DOI: 10.3745/KIPSTD.2008.15-D.4.557

개념계층구조에 기반한 온톨로지 분석 및 오류검출도구

황 석 형†

요 약

온톨로지는 시멘틱 웹의 핵심요소로서, 도메인의 공유지식을 개념화하여 정형적으로 표현한 것이다. 잘 정의된 온톨로지를 사용함으로써 상호운용성을 기반으로 하는 시멘틱 웹 분야의 정보시스템 품질을 향상시킬 수 있다. 그러나, 실제로는 오류를 포함하지 않는 고품질의 온톨로지를 개발하는 것은 수월하지 않다. 따라서, 온톨로지 설계기법과 더불어서, 온톨로지에 내재된 에러를 검출하는 기능을 지원하는 다양한 분석도구/기법들은 온톨로지 개발자들에게 많은 도움이 될 수 있다. 본 논문에서는, 형식개념분석기법을 기반으로 온톨로지의 핵심요소들을 분석하는 기법을 제안하고, 온톨로지에 내재되어 있는 오류를 검출하기 위한 지원도구를 개발하였다. 본 연구결과는, 기존의 온톨로지를 수정보완할 경우의 지침으로서 뿐만아니라, 고품질의 온톨로지를 개발할 수 있는 유용한 도구로서 이용될 수 있다.

키워드 : 시맨틱 웹, 온톨로지, 개념계층구조, 형식개념분석

An ontology analysis and error detection tool based on concept hierarchy structures

Suk-Hyung Hwang†

ABSTRACT

An ontology as the core element of Semantic Web is a formal specification of a conceptualization of shared domain knowledge. The use of well-defined ontologies can increase the quality of interoperable information systems in the area of Semantic Web. However, in practice, it is not easy to develop high-quality ontologies which have no errors. Therefore, with methodologies for ontology design, various methods or tools for ontology analysis supporting for error-detection might be very helpful for ontology developers. In this paper, we propose a novel approach for analyzing the core constructs of ontology based on the Formal Concept Analysis and develop a tool that supports error-checking ontologies. Our approach can serve not only as a guidance to modify the existing ontologies, but also as a valuable tool in developing high-quality ontologies.

Key Words : Semantic Web, Ontology, Concept Hierarchy Structure, Formal Concept Analysis

1. 서 론

온톨로지는, 시멘틱 웹, 의료정보, 바이오정보, 인공지능, 에이전트, 유비쿼터스 컴퓨팅, 전자상거래 등과 같은 다양한 분야에서 지식공유 및 지식의 재사용을 위해 구축된 정형화된 지식 표현의 형태이다. 특히, 시멘틱 웹분야에서의 온톨로지는 상호운용성의 토대가 되는 핵심요소로 자리 잡고 있다[1, 2]. 현재, 온톨로지를 기술할 수 있는 다양한 언어들과 온톨로지 구축지원도구들이 개발되어, 관련 전문가들의 의견과 지식, 경험 등을 토대로 온톨로지 개발자들이 협업하여 온톨로지를 구축할 수 있는 환경을 제공하고 있다[3].

그러나, 최근의 연구결과[4, 5]에 따르면, 구축하려는 온톨로지의 도메인으로부터 온톨로지의 구성요소를 추출하기 매우 어려우며, 이러한 온톨로지의 구성요소를 가지고 온톨로지를 모델링 하는 작업 또한 쉽지 않다. 특히, 대다수의 온톨로지개발관련 프로젝트에서 온톨로지 개발자들이 개발방법론을 적용하지 않고, 자신의 경험에만 의존하여 온톨로지를 수작업으로 구축하고 있는 것으로 보고되고 있다. 또한, 도메인이 광범위하고 복잡하며 방대한 정보를 대상으로 온톨로지를 개발하는 경우, 다종다양한 오류들이 포함될 수 있다. 따라서, 실제로 오류를 포함하지 않는 고품질의 온톨로지를 구축하는 작업은 수월하지 않다. 이와같은 상황에서는, 다양한 측면에서 온톨로지에 대한 분석 및 오류를 검출하기 위한 제반 기법들과 지원도구들을 제공함으로써, 온톨로지 개발자들에게는 고품질의 온톨로지를 개발하고 수정보완할 수 있는 유용한 지침 및 도구로서의 역할을 할 수 있다

우수한 품질의 시멘틱 웹 시스템을 개발, 운용하기 위해

※ 본 논문은 2007학년도 선문대학교 교수연구년제도의 지원을 받아서 수행되었음.
† 종신회원 : 선문대학교 컴퓨터공학부 부교수
논문접수 : 2008년 1월 15일
수 정 일 : 1차 2008년 3월 19일, 2차 2008년 4월 8일
심사완료 : 2008년 4월 10일

서는, 시멘틱 웹의 기반이 되는 온톨로지에 대한 보다 체계적인 개발과 이를 뒷받침하기 위한 평가 및 분석 기능이 요구되고 있으며, 아래와같이 온톨로지 평가 및 분석을 지원하는 다양한 도구들[6~11]이 개발되고 있다.

- OntoKBEval[6]은 기술논리(Description Logic)를 이용하여 정성적인 방법과 정량적인 방법으로 OWL온톨로지를 평가하는 도구이다. OntoKBEval은 개념과 개념들간의 관계(Roles)의 개수, Children의 평균개수, 그리고 계층구조에서 이와같은 요소들의 분포 등 온톨로지의 개요에 대한 정보와 더불어서, TBox와 ABox의 기술논리 관점으로 OWL-DL온톨로지의 구조를 설명하는 계층구조를 제공한다.
- CleanOnto[7]는 OWL온톨로지의 구문상의 일관성을 평가하기 위한 도구로서, WordNet(http://wordnet.princeton.edu)의 정의들을 사용하여 온톨로지의 분류학적인 관계를 평가한다.
- OntoEdit[8]의 플러그인 기능으로 제공되는 OntoAnalyzer[9]는 온톨로지의 일관성과 속성들의 일치성에 초점을 맞춘 온톨로지 평가도구이다. OntoEdit를 사용하여 개발된 온톨로지에 대해서, OntoAnalyzer를 사용하여 자동적으로 분석한다.
- OWLDoc[10]은 OWL로 작성된 온톨로지의 소스코드로부터 온톨로지의 구조적인 정보들(클래스, 프로퍼티, individual)을 JavaDoc과 유사한 html 문서형태로 변환하여준다. OWLDoc은 Protégé의 확장기능으로 구현되어 있으며, 웹브라우저를 이용하여 온톨로지의 구조와 정보를 수월하게 파악할 수 있다.
- FCAViewTab[11]에서는 Protégé[12]에서 작성된 온톨로지에 대하여, 형식개념분석기법을 이용하여 각종 정보들을 개념단위로 추출하여 격자형태로 가시화하는 기능을 제공한다. 그러나, FCAViewTab은 온톨로지의 인스턴스와 boolean type의 슬롯들에 대한 분석정보를 가시화하는 것으로 한정되어 있기 때문에 온톨로지의 전체적인 구조파악 및 분석에는 한계가 있다.

이와같은 기존의 온톨로지 분석도구들은, 구문에러에 대한 검출과 더불어 의미정보를 가미한 형태의 온톨로지 분석기능들을 제공하고 있다. 본 논문에서는, 보다 품질좋은 온톨로지의 개발을 지원하기 위하여, 형식개념분석기법(Formal Concept Analysis)을 토대로 하는 온톨로지 분석 및 오류검출 기법을 제안한다. 형식개념분석기법은 개념격자(Concept Lattice)라는 수학적 개념계층구조화 모델을 기반으로 하는 데이터분석기법으로서, 개념적인 데이터분석에 의한 지식추출 및 포섭관계를 나타내는 개념계층의 구조화가 가능하다. 따라서 본 논문에서는 이와 같은 형식개념분석기법을 확장하여 온톨로지의 구성요소들을 추출, 분석하기위한 모델을 정의하고, 온톨로지 소스코드로부터 온톨로지를 분석하여, 포함된 오류를 자동으로 검출하기위한 방법을 제안하고 이에대한 지원도구를 개발하고자 한다.

본 논문은 다음과 같이 구성되어 있다. 제2장에서는 온톨로지를 표현하기 위한 기본적인 구성요소들과 형식개념분석기법을 소개한다. 제3장에서는 본 연구에서 개발된 온톨로지 분석 및 오류검출 도구에 대해서 설명하고, 제4장에서는 분석 및 오류검출에 관한 실험사례를 설명한다. 그리고, 제5장에서는 결론 및 향후과제를 논의한다.

2. 온톨로지의 요소와 형식개념분석

2.1 온톨로지의 요소

온톨로지는 공유된 개념화에 대한 정형화되고 명시적으로 기술해놓은 지식모델을 말한다. 즉, 특정 도메인의 개념들을 정의하고, 그들 사이의 관계를 계층적으로 표현하고 있으며, 추가적으로 추론 규칙이 포함되어 있다[1]. 온톨로지를 표현하기 위해 스키마와 구문구조 등을 정의한 언어가 온톨로지 언어이며, XML 기반의 XOL, OML, SHOE와 W3C에서 제정한 RDF와 OIL, DAML, DAML+OIL, OWL 등이 있다[3,4]. 특히, RDF[13]는 W3C에서 발표한 메타데이터의 기술과 교환을 위한 표준이며, OWL[14]은 RDF와 RDF Schema의 한계를 보완하고자 개발된 언어로서, 풍부한 어휘와 형식적 의미론을 포함하고 있기 때문에 기계해석이 가능한 웹 콘텐츠를 저작하는 경우, XML이나 RDF보다 우수하여, 현재 온톨로지 언어의 표준으로 자리잡고 있다.

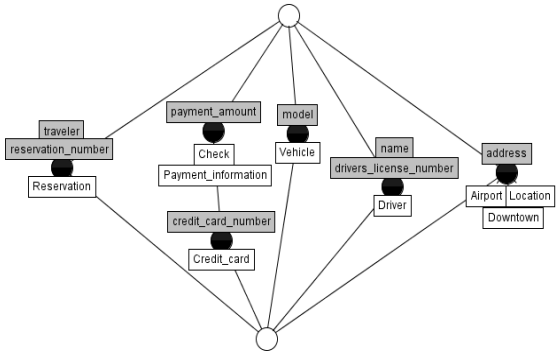
온톨로지들은 다양한 언어로 표현되고 있음에도 불구하고 구조적으로 많은 유사성을 보이고 있다. 대부분의 온톨로지들은 individuals(instances), classes(개념들), 속성(attributes), 관계(relations)를 기술하고 있다. 예를들어, Protégé와같은 도구에 의해 작성된 온톨로지의 각 구성요소들은 OWL언어를 기반으로, owl:Class키워드를 사용하여 클래스들을 정의하고, 그들 사이의 is-a관계를 subClassOf를 이용하여 정의될 수 있다. OWL에는 2종류의 property(datatype property, object property)가 제공되며, datatype property는 문자열이나 숫자타입의 range속성을, 한편으로 object property는 individual을 range값으로 갖는 프로퍼티를 나타낸다. 또한, individual은 특정 클래스에 속하는 인스턴스로서 해당 클래스가 갖는 속성들의 실제 데이터 값을 포함하고 있다.

2.2 형식개념분석

형식개념분석기법[15]은 개념격자라는 수학적 이론을 기반으로 하는 데이터클러스터링 기법의 일종으로서, 개념적인 데이터분석과 지식기반 정보처리분야의 제반문제들에 대한 수학적 해법을 제공하고 있다. 형식개념분석의 기본이 되는 구조 Formal context K=(G, M, I)는 객체들(Objects)의 집합 G와 속성들(Attributes)의 집합 M, 그리고 G와 M 사이의 이항관계 $I \subseteq G \times M$로 구성된다. 어떤 객체 g가 속성 m을 가지고 있을 경우, gIm 또는 $(g, m) \in I$ 로 나타내며, g는 m을 갖는다는 것을 의미한다. 표1은 Protégé에서 제공하는 car reservation온톨로지의 클래스와 속성들을 각각 객체집합G와 속성집합M으로 하여 구성한 context의 예이다.

〈표 1〉 car reservation에 관한 context

	reservation_number	credit_card_number	payment_amount	drivers_license_number	name	model	address	traveler
Airport							×	
Location							×	
Driver				×	×			
Reservation	×							×
Check			×					
Payment_information			×					
Vehicle						×		
Credit_card		×	×					
Downtown							×	



(그림 1) 〈표 1〉에 대한 context 의 개념격자

context K=(G, M, I)에 대하여, O⊆G, A⊆M일 때, intent(O) =A ∧ extent(A)=O를 만족하는 (O, A)을 형식개념(formal concept) 또는 간단히 개념이라고 한다. 단, intent(O):={a∈M|∀o∈O:(o,a)∈I}, extent(A) := {o∈G|∀a∈A:(o, a)∈I}. 임의의 O⊆G에 대하여, intent(O)에 의해 O의 모든 객체들이 공통적으로 갖는 속성들의 집합을 구할 수 있고, 또한, 임의의 A⊆M에, extent(A)에 의해 A의 속성들을 갖는 객체들의 집합을 구할 수 있다. 예를 들면, 위의 <표 1>의 context에서, O = {Airport, Location, Downtown}에 대하여, intent(O) = {address}이고, A={reservation_number, traveler}에 대하여, extent(A) = {Reservation}이다. 즉, 각 개념들은 (O, A)와 같은 형태의 쌍(pair)으로 정의되며 특히, 객체집합O는 속성집합A의 extent이며, 동시에 속성집합A는 객체집합O의 intent가 된다. 임의의 개념 (O_1, A_1), (O_2, A_2)에 대하여, $O_1 \subseteq O_2$ $(\Leftrightarrow A_1 \supseteq A_2)$라면 개념 (O_1, A_1), (O_2, A_2)은 상위-하위개념관계이며 $(O_1, A_1) \leq (O_2, A_2)$과 같이 표현한다. context K=(G, M, I)로부터 만들어진 모든 개념들 간의 상위-하위개념관계≤는 일종의 반순서관계에 해당하며, 개념들과 그들 사이의 상위-하위개념관계에 의해 만들어진 계층적 개념구조를 개념격자(Concept Lattice)라고 부르며, (그림 1)과 같이 Hasse Diagram을 사용하여 가시화할 수 있다.

개념격자를 나타낸 Hasse Diagram에서는, 각 개념들과 이들 사이의 상하위관계가 링크에 의해 표시되며, 특히, 개념들 간의 링크에 의해 만들어지는 경로에 의해 상위개념으로부터 하위개념으로 속성들이 상속되며, 하위개념으로부터 상위개념으로 해당 객체들이 전파된다. 예를 들어, (그림 1)에서 Credit_card는 자신만의 고유속성으로서 credit_card_number를 가지며, 상위개념들로부터 payment_amount속성을 상속받는다. 한편 payment_amount를 속성으로 갖는 객체로서는 Check, Payment_information, 그리고 Credit_card가 됨을 알 수 있다. 이와 같은 방법을 사용함으로써, 주어진 문제영역의 객체들과 이들이 갖는 속성들을 context형태로 파악하여, 개념들을 추출하여 격자구조화하여 나타냄으로써, 도메인 내의 개념들을 분류하고 체계화할 수 있는 계층화된 개념구조를 수월하게 구축할 수 있다.

형식개념분석기법에서는 여러 가지 다양한 값을 가지는 속성들을 포함한 context를 Many-valued context라고 부르며, Many-valued context K=(G, M, W, I)는 객체들(Objects)의 집합 G와 속성들(Many-valued Attributes)의 집합 M, 속성의 값 W, 그리고 G와 M과 W사이의 관계 I⊆G×M×W로 구성된다. 즉, G, M 그리고 W는 각각 객체들과 각 객체들이 가질 수 있는 속성들, 그리고 그 속성의 값들을 나타낸다. 또한, 어떤 객체g가 속성m을 가지고 있고 그 속성의 값이 w인 경우, (g, m, w)∈I 또는 m(g)=w로 나타낸다. Many-valued context은 표1과 같은 One-valued context를 나타내는 테이블로 나타낼 수 있으며, 테이블의 각 셀에는 "X"표시 대신 해당 객체가 갖는 속성들의 값을 표시한다.

형식개념분석기법은 관심대상영역의 데이터를 객체들과 속성들로 구성되는 context형태로 입력받아서, 공통속성을 갖는 객체들의 집합과 객체들의 속성집합으로 구성되는 형식개념을 추출하고, 개념들사이의 상하위관계를 파악하여, 개념격자라는 계층구조를 구축함으로써, 입력데이터에 내재되었으나 명시적으로는 드러나지 않았던 개념들과 개념계층구조를 파악하여 숨겨진 유용한 지식을 발견해 낼 수 있는 데이터분석기법이다.

본 연구에서는, 온톨로지의 주요요소들을 분석하고 내재된 오류를 검출하기 위하여, 형식개념분석기법을 사용하여 주어진 온톨로지 소스코드로부터 온톨로지의 핵심기본요소

들과 요소들 사이의 다양한 관계 및 관련 상세정보들을 추출하고, 개념들과 개념격자를 토대로 개념계층구조를 구축한다. 이와같이 구축된 각 개념계층구조들에 대해서, 본 논문에서 정의한 온톨로지 오류들에 대한 정의(정의 3~5)들을 토대로 개발한 오류검출모듈(그림 4)을 적용하여 subClassOf오류, 프로퍼티오류, 그리고 Individual오류 등을 검출한다.

따라서, 형식개념분석기법의 "개념"에는, 입력된 온톨로지의 주요요소들에 대한 관련상세정보가 추출되어 들어가 있으며, 이러한 개념들과 그들 사이의 상하위관계들에 의해 구성되는 개념계층구조에는 입력된 온톨로지에 대한 각종 핵심요소들의 정보와 더불어, 명시적으로 드러나지 않았던 정보들(공통속성들, 공통개념들, 개념계층구조 등)이 포함되어 있다. 본 연구에서는 이러한 개념계층구조를 토대로 온톨로지에 내포된 각종 정보를 분석하고 오류를 검출하는 기법과 도구를 개발하였다.

3. 온톨로지 분석 및 오류검출 도구의 개발

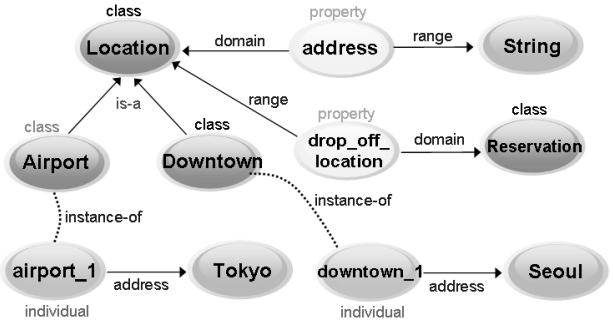
3.1 Ontology Context Model

형식개념분석기법을 이용하여 온톨로지를 분석하기 위해서는, 온톨로지를 구성하는 각 요소들의 정보를 Context형태로 작성하여 입력해야 한다. 그러나 온톨로지의 정보는 매우 복잡하고, 방대하기 때문에 하나의 Context로 온톨로지 구성요소들의 다종다양한 정보를 적절하게 표현하기 어렵다. 따라서 본 연구에서는, 온톨로지의 핵심기본요소들을 표현할 수 있는 Ontology Context Model을 정의하였다.

Ontology Context Model은 총 3개의 One-valued Context와 2개의 Many-valued Context로 구성되어 있으며, 온톨로지의 핵심요소들(Class, Is-a관계, Property, Individual등)과 요소들 사이에 다양한 관계(Is-a관계, Instance-of, Has-property, Association) 및 관련정보(Property, individual의 상세정보)들을 표현하고 있다. Ontology Context Model의 정형화된 정의는 아래와 같다.

[정의 1] Ontology Context Model $CF = \{S_K, B_J, I_J\}$는 온톨로지의 주요요소들의 집합 S_K와 온톨로지의 요소들 사이의 이항관계들의 집합 B_J , 그리고 삼항관계 I_J 로 구성된다.

- $S_K = \{ S_c , S_p , S_i , S_d , S_v , S_r \}$
  S_c : 온톨로지의 클래스들의 집합
  S_p : 온톨로지의 프로퍼티들의 집합
  S_i : 온톨로지의 individual들의 집합
  S_d : 온톨로지의 primitive data type들의 집합
  S_v : 온톨로지의 primitive data value들의 집합
  S_r = {domain, range}
- $B_J = \{ B_{cc} , B_{cp} , B_{ci} \}$
- $B_{cc} : \tau(B_{cc}) = (S_c , S_c) , B_{cc} \subseteq S_c \times S_c$
  (클래스들 사이의 Is-a관계)



(그림 2) car reservation 온톨로지의 일부정보에 대한 유방향 그래프 표현

- $B_{cp} : \tau(B_{cp}) = (S_c , S_p) , B_{cp} \subseteq S_c \times S_p$
  (클래스들과 프로퍼티들 사이의 Has-property관계)
- $B_{ci} : \tau(B_{ci}) = (S_c , S_i) , B_{ci} \subseteq S_c \times S_i$
  (클래스들과 individual들 사이의 instance-of관계)
- $I_J = \{ I_{pr} , I_{ip} \}$
- $I_{pr} : \gamma(I_{pr}) = (S_p, S_r , S_c \cup S_d) , B_{pr} \subseteq S_p \times S_r \times (S_c \cup S_d)$
  (프로퍼티들의domain, range정보)
- $I_{ip} : \gamma(I_{ip}) = (S_i , S_p, S_i \cup S_v) , B_{ip} \subseteq S_c \times S_p \times (S_i \cup S_v)$
  (individual들의 실제 데이터정보)

단, $\tau(J) = (k1, k2)$함수는 이항관계 집합을 입력으로하여, 이항관계를 만족하는 k1, k2집합을 출력하고, $\gamma(J) = (k1, k2, k3)$함수는 삼항관계 집합을 입력으로하여, 삼항관계를 만족하는 k1, k2, k3집합을 출력한다. ■

위와 같은 온톨로지의 핵심요소들의 집합과 핵심요소들 사이의 이항관계 및 삼항관계 집합을 이용하여 온톨로지의 주요요소들과 그들 사이의 관계를 표현할 수 있다. 또한, Ontology Context Model은 아래와 같이 5개의 Context로 이해할 수 있다.

[정의 2] Ontology Context Model *CF*는 다음과 같이 3개의 One-valued Context집합 OVC_J와 2개의 Many-valued Context집합 MVC_J로 정의된다.

- $OVC_J = \{(S_m, S_n, B_j) \mid S_m, S_n \in S_K, B_j \in B_J \text{ and } B_j \subseteq S \times S_n \text{ if } \tau(B_j)=(S_m,S_n)\}$
  $OVC_{cc}=\{(S_c, S_c, B_{cc}) \mid S_c \in S_K, B_{cc} \in B_J \text{ if } \tau(B_{cc})=(S_c, S_c)\}$
  $OVC_{cp}=\{(S_c, S_p, B_{cp}) \mid S_c,S_p \in S_K, B_{cp} \in B_J \text{ if } \tau(B_{cp})=(S_c, S_p)\}$
  $OVC_{ci}=\{(S_c, S_i, B_{ci}) \mid S_c, S_i \in S_K, B_{ci} \in B_J \text{ if } \tau(B_{ci})=(S_c, S_i)\}$
- $MVC_J = \{(S_m, S_n, S_o, I_j) \mid S_m, S_n, S_o \in S_K, I_j \in I_J \text{ and } I_j \subseteq S_m \times S_n \times S_o \text{ if } \gamma(I_j) = (S_m, S_n, S_o)\}$
  $MVC_{pr}=\{(S_p, S_r, (S_c \cup S_d), I_{pr}) \mid S_p, S_r, S_c, S_d \in S_K, I_{pr} \in I_J \text{ if } \gamma(I_{pr}) = (S_p, S_r, (S_c \cup S_d))\}$
  $MVC_{ip}=\{(S_i, S_p, (S_i \cup S_v), I_{ip}) \mid S_i, S_p, S_v \in S_K,$

$I_{ip} \in I_J$ if $\gamma(I_{ip}) = (S_i, S_p, (S_i \cup S_v))\}$ ■

(그림 2)는 Protégé에서 제공하는 car reservation온톨로지의 일부정보를 유방향그래프 형태로 표현하고 있다. car reservation온톨로지는 4개의 클래스(Location, Airport, Downtown, Reservation)들과 2개의 프로퍼티(address, drop_off_location)들, 그리고 2개의 individual(airport_1, downtown_1)의 요소를 가지고 있다. 또한, 각 요소들 사이의 다양한 관계(is-a relationship, instance-of, has-property등)들을 표현하고 있다. 이러한 온톨로지 정보는 다음과 같은 Ontology Context Model로 나타낼 수 있다.

〈표 2〉 is-a정보에 대한 Context

	Location	Airport	Downtown	Reservation
Location		X	X	
Airport				
Downtown				
Reservation				

〈표 3〉 has-property정보에 대한 Context

	address	drop_off_location
Location	X	
Airport	X	
Downtown	X	
Reservation		X

〈표 4〉 instance-of정보에 대한 Context

	airport_1	downtown_1
Location		
Airport	X	
Downtown		X
Reservation		

〈표 5〉 프로퍼티에 대한 Context

	domain	range
address	Location	String
drop_off_location	Reservation	Location

〈표 6〉 individual에 대한 Context

	address	drop_off_location
airport_1	Tokyo	
downtown_1	Seoul	

Ontology Context Model $CF = \{S_K, B_J, I_J\}$

- $S_K = \{ S_c, S_p, S_i, S_d, S_v, S_r\}$
  - $S_c = \{$ Location, Airport, Downtown, Reservation $\}$
  - $S_p = \{$ address, drop_off_location $\}$
  - $S_i = \{$ airport_1, downtown_1 $\}$
  - S_d : integer, string, boolean, float등 온톨로지 언어에서 지원하는 데이터타입의 집합
  - S_v : 온톨로지 언어에서 지원하는 데이터타입의 실제 데이터들의 집합
  - $S_r = \{$ domain, range $\}$
- $B_J = \{ B_{cc}, B_{cp}, B_{ci} \}$
  - $B_{cc} = \{$(Location, Airport), (Location, Downtown) $\}$
  - $B_{cp} = \{$(Location, address), (Airport, address), (Downtown, address), (Reservation, drop_off_location) $\}$
  - $B_{ci} = \{$ (Airport, airport_1), (Downtown, downtown_1) $\}$
- $I_J = \{ I_{pr}, I_{ip} \}$
  - $I_{pr} = \{$(address, domain, Location), (address, range, String), (drop_off_location, domain, Reservation), (drop_off_location, range, Location) $\}$
  - $I_{ip} = \{$ (airport_1, address, Tokyo), (downtown_1, address, Seoul) $\}$

한편, (그림 2)의 온톨로지는 정의 2를 기반으로하여 <표 2>~<표 6>과 같은 5개의 context로 나타낼 수 있다. 이와 같은 Ontology Context Model을 기반으로, (그림 4)에 나타낸바와 같이, 본 연구에서는 방대하고 복잡한 온톨로지의 구성요소에 관한 정보를 입력으로 하고, 형식개념분석기법을 적용하여 온톨로지에 내재되어 있는 개념들과 개념들로 구성되는 계층구조를 파악함으로써, 온톨로지에 대한 개념분석이 가능하다.

구체적으로 설명하면, 본 연구에서 개발한 온톨로지 분석 및 오류검출은, (그림 4)에 나타낸 바와같이, 다음과 같은 절차에 의해 처리된다.

3.1.1 Ontology Context Model구성

분석대상으로 지정된 온톨로지 소스코드로부터 Jena를 사용하여, 해당 온톨로지의 핵심요소들을 추출하여 5개의 context들(<표 2>~<표 6>)로 구성되는 Ontology Context Model을 구성한다.(<표 2>~<표 6>은 Protégé에서 제공하는 car reservation 온톨로지로부터 추출되었다)

3.1.2 형식개념분석기법의 적용

추출된 5개의 context(<표 2>~<표 6>)들은 형식개념분석기법의 입력으로 제공되어, 2.2절에서 설명한 바와같이, 형식개념들과 개념들간의 상하위관계를 추출/파악하여 개념격자를 구성함으로써 온톨로지에 내재된 개념계층구조를 분석한다(그림 6). 여기서 분석되는 개념계층구조로서는, is-a관

계에 대한 개념격자(그림 7), has-property에 대한 개념격자(그림 8), instance-of에 대한 개념격자(그림 12), 프로퍼티타입에 대한 개념격자(그림 13), Object프로퍼티에 대한 개념격자(그림 14) 등이다.

3.1.3 오류검출

또한, 앞서 수행한 형식개념분석기법의 적용결과로부터 구성된 각 개념계층구조를 토대로, 본 연구에서 개발한 오류검출모듈을 적용하여 온톨로지에 내재된 오류들을 검출한다(그림 9, 10, 11).

3.2 온톨로지의 오류

여기서는, Gómez-Pérez의 연구결과[16]를 토대로, 온톨로지를 개발하면서 발생할 수 있는 오류들의 유형을 소개하고, 본 연구에서 주목하고 있는 형식개념분석에 의한 기본적인 3가지 오류들(subClassOf관계 오류, 프로퍼티 오류, individual의 오류)을 명확하게 정의한다. 이와 같은 정의들을 토대로, 온톨로지 소스코드를 Ontology Context Model로 변환하여 개념들을 추출하고 계층 구조화하여 분석함으로써, 온톨로지에 존재하는 구조적인 오류들을 수월하게 파악할 수 있다.

3.2.1 온톨로지의 오류유형

Gómez-Pérez의 연구[16]에서는 도메인 지식을 대상으로 개념들을 분류/체계화하여 온톨로지를 개발할 때 발생할 수 있는 오류들을 아래와 같이 3가지 유형화하고 있다.

3.2.1.1 불일치형 오류

- **순환 오류** : subclass-of관계에 의해 구성되는 계승구조상에 순환구조가 존재할 경우에 발생.
- **분할 오류** : subclass-of관계에 의해 개념들을 분할하여 계층화하는 경우에 발생.
- **의미 오류** : 의미적으로 올바르지 않게 분류했을 경우에 발생.

3.2.1.2 불완전형 오류

- **불완전한 개념 분류** : 도메인에 존재하는 개념들 전부를 고려하지 못하고 일부만을 토대로 분류하여 정의하는 경우에 발생.
- **서브클래스 분할 생략** : 어떤 클래스를 분할하여 서브클래스들을 정의할 때, 각 서브클래스들이 서로 분리(disjoint)되어 있다는 제약사항을 고려하지 않고 분할하는 경우에 발생.
- **망라적인 서브클래스 분할 생략** : 임의의 클래스에 대하여 분할하여 서브클래스들을 정의할 때, 새롭게 정의되는 서브클래스들이 갖추어야할 속성정보를 완전하게 정의하지 못하는 경우에 발생.

3.2.1.3 중복형 오류

- **문법적인 중복 오류** : 클래스들 사이에 1개 이상의 중복된 subclass-of관계가 존재하는 경우에 발생하거나, 또는 인스턴스와 클래스 사이의 instance-of관계가 중복하여 정의된 경우에 발생.
- **클래스 정의의 중복 오류** : 온톨로지 내에 같은 정의를 갖는 클래스들이 2개 이상 존재하는 경우에 발생.
- **인스턴스 정의의 중복 오류** : 온톨로지 내에 같은 정의를 갖는 인스턴스들이 2개 이상 존재하는 경우에 발생.

위와같은 오류들의 유형은, 온톨로지를 평가하기 위한 기본적인 척도들(일관성, 완전성, 간결성, 확장성 등)을 기반으로 정의되었다. 한편, 본 연구에서는 온톨로지의 기본요소들을 대상으로 형식개념분석기법을 적용하여 추출된 개념들을 기반으로, 기본적인 3가지 오류들(subClassOf관계 오류, 프로퍼티 오류, individual의 오류)을 정의한다.

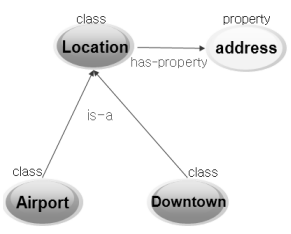
3.2.2 형식개념분석을 기반으로 정의된 온톨로지 오류

3.2.2.1 subClassOf관계 오류

온톨로지에서 가장 중요하고, 핵심이 되는 요소는 클래스(Class)이다. 클래스는 클래스가 가지는 프로퍼티(Property)에 의해서 표현되어 진다. 따라서 특정 개념을 클래스로 표현한다면 개념이 가지고 있는 속성을 잘 파악하여, 클래스에 프로퍼티로 정의함으로써, 도메인의 정보가 제대로 반영된, 유용한 온톨로지가 될 수 있다. 그러나 이러한 기본적인 원칙을 간과한다면, 클래스는 물론, 클래스의 계층구조에도 문제가 발생할 가능성이 높다.

subClassOf관계 오류란 온톨로지의 가장 핵심요소인 클래스들 사이에 상-하위관계를 클래스가 갖는 속성을 고려하지 않고, 형식적으로만 subClassOf관계(is-a관계)를 명시한 상태를 말한다. 하위클래스는 상위클래스의 프로퍼티를 상속받고, 상위클래스와 구분이 되는 프로퍼티를 추가적으로 선언하거나, 프로퍼티의 range를 더 상위개념으로 선언함으로써 프로퍼티 중심적인 디자인관점에서 상-하위 클래스사이에 엄밀한 포섭관계(Subsumption)가 성립되어야 한다. 그러나 대부분의 온톨로지는 매우 복잡하고 사람에 의해서 수작업으로 구축되어지므로, subClassOf관계 오류가 빈번하게 발생한다.

예를 들어 (그림 3)의 Location은 하위클래스로서 Airport, Downtown클래스가 선언되어 있다. 그러나, Airport와 Downtown 클래스들은 Location의 하위클래스로서, Location과 구분이 되는 속성을 갖지 않기 때문에 subClassOf관계 오류가 발생한다. subClassOf관계 오류의 정형화된 정의는 다음과 같다.




```
<owl:Class rdf:ID="Location"/>
<owl:Class rdf:ID="Downtown">
  <rdfs:subClassOf rdf:resource="#Location"/>
</owl:Class>
<owl:Class rdf:ID="Airport">
 <rdfs:subClassOf rdf:resource="#Location"/>
</owl:Class>
 <owl:DatatypeProperty rdf:ID="address">
   <rdfs:domain rdf:resource="#Location"/>
   <rdfs:range rdf:resource="http://www.w3.org/
              2001/XMLSchema#string"/>
</owl:DatatypeProperty>
```

(그림 3) 온톨로지의 is-a관계에 대한 예제

[정의 3] 임의의 온톨로지O에 포함되는 클래스들의 집합을 C라고 하자. 임의의 C_1, $C_2 \in C$에 대해서, C_1이 C_2와 subClassOf관계로 명시되어 있을 때 (C_1이 C_2의 subClass), C_1이 갖는 프로퍼티집합 P_1과 C_2가 갖는 프로퍼티집합 P_2가 서로 같다면 C_1과 C_2사이에는 "subClassOf관계 오류가 존재한다."라고 한다. ■

3.2.2.2 프로퍼티 오류

온톨로지의 프로퍼티는 기본적으로 domain과 range정보로 구성된다. 따라서 프로퍼티를 정의하기 위해서는 반드시 domain과 range항목을 명확하게 기술해야 한다. domain에는 해당 프로퍼티가 선언되어진 온톨로지 또는 참조되어지는 온톨로지의 클래스가 기술되어야 하며, range에는 클래스 또는 온톨로지에서 지원하는 데이터타입을 선언해야 한다. 그러나 많은 온톨로지에서는 프로퍼티의 domain과 range 속성을 간과하고, 형식적으로만 프로퍼티를 선언하는 경우가 많다. 프로퍼티의 오류에 대한 정형화된 정의는 다음과 같다.

[정의 4] 임의의 온톨로지O에 포함되는 클래스들의 집합 C, 프로퍼티들의 집합P, O에서 지원하는 데이터타입을 D라고 하자. 임의의 $P_1 \in P$에 대해서, $domain(p_1) \notin C$ 이거나 $range(p_1) \notin (C \cup D)$라면, P_1에는 "프로퍼티 오류가 존재한다."라고 정의한다. 단, $domain(p_1)$과 $range(p_1)$은, 각각 p_1의 domain과 range이다. ■

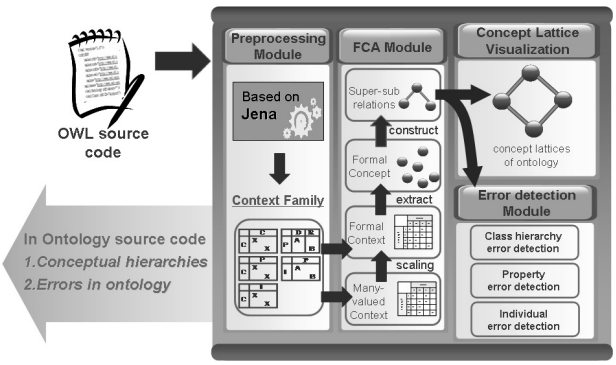
3.2.2.3 Individual 오류

온톨로지에 선언되어진 individual은 기본적으로 individual이 포함되는 클래스가 갖는 프로퍼티들의 range에 정의된 데이터타입의 데이터 값 또는 다른 individual을 참조해야 한다. 그러나 많은 온톨로지에서는 individual을 선언함에 있어서, individual이 속하는 클래스의 프로퍼티 타입에 적합하지 않는 데이터 값을 할당하여 individual의 오류를 발생시킨다. Individual의 오류에 대한 정형화된 정의는 다음과 같다.

[정의 5] 임의의 온톨로지O에 포함되는 Individual들의 집합을 I, 클래스들의 집합을 C라고 하자. 임의의 $I_1 \in I$, $C_1 \in C$에 대해서, I_1이 C_1과 instance-of 관계로 명시 되어있고(I_1이 C_1의 instance), C_1이 갖는 속성집합이 P일 때, I_1이 $P_1 \in P$의 데이터타입을 만족하지 않는 데이터 값을 갖는 경우 "I_1은 individual 오류가 존재한다."라고 한다. ■

3.3 온톨로지 분석 및 오류검출도구

본 연구에서 개발한 온톨로지 분석 및 오류검출도구는, (그림 4)와 같이 4개의 모듈로 구성되어 있으며, 온톨로지 소스코드를 입력으로하여 온톨로지의 개념구조를 가시화하고 온톨로지에 내재된 에러들을 검출한다.



(그림 4) 본 연구에서 개발한 온톨로지 분석도구의 구성

3.3.1 전처리 모듈

전처리모듈은 입력된 온톨로지 소스코드로부터 Jena[17]라는 온톨로지 추론도구를 사용하여 온톨로지의 주요요소(클래스, 프로퍼티, individual)들 및 요소들 사이의 다양한 관계들(is-a, instance-of, has-property 등)을 추출하여 5종류의 Context로 구성된 Ontology Context Model을 작성하는 모듈이다.

3.3.2 형식개념분석 모듈

2.2절에서 언급한 형식개념분석기법의 제반정의를 구현한 모듈로서, 전처리 모듈에서 생성된 Ontology Context Model을 입력으로하여 개념격자형태의 온톨로지 개념구조들을 구축하는 모듈이다. 형식개념분석 모듈에서 구축되어지는 개념격자를 자세히 분석함으로써, 온톨로지의 상세정보 및 구조적인 분석이 가능하다.

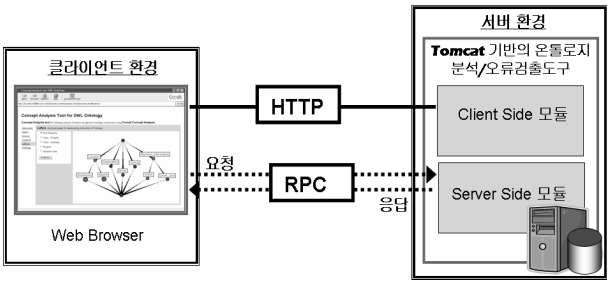
3.3.3 개념격자 가시화모듈

개념분석기법 모듈의 결과물인 개념격자들을 JPowerGraph[18]라는 그래프라이브러리를 사용하여, 사용자에게 그래픽적인 표현을 제공한다.

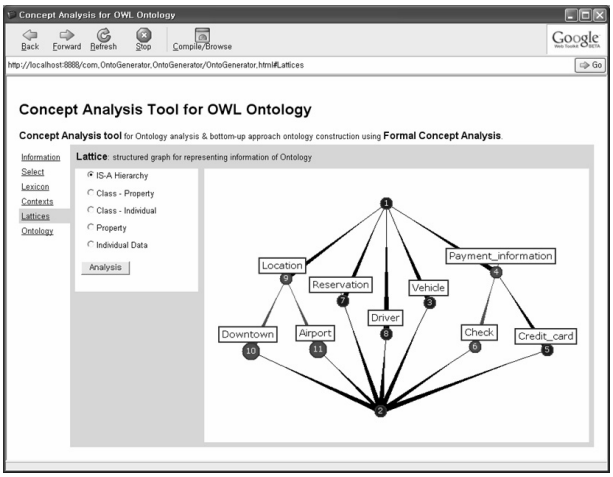
3.3.4 에러검출 모듈

에러검출 모듈은 개념격자로 표현된 온톨로지의 정보를 3.2.2절에서 정의한 온톨로지의 3가지 구조적인 오류를 자동으로 검사하여 추출하는 모듈이다.

시스템적인 측면에서, 온톨로지 분석/오류검출도구는 AJAX[19] 기반의 웹 어플리케이션이다. 따라서 사용자는 웹브라우저를 이용하여 분석할 온톨로지의 파일을 서버에 업로드하고, 서버는 업로드 된 파일을 기반으로 분석된 온톨로지의 개념구조와 오류들에 대한 정보를 클라이언트로 전송한다. 본 시스템에서는 이러한 일련의 과정을 RPC(Remote Procedure Calls) 메커니즘을 이용하여 비동기식으로 전송함으로써, 불필요한 웹 페이지의 로딩을 피하고, 필요최소의 데이터만을 전송한다. (그림 5)에서 서버환경의 Client Side 모듈은 클라이언트의 User Interface를 위한 html페이지를 의미하며, Server Side 모듈은 온톨로지의 개념구조를 추출하고, 오류를 검출하는 일련의 과정을 RPC메커니즘으로 구현하였다.



(그림 5) 본 연구에서 개발한 온톨로지 분석도구의 시스템구조



(그림 6) 온톨로지 분석 및 오류검출도구 실행화면

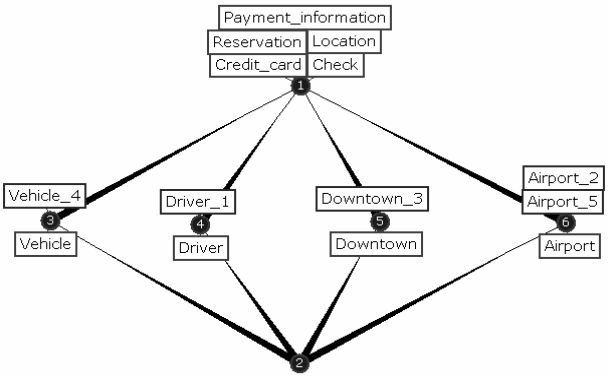
온톨로지 분석 및 오류검출도구는 다음과 같은 순서에 의해서 실행된다(그림 6).

1. 웹브라우저를 이용하여 온톨로지 분석 및 오류검출도구에 접속하여 분석하고자 하는 온톨로지 파일을 서버로 업로드 한다.
2. 사용자가 전송한 온톨로지의 소스코드를 토대로, 서버에서는 Ontology Context Model을 작성하고, 개념계층구조를 생성하여 이를 토대로 온톨로지의 구조적 오류를 검사한다.
3. 온톨로지의 개념구조와 더불어서, 검출된 오류들을 사용자에게 가시화하여 제공한다.

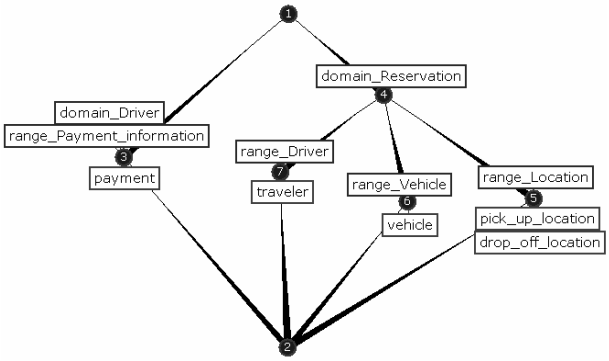
4. 온톨로지 분석 및 오류검출 사례

Prompt[20]의 튜토리얼 예제로 제공되는 온톨로지들(Car Reservation온톨로지, Pizza온톨로지, Travel온톨로지)을 대상으로, 본 연구에서 개발한 온톨로지 분석도구를 사용하여 분석하고 구조적인 오류를 검출하는 사례를 설명한다.

3.2.2절에서 설명한 온톨로지의 구조적 오류를 파악하기 위해 사용된 개념구조뿐만 아니라, 사용되지 않은 개념구조들도 온톨로지의 의미 있는 정보를 표현하고 있다. instance-of관계를 표현한 개념격자를 이용하여 온톨로지의 모든 individual



(그림 7) Car Reservation온톨로지의 instance-of에 대한 개념격자

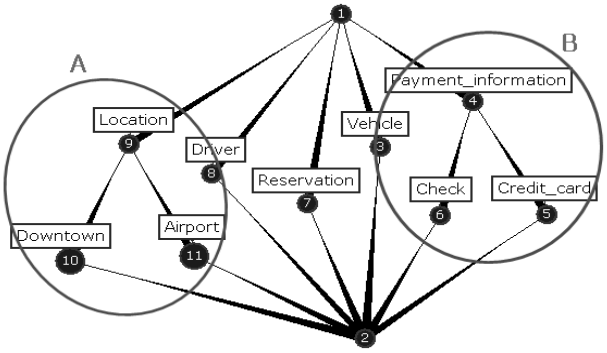


(그림 8) Car Reservation 온톨로지의 Object프로퍼티에 대한 개념격자

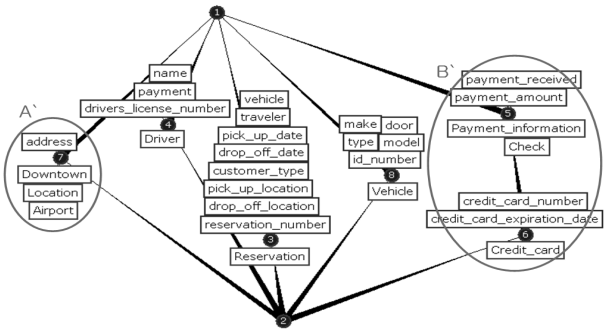
들을 해당 클래스별로 클러스터링한 정보를 파악할 수 있다. 예를 들어 (그림 7)은 Car Reservation 온톨로지의 instance-of 관계를 표현한 개념격자로서, Airport에 속하는 individual(Airport_2, Airport_5)들, 그리고 Vehicle, Driver, Downtown클래스에 각각 한 개의 individual들 속하고 있음을 파악할 수 있다.

프로퍼티를 표현한 Many-valued Context로부터 Range속성만을 스케일링하여 구축된 개념격자는, 프로퍼티를 데이터 타입별로 분류하여 분석해주고 있다. 예를 들어 (그림 8)은 Car Reservation 온톨로지의 프로퍼티들을 데이터 타입별로 분석한 개념격자이다. 그림13에서 int형 타입의 프로퍼티는 credit_card_ number, reservation_number, door임을 알 수 있다. 또한, Car Reservation 온톨로지에는 4가지 클래스(Driver, Location, Vehicle, Payment_information)타입과 4가지 primitive데이터 타입(string, boolean, float, int)들의 프로퍼티들이 정의되어 있음을 파악할 수 있다.

Object프로퍼티의 domain과 range속성을 스케일링하여 생성된 개념구조는 클래스들 사이에 association관계를 표현하고 있다. (그림 9)는 Car Reservation 온톨로지의 Object프로퍼터에 대한 개념격자이다. 예를 들어 Payment_information 클래스와 Driver클래스는 payment라는 object프로퍼티에 의해 연관관계를 맺고 있다. 또한 Location, Driver, Vehicle클래스들은 공통적으로 Reservation클래스와 연관관계가 존재함을 파악할 수 있다.



(그림 9) Car reservation온톨로지의 is-a관계에 대한 개념격자



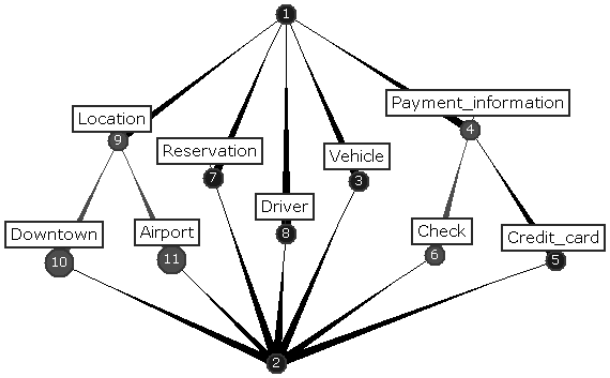
(그림 10) Car reservation온톨로지의 has-property에 대한 개념격자

본 연구에서 개발된 온톨로지 분석도구를 실행하여 얻어진 (그림 7~9)의 각 개념계층구조를 토대로, 다음과 같이 온톨로지의 오류들을 검출한다.

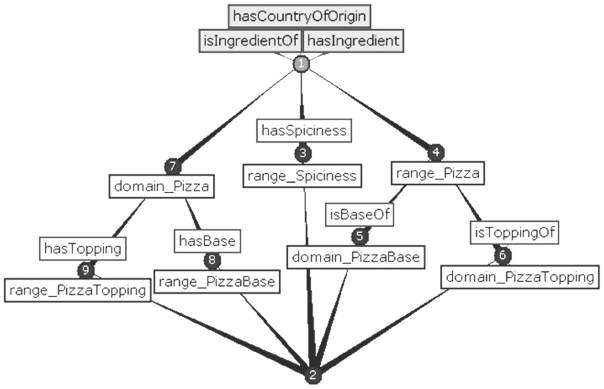
4.1 subClassOf관계 오류

3.2.2절에서 언급한 온톨로지 subClassOf관계오류는 온톨로지의 is-a관계에 대한 개념구조를 나타내는 (그림 10)과 온톨로지의 has-property관계에 대한 개념구조를 표현한 (그림 11)을 비교함으로써 문제를 파악할 수 있다. 클래스들 사이에 is-a관계를 표현하고 있는 (그림 10)의 A부분에서 Location 클래스의 하위클래스로 Downtown과 Airport클래스가 존재하지만, 클래스가 갖는 프로퍼티를 표현하고 있는 (그림 11)의 A`부분에서는 3개의 클래스(Location, Downtown, Airport)들이 모두 address프로퍼티만 정의하고 있기 때문에 같은 개념에 속하고 있다. 따라서 온톨로지 개발자가 의도한 대로 온톨로지의 구조가 성립 하기위해서, Airport, Downtown 클래스들에 각각의 클래스가 갖는 새로운 프로퍼티가 정의되어야 한다. 또한 (그림 10)의 B부분과 (그림 11)의 B`부분을 비교함으로써 subClassOf관계 오류를 검출할 수 있다.

(그림 12)는 본 연구에서 개발된 도구를 사용해서 클래스 계층구조를 자동으로 추출하여 가시화한 모습이다. 붉게 표시된 노드와 에지들이 현재 문제가 되고 있는 subClassOf관계 오류를 나타낸다. 현재 Car Reservation 온톨로지에는 Location - Downtown, Location - Airport 그리고 Payment_information - Check클래스사이에 is-a관계가 엄밀한 포섭관



(그림 11) Car Reservation온톨로지의 subClassOf관계 오류 검출



(그림 12) Pizza 온톨로지의 프로퍼티 오류검출

계에 의해서 정의되지 않았음을 파악할 수 있다.

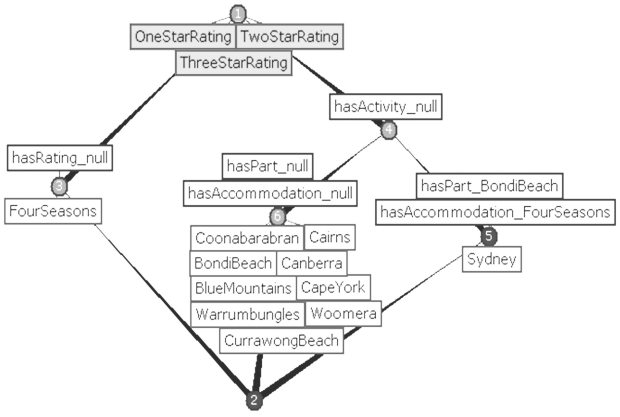
4.2 프로퍼티 오류

Pizza 온톨로지에 대하여, 프로퍼티의 개념구조를 표현한 개념격자를 분석하여 프로퍼티의 오류를 파악할 수 있다. 프로퍼티의 개념격자에 포함된 개념들에서 intent정보를 갖지 않는 개념들을 추출함으로써, 온톨로지에 프로퍼티 중에서 domain과 range속성을 갖고 있지 않거나 비어있는 프로퍼티들을 추출할 수 있다.

(그림 13)은 Pizza 온톨로지의 프로퍼티에 관한 개념격자이다. 최상위개념은 intent정보를 갖고 있지 않기 때문에, 최상위개념에 속하는 3개의 프로퍼티(hasCountryOfOrigin, hasIngredient, isIngredientOf)들은 오류가 있음을 파악할 수 있다. 또한, (그림 10)의 개념격자로부터 프로퍼티들에 공통적인 특징을 파악할 수 있다. 예를 들어 hasTopping과 hasBase프로퍼티들은 domain속성으로 Pizza클래스가 공통적으로 포함되며, 또한 isBaseOf와 isToppingOf프로퍼티들은 range속성으로 Pizza클래스가 공통적으로 포함된다.

4.3 Individual 오류

Travel온톨로지에 대하여, individual의 개념구조를 표현한 개념격자를 분석하여 individual의 오류를 파악할 수 있다. individual의 개념격자에 포함된 개념들 중에서 intent정



(그림 13) Travel온톨로지의 individual 오류검출

보를 갖지 않는 개념들을 추출함으로써, 온톨로지에 정의된 individual중 해당 클래스의 프로퍼티에 적합한 데이터 값을 제대로 정의하지 않는 individual들을 검출할 수 있다. (그림14)에서는 Travel온톨로지에는 3개의 individual(OneStarRating, ThreeStarRating, TwoStarRating)들에 오류가 있음을 파악할 수 있다.

이미, Protégé와 같은 온톨로지 도구들에서는 다양한 평가척도를 기반으로하는 오류체크 기능들과 온톨로지의 논리적 구조를 분석해서 해당 온톨로지가 형식적으로 잘 구성되어 있는지를 판단하는 일관성 체크 기능 및 재구성하는 기능 등을 제공하고 있다[12]. 한편, 본 연구에서 제안한 기법과 지원도구에서는, 형식개념분석기법을 이용하여 온톨로지로부터 “개념”과 “개념계층구조”를 추출하여 온톨로지의 핵심요소들을 분석하고 3.2.2절에서 제안한 3가지 오류를 검사하고 있다. 따라서, 기존의 온톨로지 도구들에서 제공하는 오류검출기능과 본 연구에서 개발한 오류검사기능은, 온톨로지에 대한 분석방법과 오류에 대한 정의에 내재된 근본적인 개념이 서로 다르다.

이상의 분석사례와 더불어서, 기존의 온톨로지 구축도구(Protégé, Hozo, ez-owl, KAON) [12, 21, 22, 23]에서 제공하고 있는 온톨로지들 중에서 표8과 같이 임의로 18개의 온톨로지들을 대상으로 하여, 본 논문에서 개발한 온톨로지 분석 및 오류추출도구를 사용하여 온톨로지 분석과 오류검사를 실시하였다. 이와같은 실험에서는, 기존의 도구들과 본 연구에서 개발된 도구의 성능이나 기능을 비교평가하기 위한 것이 아니며, 다만, 본 연구에서 개발된 도구의 오류검출기능을 보다 철저하게 시험해 보기 위하여, 다양한 도구들에 의해 구축된 온톨로지들(기존의 각 도구들에서 예제 형태로 제공하는 온톨로지들)을 대상으로 오류검사를 실시하였다.

<표 7>과 같이, Protégé에서 제공한 7개의 온톨로지중에서 subClassOf관계 오류는 7개의 온톨로지에서, 프로퍼티 오류는 3개, individual의 오류는 3개의 온톨로지에서 검출되었다. 또한 Hozo에서 제공한 4개의 온톨로지 중 subClassOf관계 오류는 4개, 프로퍼티 오류는 4개의 온톨로지에서 검출되었으며 individual의 오류는 검출되지 않았다. 나머지 ez-owl과 KAON에서 예제로 제공하고 있는 각각의 온톨로지들에

〈표 7〉 온톨로지들에 대한 오류검출 실험결과

온톨로지	subClassOf관계 오류	프로퍼티 오류	individual 오류	비고
air_reservation	○	×	×	Protégé에서 제공
car_reservation	○	×	×	
pizza	○	○	○	
wine	○	○	×	
biopax-level1	○	×	×	
people+pets	○	○	○	
travel	○	×	○	
bike_eng	○	○	×	Hozo에서 제공
travelling	○	○	×	
TOofSt_Ex	○	○	×	
vehicle	○	○	×	
ontologies	○	○	×	ez-owl에서 제공
delegation	○	○	×	
fsm	○	○	○	
publication	○	○	×	KAON에서 제공
jaguar	○	×	○	
acm_ccs	○	○	○	
kaonportal	○	○	○	

대한 오류검사결과는 <표 7>과 같다.

5. 결 론

온톨로지 개발에 있어서 대상 도메인이 광범위하고 복잡하며 방대한 용량의 정보를 포함하는 경우에는, 다종다양한 오류들이 포함될 수 있다. 따라서, 다양한 측면에서 온톨로지에 대한 분석 및 오류 검출에 관한 제반 기법과 지원도구들을 제공함으로써 온톨로지 개발자들에게 많은 도움이 될 수 있다.

본 논문에서는, 형식개념분석기법을 이용하여, 온톨로지의 소스코드로부터 핵심요소들을 추출/분석하여 오류를 검출하기 위한 기법과 지원도구를 개발하였다. 구체적으로는, (1)온톨로지 소스코드(OWL 또는 RDF)로부터 온톨로지를 구성하는 핵심요소들을 추출하여, (2)이를 형식개념분석기법에 적용시키기 위한 데이터모델(Ontology Context Model)을 정의하였고, (3)해당 온톨로지의 다양한 개념구조(Concept Hierarchy)들을 파악하기 위한 제반기법들을 제안, (4)해당 온톨로지의 각 요소들의 상세한 정보를 추출/분석하여 온톨로지에 내재되어 있는 구조적인 오류들을 자동으로 검출하기위한 지원도구를 개발하였다.

본 연구결과는, 기존의 온톨로지를 수정보완할 경우의 지침으로서 뿐만아니라, 보다 좋은 개념계층구조를 갖는 고품질의 온톨로지를 개발할 수 있는 유용한 정보를 제공하는 도구로서 이용될 수 있다. 앞으로의 연구과제로서, 본 논문에서 정의된 오류들(정의 3~5)은 실제로 온톨로지의 subclass관계, property관계, 그리고 individual관계에 대한 부분적인 사례를 대상으로 오류를 검출할 뿐, 이들 관계를 완전하게 커버하지 못하는 한계점이 있다. 따라서, 이러한 한계점을 인식하고 개선하기 위하여 향후연구에서는, 본 연구결과를 토대로 보다 다양한 오류를 검출할 수 있도록 개선하고, 보다 정교하고 상세한 온톨로지의 분석을 위하여 다양한 온톨로지의 요소들(Axiom, Constraint, Cardinality 등)을 고려하여 Ontology Context Model을 확장하고, 제반분석기능을 추가할 필요가 있다. 특히, 본 논문에서 개발한 도구를 이용하여 분석한 결과와 도메인 전문가가 분석했을 때의 차이 여부 및 정확도 등에 관하여 보다 상세한 연구가 진행되어야 할 것이다.

참 고 문 헌

[1] T. R. Gruber, “Toward principles for the design of ontologies use for knowledge sharing,” the Padua workshop on Formal Ontology, March, 1993.

[2] J. Heflin and J. Hendler, “Semantic Interoperability on the web,” Proceedings of Extreme Markup Language 2000. Graphic Communications Association, pp.111-120, 2000.

[3] A. Gomez-Perez and O. Corcho, “Ontology languages for the Semantic Web,” IEEE Intelligent Systems, Vol.17, No.1, pp.54-60, January/February, 2002.

[4] E. P. B. Simperl and C. Tempich, “Ontology Engineering: A Reality Check,” OTM Conferences, LNCS 4275, 2006.

[5] J. Cardoso, “The Semantic Web Vision: Where Are We?,” IEEE Intelligent Systems, Vol.22, No.5, pp.84-88, 2007.

[6] Q. Lu and V. Haarslev, “OntoKBEval: A Support Tool for DL-based Evaluation of OWL Ontologies,” Proceedings of the 2006 International Workshop on OWL: Experiences and Directions 2006(OWLED-2006), Athens, Georgia, USA, Nov. 10-11, 2006.

[7] D. Sleeman and Q. Reul, “CleanONTO: Evaluating Taxonomic Relationships in Ontologies,” Proceedings of the 4th International EON Workshop, Edinburgh International Conference Center, Edinburgh, United Kingdom, May, 22nd, 2006.

[8] Y. Sure and M. Erdmann and J. Angele and S. Staab and R. Studer and D. Wenke, “OntoEdit: Collaborative Ontology Development for the Semantic Web,” Proceedings of the First Semantic Web Conference, June, 2002.

[9] J. Angele and Y. Sure, “EFFORT-evaluation framework for ontologies and related technologies,” Technical report, Institute AIFB, University of Karlsruhe and Ontoprise GmbH, 2002.

[10] OWLDoc, http://www.co-ode.org/dowmloads/owldoc/co-ode-index.php

[11] G. Jiang and et al. “FCAViewTab: A concept-oriented view generation tool for clinical data using formal concept analysis,” Proceedings of the 8th International Protégé Conference, Madrid, Spain, July, 2005.

[12] Protégé : http://protege.stanford.edu/

[13] RDF_Primer : http://www.w3.org/TR/rdf-primer/

[14] OWL_Overview : http://www.w3.org/TR/owl-features/

[15] B. Ganter and R. Wille, ‘Formal Concept Analysis: Mathematical Foundations,’ Springer-Verlag, 1999.

[16] A. Gómez-Pérez, “Ontology Evaluation,” Handbook on Ontologies, Springer-Verlag, pp.251-274, 2003.

[17] Jena (A Semantic Web Framework for Java) : http://jena.sourceforge.net.

[18] JPowerGraph, http://sourceforge.net/projects/jpowergraph/

[19] AJAX, http://www.w3schools.com/ajax/default.asp

[20] Prompt, http://protege.stanford.edu/plugins/prompt/prompt.html

[21] Hozo, http://www.ei.sanken.osaka-u.ac.jp/hozo/eng/index_en.php.

[22] ez-owl, http://iweb.etri.re.kr/ezowl

[23] KAON, http://kaon.semanticweb.org

황 석 형

e-mail : shwang@sunmoon.ac.kr

1991년 강원대학교 전자계산학과 조기졸업 (이학사)

1993년 일본 오사카대학교 대학원 정보공학과 (공학석사)

1997년 일본 오사카대학교 대학원 정보공학과 (공학박사)

1997년~현 재 선문대학교 컴퓨터공학부 부교수

2007년 1월~2008년 1월 아일랜드국립대학교 DERI 객원연구원

관심분야 : 소프트웨어공학, 객체지향, 온톨로지공학, Formal Concept Analysis, 시맨틱 웹