보안

박 호 병

1999 : 숭실대학교 전자계산학과 학사
2002 : 숭실대학교 컴퓨터학과 석사
2004 : 숭실대학교 컴퓨터학과 박사수료
2006 - 2008 : 고등기술연구원 선임연구원
현재 : (주)엠케이 부설연구소 선임연구원
관심분야 : 컴파일러, OWL, VIPNET

김 형 진

1991 : 미국 톨리도 대학교 물리학 학사
1993 : 미국 셀트럴 미시간 대학교 물리학 석사
1995-현재 : 고등기술연구원 수석 연구원
관심분야 : 플랜트 정보 처리 기술, ISO 15926, GPM, PLM



김 명 호

1989 : 숭실대학교 컴퓨터학부 학사
1991 : 포항공과대학교 전자계산학과 공학석사
1995 : 포항공과대학교 전자계산학과 공학박사
1995 : 한국전자통신연구소 선임연구원
1998, 2006 : 미국 테네시주립대 교환교수
1995-현재 : 숭실대학교 컴퓨터학부 교수
관심분야 : 분산/병렬 컴퓨팅, 그리드, 웹서비스, BI, 보안