객체지향 온톨로지 관리기를 이용한 바이오 온톨로지 생성

양 경 아[†] · 양 형 정[††] · 양 재 동[†††]

요 약

본 논문에서는 온톨로지를 구축하고 관리하는 객체 지향 온톨로지 관리기(Object-oriented Ontology Manager, OOM)를 이용한 바이오 온톨로지의 구축과 관리를 보인다. OOM은 객체지향 패러다임을 기반으로 보다 정교한 온톨로지의 구축을 위해 다양한 관계들을 지원하며, 다른 온톨로지와의 병합을 위해 다른 언어로의 온톨로지 번역기를 제공한다. OOM은 생명 정보학 관련 온톨로지의 많은 지식을 내포하고 있는 용어와 이들 간의 복잡한 관계를 쉽게 표현할 수 있도록 가시적인 개념화에 의해 온톨로지를 구축한다. 또한, 관계 링크에 기반한 추론 기능을 지원하여 온톨로지 구축과 관리를 용이하게 하고, 질의 시 적합한 결과를 제시한다. 따라서 OOM은 기존의 온톨로지 툴에 비해 단순하면서도 의미적 표현력이 뛰어나 바이오 온톨로지와 같은 복잡한 응용의 온톨로지도 개념적으로 쉬운 방식으로 모델링 할 수 있다. 또한 OOM에서 지향하는 객체지향 패러다임은 표준 온톨로지 언어의 내부 스키마와 유사함으로 의미적 손실없이 표준 온톨로지 언어로 쉽게 변환되어 온톨로지들 간의 병합이 가능하다.

Bio-Ontology Generation Using Object-Oriented Ontology Manager

Kyung-Ah Yang[†] · Hyung-Jeong Yang[††] · Jae-Dong Yang[†††]

ABSTRACT

This paper presents an approach to the development of bio-ontology using the Object-oriented Ontology Manager(OOM). OOM views a term of an ontology as an object which can be an instance or a concept. OOM facilitates the semi-automatic construction of ontologies by an intuitive interface and by inferencing with links among complicated and informative ontology terms. The main advantage of OOM is simple-to-use not compromising expressiveness so that ontologies in a complicated domain such as bioinformatics can be modeled intuitively. The ontologies constructed by OOM are easily exported to ontologies in other ontology languages without semantic loss because the structures of both the ontology by OOM and the ontologies in most of standard ontology languages are analogous. A translator to another standard ontology language is also provided by OOM so that the ontology can be combined with others to be applied to more complicated applications.

키워드 : 온톨로지(Ontology), 바이오 인포매틱스(Bioinformatics), 시맨틱 웹(Semantic Web), 온톨로지 구축 툴(Ontology Development Tools)

1. 서 론

최근에 생명 정보학에 관한 관심이 높아지면서 생명 정보학 분야에서 여러 데이터베이스들을 통하여 커뮤니티 지식을 표현하고 이를 공유하기 위한 온톨로지의 역할이 중요시되고 있다[1-3]. 변화하는 생명 정보학 데이터를 표현하고 데이터들 간의 시맨틱을 유추하는데 온톨로지를 이용하는 연구들이 이루어지고 있으며, 생명공학 분야 전반에 폭넓게 활용하기 위해 Gene Ontology(GO), MGED 온톨로지, Open Biological Ontology(OBO) 등과 같은 바이오 온톨로지들이 개발되고 있다. 바이오 온톨로지는 생물 정보학 도메인에 대한 커뮤니티의 관점을 공유 가능한 형식으로 제공함으로써 유전자와 분자학 분야에서 생물학적 기능을 예측하거나 해석하는 작업에 활용될 수 있다[3-6].

사용자들은 바이오 온톨로지를 쉽고 편하게 작성하기 위해 주로 온톨로지 구축 시스템을 사용하는데, 온톨로지 구축 툴로서 OntoEdit[7], Protege-2000[8], OilEd[9], WebODE[10], Ontolingua[11] 등이 개발되었다. 이 중 Protege-2000, OntoEdit, OilEd 등과 같은 온톨로지 툴은 예술가 저작목록, 컴퓨터 교육, 미술 공예, 생명 정보학과 같은 다양한 영역에서 사용되고 있다[3, 13-15]. Ontoweb deliverable 1.3[12]는 이들 온톨로지 개발 툴들을 구조, 상호운영성, 지식 표현, 추론, 사용성 등의 여러 관점에서 비교하였으며, [2]에서는 GO를 이용하여 Protege-2000, Chimaera, Dag-Edit를 바이오 온톨로지 툴의 관점에서 평가하였다.

또한 온톨로지 언어인 OWL(Ontology Web Language)을 편집하는 도구로 Protege-2000과 OilEd 등을 비롯한 여러 온톨로지 툴들이 소개되었다. 이러한 도구들은 온톨로지 개념을 계층화하여 온톨로지를 편집할 수 있도록 지원하며 외

† 준 회 원 : 전북대학교 대학원 컴퓨터통계정보학과
†† 준 회 원 : 카네기멜론대학 컴퓨터과학과 포스트닥터 연구원
††† 정 회 원 : 전북대학교 컴퓨터과학과 교수
논문접수 : 2004년 3월 25일, 심사완료 : 2004년 6월 11일

부 추론기와 연동할 수 있는 인터페이스를 제공하여 그 결과를 OWL로 저장할 수 있다. 그러나 이 방식들은 모델링을 위해 제공하는 표기법들이 지나치게 복잡해서 온톨로지의 시맨틱을 직관적으로 이해하기 어려운 문제점이 있다.

본 논문에서는 온톨로지를 구축하고 관리하는 객체 지향 온톨로지 관리기(Object-oriented Ontology Manager, OOM)를 제안한다. OOM에서는 객체지향 패러다임을 기반으로 보다 정교한 온톨로지 작성을 위해 다양한 관계들을 지원하며, 다른 온톨로지와의 병합을 위해 다른 언어로의 온톨로지 번역기를 구축하였다. OOM은 생명 정보학 관련 온톨로지의 많은 지식을 내포하고 있는 용어와 이들 간의 복잡한 관계를 쉽게 표현할 수 있도록 가시적인 개념화에 의해 온톨로지를 생성하며, 관계 링크에 기반한 추론 기능을 지원하여 온톨로지 구축과 관리를 용이하게 하고, 질의 시 적합한 결과를 제시한다. 따라서 OOM은 Protege-2000이나 OilEd 등의 온톨로지 툴에 비해 단순하면서도 의미적 표현력이 뛰어나 바이오 온톨로지와 같은 복잡한 응용의 온톨로지도 개념적으로 쉬운 방식으로 모델링 할 수 있다. 또한 OOM에서 지향하는 객체지향 패러다임은 OIL, DAML+OIL, OWL 등의 온톨로지 언어의 내부 스키마와 유사함으로 의미적 손실없이 이들 온톨로지 언어로 쉽게 변환되어 온톨로지들 간의 병합이 가능하다.

다른 온톨로지 언어로의 변환을 위해 OOM은 다음과 같은 세 가지 단계를 거치는 온톨로지 번역기를 제공한다. 우선 객체 지향 온톨로지의 구조를 RDF/RDFS 언어 구조로 대응시키기 위한 매핑 스키마를 정의한다. 두 번째로, 정의된 매핑 스키마를 기반으로 적절한 XML 네임스페이스를 설계한다. 네임 스페이스는 객체 지향 온톨로지의 객체와 관계를 RDF/RDFS로 번역하는데 시스템이 참조하기 위해 정의된다. 세 번째, XML 네임스페이스를 기반으로 XML 파서를 이용하여 온톨로지 변환을 수행한다. DAML+OIL, OWL과 온톨로지 언어 대신 RDF/RDFS를 목적 온톨로지 언어로 선택한 이유는 RDF/RDFS가 온톨로지 언어 중 기본이 되는 언어이기 때문이다. 그러므로 RDF/RDFS 형식의 온톨로지는 확장되고 변경되고 재사용하기 용이하다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구에 대해 살펴보고, 3장에서는 바이오 온톨로지의 구성에 대해 보인다. 4장에서는 관계 기반 추론을 이용한 후보용어 추천과 질의 결과 도출에 대해 논의하며 5장에서는 OOM에서 구축한 온톨로지를 RDF/RDFS 언어의 온톨로지로 변환하기 위한 세 가지 단계에 대하여 세부적으로 살펴본다. 6장에서는 OOM의 전체적 시스템 구성과 구현 내용에 대해 알아본다. 마지막으로 결론 및 향후 연구를 제시한다.

2. 관련 연구

바이오 온톨로지는 생물학 분야에서 잘 알려지지 않은 개체에 대한 연구시 필요한 사전 지식의 역할과 다양한 생물정보 데이터에 대한 자동화된 기계처리 가능성을 높이는 역할을 한다. 이러한 바이오 온톨로지 중 대표적인 것으로 기존의 DB에 저장된 데이터를 활용하여 생성한 GO(Gene Ontology)를 들 수 있다. 그동안 생명공학 분야에서 바이오 온톨로지들을 생성하는 것을 돕고 이를 효율적으로 관리하기 위해 Protege-2000이나 OntoEdit 등과 같은 온톨로지 작성 도구들이 연구, 개발되어 왔다.

[2]에서는 GO와 Signal-Ontology 내에서 정보의 중복성 문제를 해결하는 한 방안으로 서로 다른 관점이 부각되는 바이오 온톨로지들을 여러 가지 온톨로지 도구들을 이용해 새로운 온톨로지로 병합하고 이 과정에서 각각 온톨로지 도구의 기능을 평가하는 연구를 수행하였다. 이 연구에서는 서로 다른 온톨로지들을 Protege-2000을 사용하여 병합하였다. Protege-2000은 다양한 플러그인을 사용하여 사용목적에 맞게 프로그램을 확장하여 사용할 수 있으며, PROMPT와 Chimaera, 이 두 가지 플러그인을 사용하여 각 병합과정에서 그 기능들을 테스트하였다. 이들을 이용해서 병합을 수행하는 경우, 오랜 시간 동작하게 되면 비정상적인 시스템 오류가 일어나거나 병합과정에서 프로그램의 화면 인터페이스가 사용자가 이용하기에 혼란스럽게 구성되어 있다는 문제점들이 발견되었다.

[3]에서는 해부학적 컨텐츠를 강화시키기 위해서 사용되었던 FMA(Foundational Model of Anatomy)를 생명정보학 영역에서 해부학에 대한 서로 다른 관점을 연관시키기 위한 참조 온톨로지로 사용한다. FMA를 프레임 기반 온톨로지로 변환하기 위해서 인코딩 편집기인 Protege2000을 사용하였으며, 이 환경 내에서 FMA의 각 요소들을 프레임 기반에 맞게 매핑하여 편집하였다.

[16]에서는 GO에서 나타나는 정보의 중복성 문제 해결과 보다 원활한 상호운용성을 위해 Protege-2000을 사용하여 GO를 프레임 기반 표현형태로 변환하는 연구를 수행하였다. GO를 프레임 기반으로 나타내기 위해 Protege-2000 내에서 GO 내의 각 용어들은 하나의 프레임으로 매핑되고 is-a나 part-of와 같은 관계는 슬롯 정보로 매핑되어 자료가 표현된다. 하지만 GO에서 지원하는 이와 같은 관계만으로는 바이오 온톨로지와 같은 복잡한 온톨로지의 세부적 관계를 자세히 표현하기 어렵다.

[17]에서는 생명정보공학 분야에서 사용되는 온톨로지의 설계와 개발을 사용자가 용이하게 할 수 있도록 OilEd 편집기를 사용하는 사례를 연구하였다. OilEd는 온톨로지 내에서 강력한 로직을 표현할 수 있는 언어인 OIL을 편집하는 도구이며, 온톨로지의 설계를 지원하기 위한 추론 기능을 제공한다. 구문기반으로 구성되어 있는 온톨로지인 TAMBIS를 의미적으로 잘 구성된 프레임기반 형식으로 변환하기 위해 OilEd를 통해 각 개념들은 계층적으로 표현하고 관련 속성들은 슬롯형태로 나타내었다.

본 논문에서 개발한 OOM는 온톨로지의 계층관계와 상속관계를 시각화하여 나타내고 개념과 개념 간의 다양한 관계

를 직관적으로 이해하기 쉬운 방식으로 지원하여 사용자가 효과적으로 풍부한 바이오 정보를 함유한 온톨로지를 생성, 관리할 수 있다. 또한 시스템 내부에서 작성된 온톨로지를 온톨로지 언어인 RDF/RDFS 언어로 변환하는 기능이 내포되어 있기 때문에 별도의 프로그램을 거쳐 온톨로지 언어로 변환하는 수고로움을 덜 수 있다. 변환되는 온톨로지 언어인 RDF/RDFS는 시맨틱 웹 내부에서 가장 기본이 되는 언어로 인정되기 때문에 다른 온톨로지 언어로 변환된 외부 온톨로지들과 웹 상에서나 지역적인 시스템 상에서나 상관없이 온톨로지 병합을 수행할 수 있다.

3. OOM에 의한 생명 정보학 온톨로지 구축

OOM은 기본적으로 객체 지향 시소러스를 구축하고 관리하기 위해 개발된 객체지향 시소러스 관리기(Object-oriented Thesaurus Manager, OTM)[18]와 같은 패러다임을 유지하면서 보다 정교한 온톨로지의 구축을 위해 다양한 관계들을 지원한다. 본 장에서는 객체 지향 온톨로지 관리기의 가시적 개념화에 의한 생명정보학 온톨로지의 구축을 보인다.

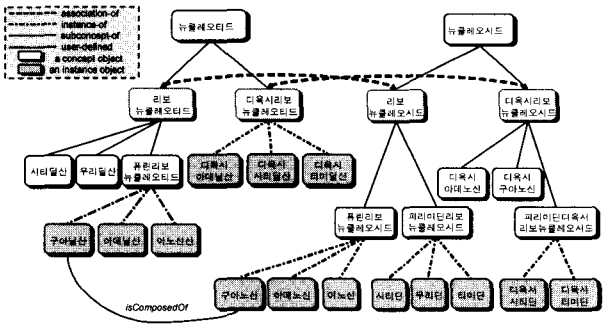
3.1 온톨로지 객체 표현

객체 지향 온톨로지 관리기에서 생명 정보학 온톨로지 용어들은 객체로 간주되며, 객체는 온톨로지 객체(*Ontology Object*), 최상위 개념 객체(*Top concept Object*), 개념 객체(*Concept Object*), 인스턴스 객체(*Instance Object*)로 나뉜다. 온톨로지 객체는 한 도메인 내에 표현된 모든 객체를 지칭하며, 따라서 온톨로지 객체에 명시된 제약사항과 속성이 이 도메인 내의 모든 객체들에 적용된다. 최상위 개념 객체는 개념 객체들 사이의 최상위 개념을 의미한다. 개념 객체는 하위 개념 객체나 인스턴스 객체를 가질 수 있다. 하위 개념 객체는 다시 더 세분화된 하위 개념 객체나 인스턴스를 가질 수 있으므로, 최상위 개념 객체로부터 인스턴스 객체까지 개념 계층을 형성할 수 있다. 즉, 다양한 분야의 온톨로지 도메인에서 온톨로지 객체로부터 최상위 개념이 존재하고 이 개념 객체로부터 세부적인 객체들이 계층적으로 추가되며, 상위 객체의 속성 정보는 하위 객체들에게 상속된다.

3.2 온톨로지 관계 표현

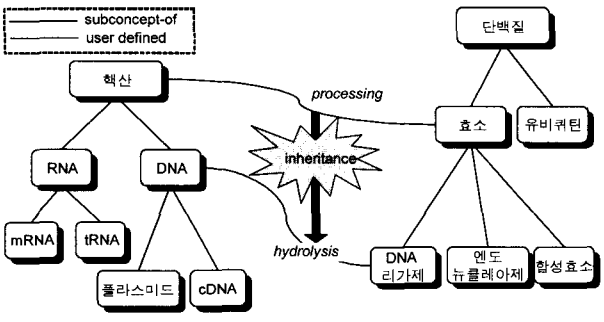
OOM은 Protege-2000이나 OILED와 같은 온톨로지 관리기에서 지원하는 제한된 시스템 정의 관계들을 세분화하여 보다 정확하고 표현력이 높은 온톨로지를 구축할 수 있도록 지원한다. OOM에서 온톨로지 객체들 사이의 관계는 *subconcept-of, instance-of, part-of, association-of, synonym-of, user-defined* 등으로 표현할 수 있다.

일반적으로 제공되는 일반화 관계는 개념 객체들 사이의 일반화(*subconcept-of*) 관계와 개념과 인스턴스 객체 사이의 인스턴스화(*instance-of*) 관계로 세분화되어 정의되며, 다양한 의미로 해석되던 수평적인 관계 역시 집단화(*part-of*), 연관화(*association-of*) 관계로 세분화하였다. 한 개념 객체가 다른 개념 객체와 집단화나 연관화 관계를 가진다면 그 관계 정보는 속성으로써 그 하위 개념 객체들에 상속될 수 있다. 예를 들어, (그림 1)에서와 같이 '뉴클레오티드'는 '리보뉴클레오티드', '디옥시리보뉴클레오티드'와 *subconcept-of*의 관계로 연결되어 있으며 이들은 더욱 세분화된 하위 객체로 표현되고, 인스턴스 객체들과 *instance-of*의 관계를 가짐으로써 개념 계층을 이룬다. '리보뉴클레오티드'와 '디옥시리보뉴클레오티드'는 '뉴클레오시드'의 하위 개념 객체인 '리보뉴클레오시드', '디옥시리보뉴클레오시드'와 각각 *association-of* 관계가 있다.



(그림 1) 생명 정보학 관련 온톨로지 일부

OOM에서 모든 객체는 각각의 이름을 지니는데, 만약 두 객체가 같은 의미를 나타내지만 다른 이름일 경우 동의어(*synonym-of*) 관계로 표현된다. 또한 OOM에서는 사용자가 사용자 정의(*user-defined*) 관계를 통해 더욱 의미가 분명한 관계를 정의할 수 있다. 직관적으로 두 객체가 관계가 있음은 OOM에서 여러 다양한 형태의 링크로 표현된다. (그림 1)에서 '뉴클레오티드' 계층의 '구아닐산' 인스턴스는 '뉴클레오시드' 계층의 '구아노신' 인스턴스에 의해 '구성된다'는 세부 관계를 지니며 그 관계는 사용자에 의해 *isComposed-Of*라고 설명될 수 있다.



(그림 2) OOM에서 온톨로지 속성의 상속성

OOM에서 온톨로지 객체가 상속성을 지니는 것과 마찬가지로 사용자 정의 관계의 속성은 하위 사용자 정의 관계로

상속될 수 있다. (그림 2)에서와 같이 '효소'와 '핵산' 객체 간에 사용자 정의 관계로 *processing*이 지정되어 있고, '효소'의 하위객체인 'DNA 리가제'와 '핵산'의 하위객체인 'DNA' 간에 *hydrolysis*가 사용자 정의 관계로 정의되었을 때, *hydrolysis*는 *processing*을 상속받음으로써 속성 간의 계층 관계를 형성할 수 있다.

살펴본 바와 같이 객체지향 온톨로지는 객체들과 객체들 간의 관계들로 표현되며 다음과 같이 정의될 수 있다.

[정의 1] 객체지향 온톨로지 Onto(T)는 다음과 같은 트리플의 집합으로 정의된다.

$$Onto(T) = \{t = <r, o_1, o_2> \mid o_1, o_2 \in O,\ r \in R\}$$

여기에 R = {subconcept-of, instance-of, part-of, association-of, synonym-of, user-defined}, O는 객체(개념 객체 또는 인스턴스 객체)들의 집합이다. 따라서 $<r, o_1, o_2>$는 o_1과 o_2 사이에 r인 의미 관계가 존재한다는 뜻이다. 예를 들어, (그림 2)의 객체지향 온톨로지에서 <processing, 효소, 핵산>, <hydrolysis, DNA 리가제, DNA> ∈ Onto(T)이다.

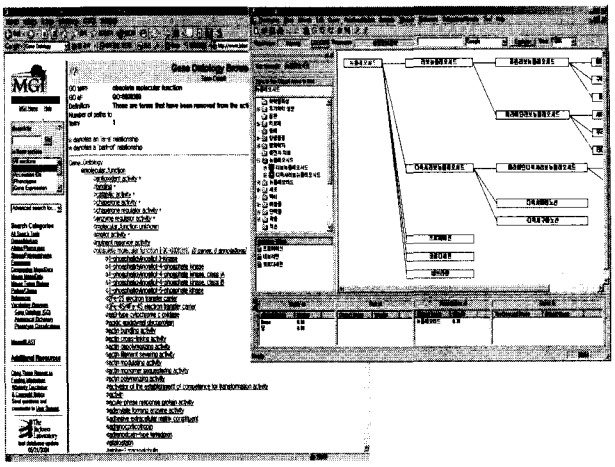
또한 객체지향 온톨로지의 관계집합은 {subconcept-of, instance-of}로 정의되는 수직적 관계 집합 $Onto_1(T)$과 {part-of, association-of, synonym-of, user-defined}로 정의되는 수평적 관계 집합 $Onto_2(T)$로 구분할 수 있다. 다음은 이들에 관한 정의이다.

[정의 2] R_1 = {subconcept-of, instance-of}이고 R_2 = {part-of, association-of, synonym-of, user-defined}일 때, 모든 t = $<r. o_1, o_2>$ ∈ Onto(t)에 대해

$r \in R_1$이면 $t \in Onto_1(T)$이고 $r \in R_2$이면 $t \in Onto_2(T)$이다.

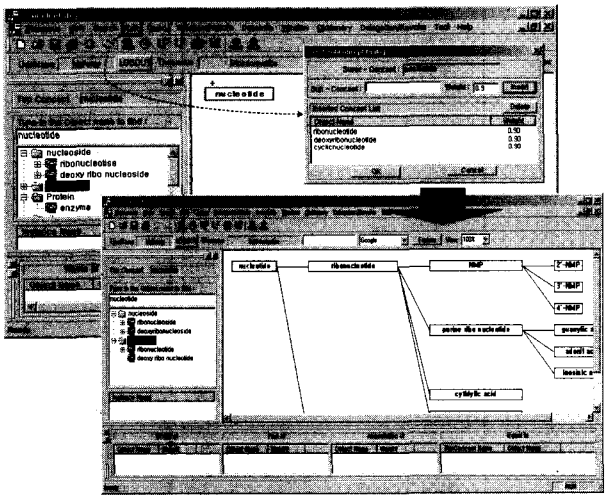
결과적으로 $R = R_1 \cup R_2$이고 $Onto(T) = Onto_1(T) \cup Onto_2(T)$이다.

OOM에서 온톨로지 구축 시 객체의 생성과 객체들 간의 관계 설정은 같은 성질의 개념을 한 곳으로 분류하는 가시적 개념화에 의해 이루어진다. 가시적 개념화는 펼침/줄임 및 항해방식의 브라우징, 그룹핑 등을 통해 지원되며 생명정보학의 풍부한 지식이 쉽게 표현될 수 있도록 한다. 즉, 펼침/줄임과 항해 방식의 개념 브라우징 기능은 매우 복잡하게 표현된 온톨로지 객체들을 사용자의 관점에 따라 순차적으로 찾아볼 수 있게 함으로써 사용자가 온톨로지 구축 시 일반적인 의미의 객체로부터 구체적인 객체를 점진적으로 구축할 수 있도록 한다. 이 기능은 하나의 최상위 개념으로부터 파생되는 많은 하위 개념들 중에서 사용자가 의도한 방향에 있는 개념들만을 순차적으로 브라우징할 수 있게 함으로써 복잡한 용어들 사이의 관계에 대한 직관적인 이해를 가능하게 한다. 또한 그룹핑, 잘라내기/붙여넣기 등의 WYSIWYG 기능은 온톨로지의 유지보수 및 구조변경이 용이하도록 돕는다.



(그림 3) GO Browser와 OOM의 인터페이스 비교

(그림 3)은 생명공학 온톨로지 뷰어인 Gene Ontology Browser와 OOM의 인터페이스를 비교한 것이다. Gene Ontology Browser는 대용량의 생명공학 지식정보 체계인 GO를 텍스트 기반의 계층형태로 표현하여 사용자가 링크를 따라 원하는 정보를 찾아갈 수 있도록 지원한다. 하지만 이러한 인터페이스로는 생명 공학 도메인의 용어 간의 복잡하게 얽혀있는 관계들을 쉽게 파악하기 어려우며, 계층 레벨 간 이동을 통해 정보 깊숙한 곳까지 탐색하였을 경우 원래 있던 위치로 되돌아오기가 쉽지 않다. 반면 OOM에서는 보다 그래픽적인 뷰어를 지원함으로써 용어 간의 이동 사항을 손쉽게 알 수 있게 하였고, 온톨로지 용어에 설정되어 있는 다양한 관계들을 인터페이스 하위에 있는 윈도우들을 통해 제공함으로써 사용자가 쉽게 온톨로지의 구조 사항을 파악할 수 있도록 지원한다. 또한 OOM에서 제공하는 간략하고 필수적인 기능들을 통해 온톨로지의 생성은 물론 구축된 온톨로지의 수정도 용이하게 하였다. 아래 그림은 OOM에서 새로운 온톨로지 용어들을 삽입하는 과정을 보인 것이다.



(그림 4) OOM을 이용한 온톨로지 객체의 삽입

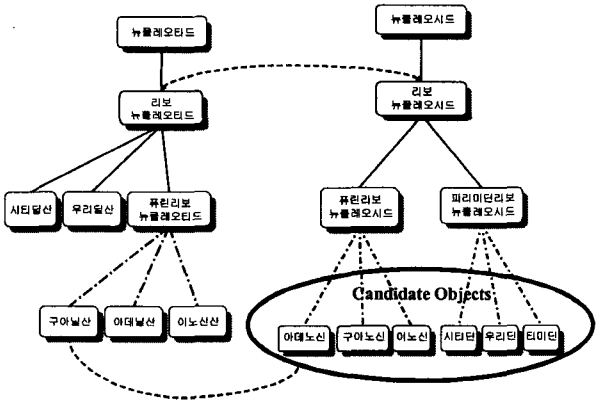
4. 관계 기반 추론

OOM에서는 객체들 사이에 설정된 관계들을 기반으로 하여 추론을 수행한다. OOM에서 관계들은 5가지의 시스템 정의 관계와 여러 사용자 정의 관계를 지원하기 때문에 이들 관계를 기반으로 한 추론은 복잡한 온톨로지로부터 지식을 이끌어 내어 사용자의 이해를 돕는다. 추론 결과는 온톨로지 구축 시 후보 객체의 추천에 사용되어 반자동의 온톨로지 구축을 가능하게 할 뿐 아니라 질의 시에도 사용되어 온톨로지가 다양한 관점의 커뮤니티의 공통된 관점을 유지하는 기능을 수행할 수 있도록 한다. 즉, 관점에 따라 다양하게 표현되는 질의에 대해 공통적으로 이해할 수 있는 결과를 보여준다.

4.1 추론에 의한 후보 객체의 추천

온톨로지의 구축 시 이미 생성된 객체들 사이의 관계를 설정하기 위해 후보 객체를 추천하는데 OOM의 관계 기반 추론 기능이 이용된다. 후보 객체의 추천은 주로 *subconcept-of*와 *instance-of*의 상속 관계에 의해 이루어진다. 즉, 객체의 속성으로 표현된 *part-of, association-of, synonym-of, user-defined* 관계가 수직 관계인 *subconcept-of*와 *instance-of*의 관계에 의해 상속되어 새로운 묵시적인 관계가 생성된다. 시스템이 생성하는 묵시적인 관계는 OOM의 사용자 인터페이스에 링크로 표시되지 않고 내부적 추론에서 사용된다.

묵시적 관계를 이용하면 객체들 사이에 복잡하게 교차하는 관계성들을 구조적으로 파악해 낼 수 있기 때문에 온톨로지의 구축 및 유지에 전문가의 부담을 감소시킬 수 있으며, 질의의 의미를 파악하기 위한 추론에 유용한 전략으로 이용될 수 있다.



(그림 5) 관계 기반 추론에 의한 후보 객체 추천

(그림 5)는 시스템이 파악한 묵시적인 관계에 의해 관계를 설정할 수 있는 후보 객체를 추천하는 예이다.'구아닐산'에 명시적인 *association-of* 관계를 설정하고자 OOM에 추천 객체를 의뢰하면, '리보뉴클레오티드'와 '리보뉴클레오시드'의 *association-of* 관계가 상속되어 OOM이 관계 설정 가능한 {아데노신, 구아노신, 이노신, 시티딘, 우리딘, 티미딘} 집합이 후보 객체로 제시된다.

4.2 질의에 대한 추론

본 절에서는 바이오 온톨로지의 다양한 관계들에 기반한 추론을 이용하여 질의에 대해 적절한 답을 유도하는 과정을 보인다. 사용자는 질의에 대한 검색 결과를 통해 바이오 온톨로지 내의 복잡한 관계를 이해할 수 있다. 질의의 의미를 파악하기 위해 최상위 객체를 가지는 하나의 개념 계층을 부분 순서 집합(POSet : Partial Ordered Set)으로 정의했을 때, 객체 지향 온톨로지 개념 계층은 여러 개의 최소 상계(LUB : Least Upper Bound)나 최대 하계(GLB : Greatest Lower Bound)를 가지기 때문에 최소 상계 집합(LUBS : Least Upper Bound Set)과 최대 하계 집합(GLBS : Greatest Lower Bound Set)을 구할 수 있다.

객체지향 온톨로지의 질의에 대한 의미를 파악하기 위해 다음과 같이 온톨로지 계층 $Onto_1(T)$에서 임의의 객체 집합 $O' \subseteq O$의 Upper Bound Set(UBS), Lower Bound Set (LBS)를 정의할 필요가 있다.

[정의 3] $r \in R_1$과 $o' \in O' \subseteq O$에 대해, $sup(o')=\{o \mid <r, o, o'> \in Onto_1(T)\}$이고, $sub(o')=\{o \mid <r, o,' o> \in Onto_1(T)\}$일 때, $UBS(o')$과 $LBS(o')$는 다음과 같이 정의된다.

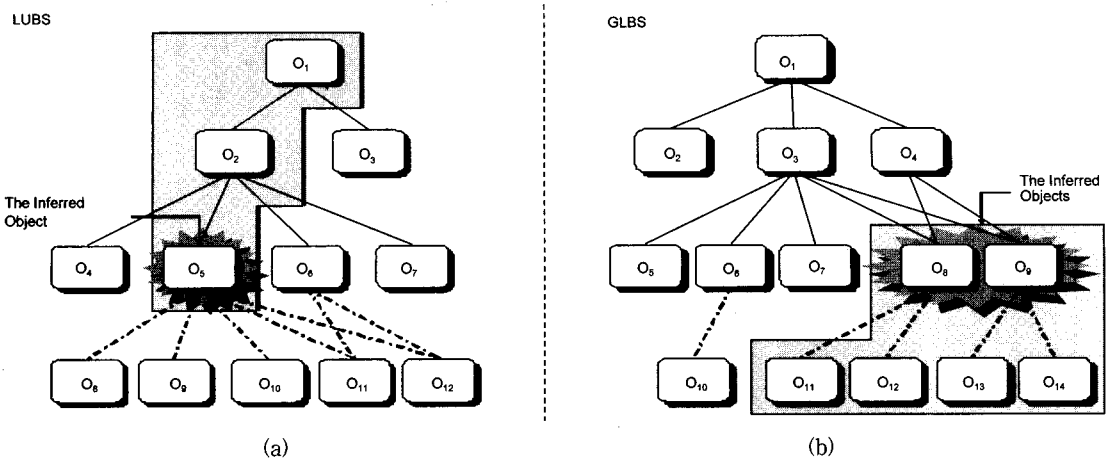
$$USB(O') = \bigcap_{\forall o' \in O'} sup(o'),\ LSB(O') = \bigcap_{\forall o' \in O'} sub(o')$$

하나의 최상위 객체로부터 파생되는 객체 계층은 여러 개의 LUB와 GLB를 가질 수 있기 때문에 격자 구조로는 표현될 수 없다. 따라서 다음과 같이 LUBS와 GLBS에 대한 정의가 필요하다.

[정의 4] $Onto_1(T)$와 $O' \subseteq O$의 모든 객체 $o'i \subseteq O'$, $i = 1, 2, \cdots, n$에 대한 $LBUS(O')$와 $GLBS(O')$는 $r \in R_1$일 때 다음과 같이 정의되며, 각각 $\vee_{i=1}^{n} o_i$ 와 $\wedge_{i=1}^{n} o_i$ 로 표시한다.

$$LUBS(O') = UBS(O') - \{o \mid o, o' \in UBS(O'), <r, o, o'> \in Onto_1(T)\}$$
$$GLBS(O') = LBS(O') - \{o \mid o, o' \in LBS(O'), <r, o'. o> \in Onto_1(T)\}$$

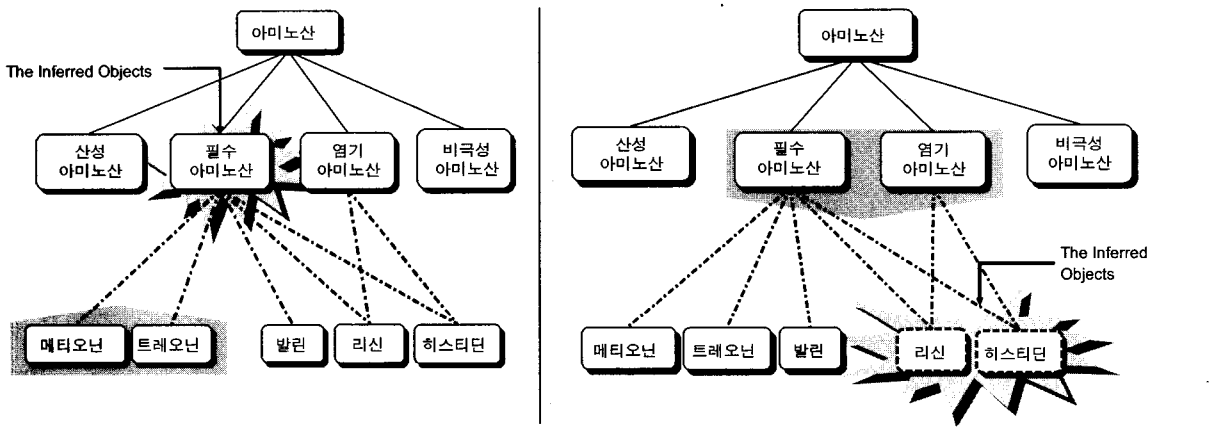
(그림 6)은 이와 같은 정의를 그래프로 도식화하여 나타낸 것이다. (그림 6)(a)에서 인스턴스 객체인 O_8과 O_9의 LUBS를 찾는 경우, 계층을 따라 두 객체의 공통된 상위 객체를 추적하여 Upper Bound Set을 구한 후 그 중에서 가장 낮은 레벨의 객체를 탐색한다. (그림 6)(b)에서 O_3과 O_4의 GLBS를 구하는 경우, 두 객체의 공통된 하위 레벨의 객체를 링크를 통해 따라가고 검색된 객체에 하위 레벨 객체로 확장하여 Lower Bound Set을 구한 후 그 중에 가장 상위 레벨의 객체들을 추론하여 제시한다.



(a) (b)

(그림 6) (a) LUBS의 처리 과정 (b) GLBS의 처리 과정

사용자의 질의는 [영역 한정질의(Scope-Query), 제약질의(Qualification-Query)]와 같이 두 개의 부분 질의로 구성된다. 영역 한정질의는 *subclass-of*와 *instance-of* 링크를 항해하며, 제약질의는 그 밖의 관계에 의한 링크를 항해하여 검색 질의를 구성할 수 있도록 한다. 즉, 영역 한정질의는 객체 지향 온톨로지의 여러 개념 계층들 중 특정 계층의 일부분으로 사용자의 브라우징 범위를 한정하며, 제약질의는 다양한 관계에 의해 영역 한정질의의 범위를 더욱 제한하기 위해 사용된다. 영역 한정질의에서는 일반화(OR) 연산과 구체화(AND) 연산이 이루어진다.



(a) (b)

(그림 7) (a) [메티오닌 OR 트레오닌, *] (b) [필수아미노산 AND 염기성아미노산, *]

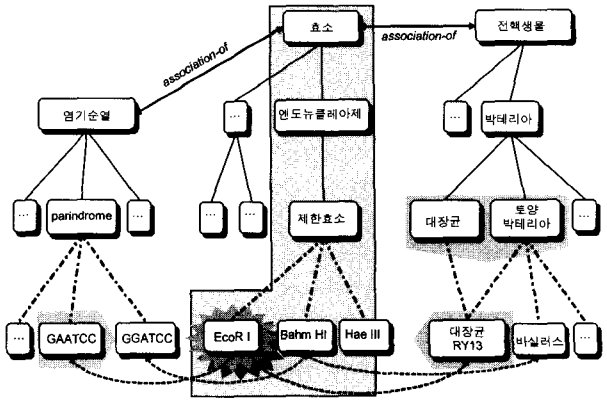
영역 한정질의에서 일반화 연산은 사용자가 이미 알고 있는 구체적인 의미의 객체들을 모두 포괄할 수 있는 일반적인 의미의 상위 객체들을 추론하기 위해 사용된다. 즉, 일반화 연산에 의해 추론된 개념은 모든 주어진 객체에 대한 LUBS로 표현될 수 있다. 이 질의로부터 시스템은 주어진 모든 객체를 하위 객체로 가지는 개념들 중에서 가장 구체적인 개념을 추론한다. 예를 들어, '메티오닌이나 트레오닌과 같은 종류의 아미노산을 보여 달라.'는 질의는 '메티오닌의 특성이나 트레오닌의 특성을 포함하는 아미노산을 찾아달라.'는 의미로 해석될 수 있으며 일반화 연산자를 이용해 이와 같은 질의를 구성할 수 있다. 영역 한정질의는 '메티오닌 OR 트레오닌'이며 제약질의는 없으므로 '*'로 표시하여 [메티오닌 OR 트레오닌, *]으로 나타낸다. (그림 7)(a)에서 보듯이 OOM에서는 일반화 연산자를 이용하여 온톨로지 계층에서 '트레오닌'과 '메티오닌'의 일반화(OR) 관계를 가지는 상위 개념인 '필수아미노산'을 링크로 따라가 추론할 수 있으며 이를 답으로 보여준다.

영역 한정질의에서 구체화(AND) 연산에 의해 추론된 개념은 두 객체에 모두 해당 되는 객체를 의미한다. 따라서 시스템은 동시에 두 객체보다 구체적인 개념 중 가장 일반화된 개념인 GLBS를 구한다. 예를 들어, (그림 7)(b)와 같은 '필수아미노산이면서 염기성 아미노산인 객체를 보여 달라.'는 질의는 [필수아미노산 AND 염기성아미노산, *]로 표현되며, 이 질의는 두 개념 객체에 모두 해당되는 보다 구체적인 객체를 요구하고 있음으로 '필수아미노산'과 '염기성 아미노산' 두 객체에 동시에 *instance-of* 관계에 있는 {리신, 히스티딘}을 추론해 보여준다.

영역 한정질의와 제약질의는 일반화 연산과 구체화 연산

을 이용하여 복합질의로 구성될 수 있다. 예를 들어, '메티오닌이나 트레오닌과 같은 아미노산으로 염기성 아미노산을 찾아달라.'는 질의는 [(메티오닌 OR 트레오닌) AND 염기성 아미노산, *]로 표현되며, '메티오닌이나 트레오닌과 같은 아미노산'은 메티오닌, 트레오닌을 모두 포함하는(OR) 아미노산을 표현하는 것이며 일반화 관계에 의해 '필수아미노산'이 추출되고 그 하위개념들인 {메티오닌, 트레오닌, 발린, 리신, 히스티딘}도 묵시적으로 내포하게 된다. 따라서 {메티오닌, 트레오닌, 발린, 리신, 히스티딘}이면서(AND) '염기성 아미노산을 찾아달라'는 질의로 구체화되며 연산 결과로 {리신, 히스티딘}을 도출한다.

제약질의는 주로 *part-of*, *association-of*, *synonym-of*, *user-defined* 관계의 링크를 항해하여 영역 한정질의의 결과를 더욱 제한하는 역할을 한다. 예를 들어, (그림 8)에서 '토양박테리아인 대장균에서 추출되고 GAATCC를 끊는 엔도뉴클레아제(효소)를 보여 달라'는 질의는 [토양박테리아 AND 대장균 AND GAATCC, 효소]로 표현된다. 즉, 효소의 하위객체 중 '토양박테리아'와 '대장균'의 구체적(AND) 객체와 연관화/집성화 관계에 있고, 'GAATCC'와 관련이 있는 객체를 연결된 링크로 찾아내어 결과를 추론할 수 있다. '토양박테리아 AND 대장균'은 토양박테리아의 하위객체이면서(AND) 대장균의 하위객체인 '대장균 RY13'를 도출함으로, 실제 추론은 '효소'의 하위객체 중 '대장균 RY13'과 'GAATCC'와 연관화/집성화 관계를 갖는 객체를 검색하여 'EcoR I'를 결과로 도출한다.



(그림 8) 토양박테리아 AND 대장균 AND GAATCC, 효소

5. RDF/RDFS 온톨로지 번역기

OOM에서 구축된 온톨로지를 다른 언어로 작성된 온톨로지와 병합할 수 있도록 지원하는 RDF/RDFS 온톨로지 번역기는 다음 세 단계에 의해 구현된다. 우선, 온톨로지 상에서 개념 레벨의 객체와 관계를 RDFS로 표현하기 위한 매핑 구조를 설계한 후, XML 네임스페이스를 정의하고, 온톨로지 생성을 위한 XML 파서를 구성한다.

5.1 매핑 스키마

OOM에서의 객체들과 관계들은 RDF/RDFS 언어 온톨로지에서 클래스(Class)와 자원(Resource)으로 표현된다. OOM에서 객체는 크게 온톨로지 객체, 최상위 개념 객체, 개념 객체, 인스턴스 객체 등 네 가지로 구분된다. 객체 지향 온톨로지에서 특정 도메인을 가리키는 온톨로지 객체는 RDFS의 'Resource'를 상속받아 'OntologyObject' 자원으로 정의하여 모든 객체를 통칭할 수 있도록 한다. 또한 개념 객체 중 최상위 개념 객체만을 별도로 지정하기 위해 RDF/RDFS언어의 온톨로지에서 'TopConcept'를 정의하였으며 이는 'OntologyObject'의 하위 클래스로 기술된다. RDF/RDFS 언어에서 'Concept'은 'TopConcept' 클래스의 하위 클래스로서 OOM에서 개념 객체에 대응된다. 'Instance'는 객체 지향 온톨로지의 개념 객체에 소속된 인스턴스 객체를 나타낸다.

OOM의 *subconcept-of*는 상위 개념 객체와 하위 개념 객체사이의 관계를 나타내며, RDF/RDFS에서 'subConceptOf' 속성과 매핑된다. 'subConceptOf'는 domain과 range의 값으로 'Concept'을 갖는다. OOM 내에서 개념 객체는 자신의 인스턴스 객체들과 *instance-of* 관계를 지니며 이 관계는 RDF/RDFS 형식으로 'instanceOf'로 정의되며 domain은 'Concept', range는 'Instance'를 갖는다. 객체지향 온톨로지 상에서 하나의 객체에 대한 의미가 다른 객체의 일부분을 의미한다면 두 객체 사이의 관계는 *part-of*로 표시되며 이는 다시 RDF/RDFS에서 'partOf' 속성으로 표현할 수 있다. 'partOf'는 각 domain, range의 성격에 따라 'conceptPartOf' 관계와 'instancePartOf' 관계로 나누어 볼 수 있다. 'Concept'들 간에 'partOf' 관계를 지정해주고 싶을 때는 'conceptPartOf', 'Instance'들 간에 'partOf' 관계를 사용하고자 할 때는 'instancePartOf'로 나누어서 사용한다.

두 객체가 서로 연관 관계에 있고 그 의미가 명확하게 정의될 수 없는 경우 두 객체 사이의 관계는 *association-of*로 설정되며 이 관계는 RDF/RDFS의 'associationOf' 관계와 매핑된다. 'associaonOf' 관계 역시 속성의 domain, range의 성격에 따라 'conceptAssociationOf', 'instanceAssociationOf' 관계로 구분하여 사용한다. 두 객체가 서로 같은 의미를 지니지만 다른 이름을 가질 수 있는데 이와 같은 관계는 'synonymOf'로 정의할 수 있다. 객체의 이름에 대한 동의어는 문자열 타입이기 때문에 'synonymOf' 관계의 domain은 'OntologyObject'로, range는 'Literal'로 설정하여 준다.

OOM에서는 보다 정교한 온톨로지의 구축을 위해 사용자 정의 관계를 지원한다. 이 관계의 domain과 range는 각각 'OntologyObject'가 되며, 두 객체 사이에 사용자가 지정하는 관계는 'userDefined' 속성을 상속받아 정의하고 두 객체가 가지고 있는 의미적 연관성을 가장 잘 나타낼 수 있는 관계 이름을 지정해 줄 수 있다. <표 1>은 객체지향 온톨로지에 대응하는 RDF/RDFS의 클래스와 속성을 나타낸 것이다.

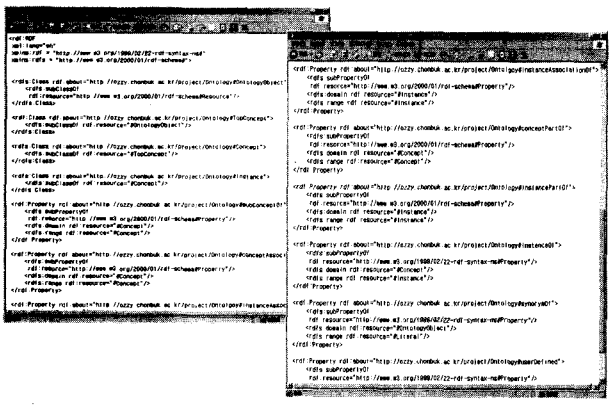
〈표 1〉 객체지향 온톨로지에서 RDF/RDFS 언어 온톨로지로의 매핑 구조

OOT		RDF/RDFS			
				subClassOf	subConceptOf
Object	Ontology object	Class	OntologyObject	Resource	
	Top Concept object		TopConcept		Concept
	Concept object		Concept		TopConcept
	Instance object		Instance		Concept
				Domain	**Range**
Relationship	subconcept-of	Property	subConceptOf	Concept	Concept
	instance-of		instanceOf	Concept	Instance
	part-of		conceptPartOf	Concept	Concept
			instancePartOf	Instance	Instance
	association-of		concept-AssociationOf	Concept	Concept
			instance-AssociationOf	Instance	Instance
	synonym-of		synonymOf	Ontology-Object	Literal
	user-defined (Given by user)		userDefined (Given by user)	Ontology-Object	Ontology-Object

5.2 매핑 스키마를 위한 네임스페이스

RDF/RDFS 네임스페이스는 오직 객체지향 온톨로지의 객체와 관계들에 대해 제한된 의미만을 제공하기 때문에 OOM이 지닌 의미를 완전하게 표현할 수 없다. 그러므로 RDFS로 이들을 적절히 표현하기 위해서 매핑스키마에 따른 'ontology' 네임스페이스를 정의하였다.

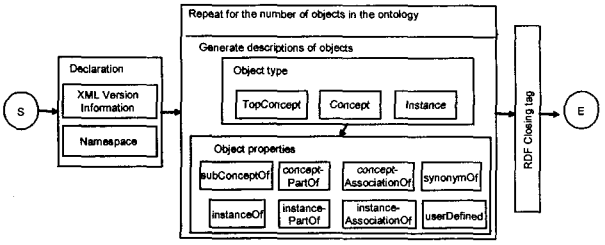
'ontology' 네임스페이스에서 객체와 관계는 각각 클래스 타입과 속성으로 정의된다. 각 속성은 객체지향 온톨로지 스키마에서 정의한 클래스들을 자원(resource)으로 하는 'domain'과 'range'의 제약사항 속성(Constraint Property)를 가진다. (그림 9)는 'ontology'네임스페이스를 지정한 웹 페이지를 보인 것이다.



(그림 9) 'ontology' 네임스페이스 정의

5.3 온톨로지 생성을 위한 XML 파서

객체지향 온톨로지는 (그림 10)과 같이 목적 언어인 RDF/RDFS 코드로 이루어진 온톨로지로 변환된다. 먼저 네임스페이스가 선언되고, 매핑 스키마를 참조하여 변환이 이루어진다.



(그림 10) 온톨로지를 목적언어인 RDF로 생성하기 위한 순서도

URL을 지닌 'rdf' ,'rdfs'의 네임스페이스가 RDF와 RDFS에서 제공하는 개념들을 사용하기 위해 채용된다. 객체지향 온톨로지의 객체와 관계를 지원하기 위해 'ontology'네임스페이스를 선언한다.

```
<rdf : RDF
xml : lang = "en"
xmlns : rdf = "http://www.w3.org/1999/02/22-rdf-syntax
              -ns#"
xmlns : rdfs = "http://www.w3.org/2000/01/rdf-schema#"
xmlns : ontology = "http://ozzy.chonbuk.ac.kr/project/
                    Ontology#">
```

'rdf : Description' 태그는 'about=' 뒤에 나타나는 객체를 설명하기 위한 것이다.

```
<rdf : Description
  rdf : about = "http://ozzy.chonbuk.ac.kr/project/Ontology#
                 object name">
</rdf : Description>
```

'rdf : type'은 'rdf : Description'의 자식 엘리먼트로 올 수 있으며 'TopConcept', 'Concept', 'Instance' 등과 같은 객체의 타입을 정의한다.

```
<rdf : type rdfs : resource = "#object type"/>
```

만약 객체가 다른 객체의 속성을 상속받는다면 이는 다음과 같이 나타낼 수 있다. '#object name'이 있는 자리는 상위 객체의 이름이 올 수 있으며, 'TopConcept'의 경우를 예로 든다면 'OntologyObject'를 상속받으므로 object name은 'OntologyObject'가 된다.

```
<ontology : subConceptOf rdfs : resource = "#object name"/>
```

다음은 Concept 객체에 소속된 Instance 객체를 나타낸 것이다.

```
<ontology : instanceOf rdfs : resource = "#object name"/>
```

'#object name'에 언급된 자원은 'Instance' 객체가 포함된 'Concept' 객체이다.

객체가 다른 객체를 구성하는 관계에 있을 때, 그 두 객체의 type이 'Concept'이라면 'ontology : conceptPartOf' 관계를 지니고 domain과 range가 'Concept'이 된다. 반면 두 객체의 type이 'Instance'라면 'ontology : instancePartOf' 관계를 지니며 이 속성의 domain과 range는 'Instance'로 지정된다. 객체와 'partOf' 관계에 있는 객체는 '#object name' 위치에 이름을 명시하여 두 객체의 관계를 정의한다.

<ontology : conceptPartOf rdfs : resource = "#object name"/>
<ontology : instancePartOf rdfs : resource = "#object name"/>

두 객체가 의미적으로 연관성을 가지고 있지만 그 의미가 매우 다양하여 명확하게 정의될 수 없는 경우 'ontology : conceptAssociationOf', 'ontology : instanceAssociationOf' 관계를 사용할 수 있다. 연관있는 두 객체가 'Concept'인 경우에는'ontology : conceptAssociationOf' 관계로 정의할 수 있으며, 이 속성의 domain과 range는 'Concept'가 된다. 두 객체가 'Instance'인 경우 역시 'ontology : instanceAssociationOf' 관계를 사용하며 domain과 range는 'Instance'를 값으로 갖는다. 객체와 'associationOf' 관계에 있는 객체의 이름을 object name 위치에 명시하여 두 객체 간 관계를 정의한다.

<ontology : conceptAssociationOf rdfs : resource = "#object name"/>
<ontology : instanceAssociationOf rdfs : resource = "#object name"/>

온톨로지 내의 객체가 다른 객체와 특정한 관계에 있을 때, 이 관계의 성격이 명확하다면 그 관계의 이름을 사용자가 지정할 수 있다. 이렇게 정의된 사용자 정의 관계는 고정적으로 지정된 다른 관계들과 함께 객체를 기술하는데 사용할 수 있다. 'user_defined_relation' 부분은 사용자 정의 관계의 이름이 되며, 사용자 정의 관계는 'ontology : user-Defined'속성을 상속받아 정의하게 된다.

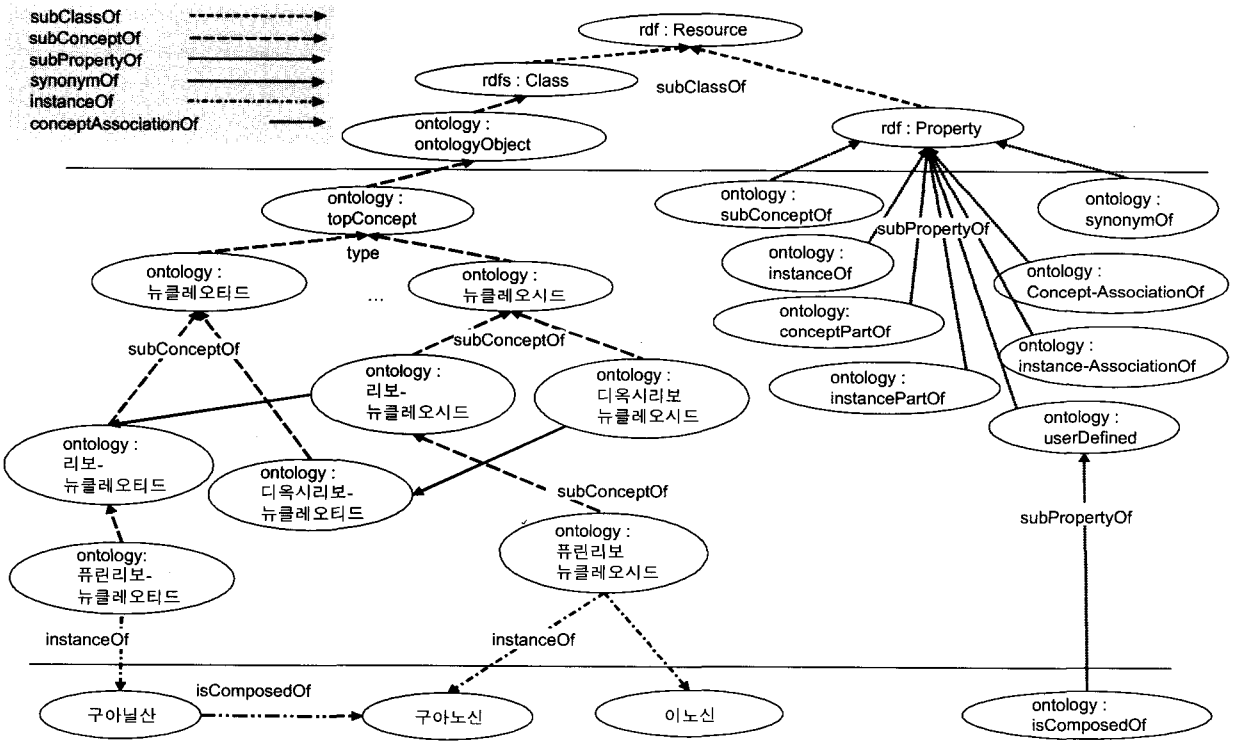
<ontology : user_defined_relation rdfs : resource = "#object name"/>

객체 간의 유의어를 나타내기 위해서 'ontology : synonym-Of' 태그를 사용한다. 유의어가 가질 수 있는 자료 타입은 'Literal'을 갖게 되며 태그 내의 문자열을 값으로 갖게 된다.

<ontology : synonymOf > value </ontology : synonymOf >

5.4 객체 지향 온톨로지의 RDF/RDSF 언어로의 변환 예

(그림 11)은 생명공학 온톨로지 영역 내에 존재하는 '뉴클레오티드' 계층과 '뉴클레오시드' 계층에 대한 온톨로지 정보를 트리 구조의 RDFS로 표현한 것이다. '뉴클레오티드' 계층과 '뉴클레오시드' 계층은 온톨로지 객체를 의미하는 'OntologyObject'를 상속받아 계층 구조를 형성한다. 온톨로지 내에서 사용자 정의 관계로 쓰였던 'isComposedOf'는 RDFS에서 'ontology : userDefined' 관계를 상속받아 정의되었다.



(그림 11) 생명 정보학 온톨로지를 RDF Schema로 표현한 그래프

아래 (그림 12)는 OOM에서 변환된 RDF 코드의 일부분을 보인 것이다. XML 버전 정보가 나온 후 RDF 형식으로 객체지향 온톨로지를 표현하기 위한 네임스페이스가 'rdf : RDF' 태그의 속성 형태로 선언된다. '뉴클레오티드'와 '뉴클레오시드'는 'TopConcept'으로 선언이 되며, Object Name은 'about=' 속성을 이용하여 정의된다. <rdf : subConceptOf >, <rdf : conceptPartOf >, <rdf : conceptAssociationOf >… 등과 같은 다른 객체와의 관계는 'Property'의 하위 속성으로 선언된다. '뉴클레오티드'를 상속받은 '리보뉴클레오티드' 개념 객체는 '뉴클레오시드' 계층의 '리보뉴클레오시드'와 conceptAssociatonOf 관계에 있다. 이들 의미는 'rdf : Description'의 자식 엘리먼트로 'ontology : subConceptOf'와 'ontology : conceptAssociationOf' 관계를 이용해 표현할 수 있다.

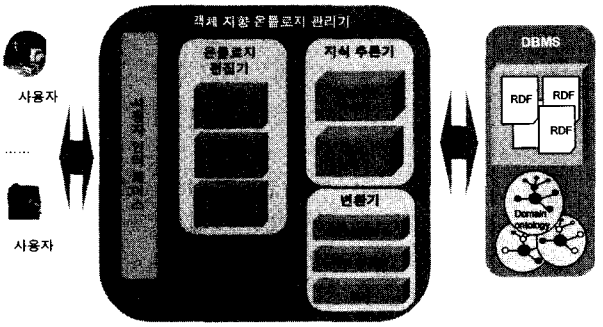
```
<rdf : RDF
.....
xmlns : ontology = "http://ozzy.chonbuk.ac.kr/project/Ontology#"
xml : base = "http://ozzy.chonbuk.ac.kr/project/Gene">

<rdf : Description rdf : about = "http://ozzy.chonbuk.ac.kr/project/
   Gene#Nucleotide">
<rdf : type  rdfs : resource = "http://ozzy.chonbuk.ac.kr/project/
   Ontology#TopConcept"/>
<ontology : subConceptOf rdfs : resource = "#OntologyObject"/>
</rdf : Description>
.....
<rdf : Description rdf : about = "http://ozzy.chonbuk.ac.kr/project/
   Ontology#RiboNucleotide">
<rdf : type  rdfs : resource = "http://ozzy.chonbuk.ac.kr/project/
   Ontology#Concept"/>
<ontology : subConceptOf resource = "#Nucleotide"/>
<ontology : conceptAssociationOf resource = "#RiboNucleocide"/>
</rdf : Description>
.....
<rdf : Description rdf : about = "http://ozzy.chonbuk.ac.kr/project/
Ontology#Guanylacid">
 <rdf : type  rdfs : resource = "http://ozzy.chonbuk.ac.kr/project/
   Ontology#Instance"/>
 <ontology : instanceOf rdfs : resource = "#PurinRiboNucleotide"/>
</rdf : Description>
....
</rdf : RDF>
```

(그림 12) OOM에서 생성한 바이오 온톨로지를 RDF/RDFS로 변환한 결과

6. 객체지향 온톨로지 편집기의 구현

온톨로지 생성기는 Visual C++을 이용하여 윈도 XP 환경에서 구현되었다. (그림 13)은 이러한 시스템 구성 모듈에 대한 기본 구성도이다.



(그림 13) 객체지향 온톨로지 편집기의 시스템 구성도

OOM은 크게 온톨로지 편집기와 지식 추론기, 온톨로지 문서 변환기로 구성되며, 온톨로지 편집기는 객체 관리기, 관계 관리기, 관계 링크 항해기로 이루어져 있고, 지식추론기는 후보 개념 추천기, 질의 실행기로, 온톨로지 문서 변환기는 네임스페이스 정의모듈, 객체 명세모듈, 파서 모듈로 이루어져 있다. 객체 관리기와 관계 관리기는 온톨로지에서 객체의 타입과 관계를 분석한다. 온톨로지 문서 변환기 내

〈표 2〉 RDFS 문서를 'W3C Validation Ser'vice'에 의해 트리플 형태로 나타낸 데이터 모델

Num	Subject	Predicate	Object
1	genid : ARP82187	http://www.w3.org/TR/2000/01/rdf-schema #resource	"http://ozzy.chonbuk.ac.kr/projct/Ontology #TopConcept"@en
2	http://ozzy.chonbuk.ac.kr/projct/Ontology #Nucleotide	http://www.w3.org/1999/02/22-rdf-syntax-ns #type	genid : ARP82187
3	genid : ARP82188	http://www.w3.org/TR/2000/01/rdf-schema #resource	"#OntologyObject"@en
4	http://ozzy.chonbuk.ac.kr/projct/Ontology #Nucleotide	http://ozzy.chonbuk.ac.kr/projct/Ontology #subConceptOf	genid : ARP82190
...	...	...	...
6	http://ktech.chonbuk.ac.kr/ontolgy #Ribonucleotide	http://www.w3.org/1999/02/22-rdf-syntax-ns #type	genid : ARP296310
7	http://ktech.chonbuk.ac.kr/ontology#Plasmids	http://www.w3.org/TR/2000/01/rdf-schema #resource	"#OntologyObject"@en
...	...	...	...
16	http://ozzy.chonbuk.ac.kr/projct/Ontology #Ribonucleotide	http://ozzy.chonbuk.ac.kr/projct/Ontology #conceptAssociationOf	http://www.w3.org/RDF/Validator/run/106785 0586702#Ribonucleocide

의 객체 기술 모듈은 *subconcept-of, instance-of*와 같은 수직적 관계와 *part-of, association-of, synonym-of, user-defined* 같은 수평적 관계를 RDF/RDFS 문법으로 변환한다. 이러한 모듈들은 XML 파서와 연동하여 RDF 문서를 생성한다.

'W3C RDF Validation Service'[19]는 온톨로지로 변환된 결과의 RDF 문법에 대한 적절성을 진단 할 수 있으며 <표 2>는 데이터 모델을 트리플 형식으로 표현하고 있다. <표 2>에서 트리플은 subject, predicate, object 등으로 구성되어 있으며, 'genid'는 'ontology' 네임스페이스에 정의된 객체를 나타내기 위해 사용되는 id이다. <표 2>의 num 1에서 'genid' 값 ARP82187는 'TopConcept'임를 나타내며, num 2에서 'Nucleotide'의 type은 'genid'가 ARP82187인 'TopConcept'임을 알 수 있다.

7. 결론 및 향후연구

본 논문에서는 바이오 관련 온톨로지를 OOM의 가시적 개념화와 관계 기반 추론을 이용하여 구축하였다. OOM은 직관적인 사용자 인터페이스를 제공하고, 다양한 관계를 표현할 수 있으므로 보다 표현력 있는 온톨로지의 구축과 관리가 용이하고, 관계 링크의 추론 기능을 통해 바이오 관련 온톨로지 용어 사이에 내포된 복잡한 지식을 쉽게 이해할 수 있다. 즉, OOM은 기존의 온톨로지 툴에 비해 단순하면서도 의미적 표현력이 뛰어나 바이오 온톨로지와 같은 복잡한 응용의 온톨로지도 개념적으로 쉬운 방식으로 모델링 할 수 있다. 또한 본 논문에서 구현된 온톨로지 번역기는 OOM에 의해 구축된 온톨로지를 온톨로지 표준언어인 RDF/RDFS로 변환함으로써 다른 온톨로지 구축 툴에서 구축된 온톨로지와 쉽게 병합 될 수 있으며, 따라서 보다 많은 지식을 표현하는 온톨로지를 구축할 수 있다.

본 연구에서는 변환 목적이 되는 온톨로지 언어로 RDF/RDFS를 지정하였지만 향후 연구로 목적 언어의 대상을 OWL과 같은 다른 온톨로지 언어로 확장할 수 있다. 객체지향 온톨로지를 현재 많이 채택되고 있는 여러 온톨로지 언어로 변환할 수 있다면 다른 온톨로지와의 병합을 통해 다양한 응용에 이용될 수 있을 것이다. 또 다른 향후 연구로서 생명 정보 데이터로부터 시맨틱을 유추하는데 객체 지향 온톨로지가 이용되는 연구를 들 수 있다.

참 고 문 헌

[1] R. Stevens, C. Goble, I. Horrocks and S. Bechhofer, "Building a Bio-ontology Using OIL," IEEE Transactions on Information Technology in Biomedicine, Vol.6, pp.135-141, 2002.

[2] P. Lambrix, M. Habbouche and M. Perez, "Evaluation of ontology development tools for bioinformatics," Bioinformatics, Vol.19, pp.1564-1571, 2003.

[3] C. Rosse and J. L. V. Megino, "A reference ontology for bioinformatics : the Foundational Model of Anatomy," Journal of Biomedical Informatics, Inpress, 2003.

[4] P. D. Karp, "An Ontology for Biological function based on molecular interactions," Bioinformatics, Vol.16, pp.269-285, 2000.

[5] L. J. Jensen, R. Gupta, H. -H. Starfeldt and S. Brunak, "Prediction of Human Protein Function According to Gene Ontology Categories," Bioinformatics, Vol.19, pp.653-642, 2003.

[6] J. D. Westbrook and P. E. Bourne, "STAR/mmCIF : Anontology for macromolecular structure," Bioinformatics Ontology, Vol.16, pp.159-168, 1999.

[7] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer and D. Wenke, "OntoEdit : Collaborative Ontology Development for the Semantic Web," in : Proceedings of the first International Semantic Web Conference, 2002.

[8] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen, "Creating Semantic Web Contents with Protege-2000," IEEE Intelligent Systems, Vol.16, No.2, pp.60-71, 2001.

[9] S. Bechhofer, I. Horrocks, C. Goble and R. Stevens, "OilEd : a Reason-able Ontology Editor for the Semantic Web," Proceedings of KI2001, LNAI Vol.2174, pp.396-408, 2001.

[10] J. C. Arprez, O. Corcho, M. Fernndez-Lpez and A. Gmez-Prez, 'WebODE in a nutshell,' AI Magazine, Vol.24, No.3, 2003.

[11] A. Farquhar, R. Fikes and J. Rice, "The Ontolingua Server : A Tool for Collaborative Ontology Construction," Int. Journal of Human-Computer Studies, pp.707-728, 1997.

[12] A. Gomez-Perez, "Deliverable 1.3 : A survey on ontology tools," OntoWeb Ontology-based information exchange for knowledge management and electronic commerce IST- 2000-29243, 2002.

[13] A. Gupta, B. Ludascher and R. W. Moore, "Ontology Services for Curriculum Development in NSDL," ACM/IEEE-cs Joint Conference on Digital Libraries, 2002.

[14] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson and M. A. Musen, "Creating Semantic Web Contents with Protege-2000," IEEE Intelligent System, Vol.2, pp.60-71, 2001.

[15] A. Moreno and C. Perez, "From Text to Ontology : Extraction and Representation of Conceptual Information," Proc. of Conf TIA, pp.11-21, 2001.

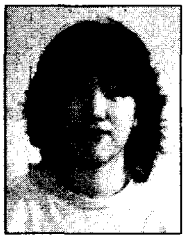
[16] I. Yeh, P. D. Karp, N. F. Noy and R. B. Altman, "Knowledge acquisition, consistency checking and concurrency control for Gene Ontology(GO)," Bioinformatics, Vol.19, No.3, pp.241-248, 2003.

[17] R. Stevens, I. Horrocks, C. Boble and S. Bechhofer, "Buil-

ding a Reason-able Bioinformatics Ontology Using OIL," IJCAI'01 Workshop on Ontology and Information Sharing, pp.81-90, 2003.

[18] J. H. Choi, J. D. Yang and D. G. Lee, "An object-based Approach to Managing Domain Specific Thesauri : Semi-automatic Thesaurus Construction and Query-based Browsing," International Journal of Software Engineering & Knowledge Engineering, Vol.10, No.4, pp.1-27, 2002.

[19] RDF Validation Service : http://www.w3.org/RDF/Validator/, 2001.



양 경 아

e-mail : kayang@chonbuk.ac.kr

2001년 한국교원대학교 컴퓨터교육학과 (학사)

2003년 전북대학교 컴퓨터정보학과(석사)

2003년~현재 전북대학교 컴퓨터통계정보학과 박사과정

관심분야 : 온톨로지, 시맨틱 웹, 바이오 인포매틱스, 정보검색

양 형 정

e-mail : hjyang@cs.cmu.edu

1991년 전북대학교 전산통계학과(학사)

1993년 전북대학교 전산통계학과(석사)

1998년 전북대학교 전산통계학과(박사)

2000년 동신대학교 컴퓨터 응용학과군 전임강사

2003년~현재 카네기멜론대학 컴퓨터과학과 포스트닥터 연구원

관심분야 : OODBs, 이미지/비디오정보검색, 온톨로지, 시맨틱 웹

양 재 동

e-mail : jdyang@chonbuk.ac.kr

1983년 서울대학교 컴퓨터공학과(학사)

1985년 한국과학기술원 전산학과(석사)

1991년 한국과학기술원 전산학과(박사)

1995년~1996년 Univer.of Florida, Visiting Scholar

현재 전북대학교 컴퓨터과학과 교수

관심분야 : OODBs, Expert System, CASE, 온톨로지